Reconocimiento de Patrones

Joaquín Salas
CICATA Querétaro
Instituto Politécnico Nacional
Cerro Blanco 141, Colinas del Cimatario, 76090,
Querétaro, México
jsalasr@ipn.mx, joaquin.salas@gmail.com

Agosto, 2018

Índice general

Prefacio

Posiblemente, la facultad de reconocer patrones es la experiencia que más cercanamente asociamos con el concepto de inteligencia. Esta capacidad está tan embebida en nosotros que la apreciamos de forma natural a nuestro alrededor, sea en un congenere cuando le asociamos un cierto comportamiento, en una mascota cuando aprende un nuevo truco, en el ir y venir de las estaciones del año. Los patrones nos dan estructura sobre lo permanente, al mismo tiempo que abren una puerta para interpretar cuando algo se sale de la norma. Mediante los patrones aprendemos a relacionarnos, adquirimos conocimiento, comprendemos las cosas. El estudio del reconocimiento de patrones es emocionante pues es una manera de entendernos mejor a nosotros mismos.

Aprendizaje e Inferencia

El estudio del reconocimiento de patrones también es importante en una variedad de cuestiones prácticas. Hoy día, computadoras cada vez más veloces y con mayor capacidad de almacenamiento están posibilitando su empleo como herramientas para volvernos más productivos. Mediante el reconocimiento de patrones podemos explorar la inmensa cantidad de datos que se produce diariamente y obtener información para realizar de mejor manera nuestras actividades.

Por la facilidad que se tiene para examinar su contenido, posiblemente por el mapeo lineal-no lineal que resulta, Lecun y Ranzato[36] distinguen entre técnicas someras y profundas. Ejemplo de las primeras técnicas con los árboles de decisión, las máquinas de soporte vectorial (SVM), boosting. Ejemplos de técnicas profundas incluyen las redes neuronales recurrentes y las redes neuronales convolucionales. Enmedio, se tienen técnicas tales como las máquinas de Boltzman o los métodos de Bayes no paramétricos. Lecun y Ranzato también distinguen entre los métodos basados en redes neuronales y los métodos probabilísticos. Es un mantra en el área que no hay método de reconocimiento de patrones que sea el mejor para todos los casos[70]. Por ello, con la finalidad de proporcionar al estudiante un conjunto de herramientas modernas y eféctivas, en este curso tenemos como meta dar un vistazo, con un enfoque mediano sobre los aspectos técnicos sobre los cuales se basan, a todos estos métodos.

Otra forma de organizar el estudio de métodos de reconocimiento de patrones es entre métodos de aprendizaje supervisados, no supervisados y de refuerzo. En el aprendizaje supervisado tenemos un conjunto de datos sobre los cuales conocemos que entradas producen que salidas. La tarea de aprendizaje es construir el mejor mapeo de unas a las otras. Sin embargo, la construcción de este mapeo es muy laborioso, tardado y propenso a errores. Esta circunstancia se vuelve más crítica cuando nos damos cuenta que estamos rodeados de muchísimas más cosas que requiren etiquetado manual. Por ello, el aprendizaje no supervisado se vuelve atractivo. Solo imagine las posibilidades que abren las millones y millones

de imágenes que se encuentran en el internet y la multitud de aplicaciones que de ellas se desprenden al sacarles provecho a partir del uso de computadoras cada vez más veloces. Por último, en aprendizaje mediante refuerzo damos una recompensa, positiva o negativa, a cada decisión que el algoritmo toma, de forma muy similar a como entrenamos a las mascotas. Como en esa analogía resulta muy importante poder correlacionar y sincronizar de la mejor manera el estímulo correcto en el tiempo preciso. En este curso, solo estudiaremos técnicas relacionadas con aprendizaje supervisado y no supervisado.

Con el tiempo, la disciplina se ha vuelto más amplia, las técnicas más elaboradas. Hoy en día, vemos casos de técnicas sobre humanas, en el sentido que su nivel de desempeño sobrepasa el que puede exhibir un humano[18]. Esto tiene importantes implicaciones sociales que rápidamente se han comenzado a experimentar. En ese sentido, el conocimiento de las técnicas brinda la posibilidad de tener más conocimiento sobre las implicaciones de las técnicas, su potencial, y las formas en las que mejor podemos aprovecharlas.

El proceso de reconocimiento de patrones se ilustra en la Figura 1. Las señales que se perciben del mundo son procesadas para extraer variables que por un lado le caracterizan y por otro son procesables por los dispositivos de cómputo. Su valor es depositado en una variable $\mathbf{x}$. Enseguida, el proceso se bifurca entre la etapa en la cual realizamos el aprendizaje (conocida como etapa de entrenamiento) y aquella en la cual realizamos la inferencia (conocida como etapa de operación). En todo caso, suponemos que contamos con un modelo del mundo, $p(\mathbf{w}|\mathbf{x}, \boldsymbol{\Theta})$, que nos permite interpretar su estado $\mathbf{w}$ mediante un conjunto de parámetros Θ y la propia caracterización $\mathbf{x}$. Justamente, el proceso de entrenamiento tiene el propósito de identificar los mejores valores de los parámetros.

El entrenamiento puede ser supervisado, no supervisado o mediante refuerzo. En aprendizaje supervisado contamos con pares $\{\mathbf{x}, \mathbf{w}\}$ que nos permite guiar el proceso. Por otro lado, en aprendizaje no supervisado esperamos que las características se asocien en grupos, de preferencia con poca variabilidad, y que estos sean diferenciables entre sí de forma clara. En aprendizaje mediante refuerzo proporcionamos un estímulo que depende de la respuesta obtenida. Aquí las dificultades incluyen la selección del tipo y oportunidad del estímulo. Por su lado, $\mathbf{w}$ puede ser de naturaleza continua o discreta. Cuando su valor es continuo estamos resolviendo un problema de regresión. Cuando es discreta estamos resolviendo un problema de clasificación. La identificación del valor de los parámetros durante el entrenamiento nos permite definir un espacio $\mathcal{C}$, que puede ser de regresión o clasificación, dependiendo del problema. Ya en operación, hay una etapa de inferencia en donde una caracterización proyectada en el espacio $\mathcal{C}$ permite la instanciación del estado que guarda el mundo $\mathbf{w}$.

En aprendizaje supervisado tenemos la oportunidad de evaluar el desempeño del algoritmo de entrenamiento. Para ello, típicamente dividimos las muestras en tres grupos. Uno de ellos nos sirve para determinar los parámetros Θ. Otro nos sirve para hacer la validación de los parámetros y posiblemente modificarlos. Finalmente, se tiene un grupo para probar. Este grupo solo se usa una vez y el resultado nos permite apreciar la capacidad de generalización del algoritmo ante casos no encontrados.

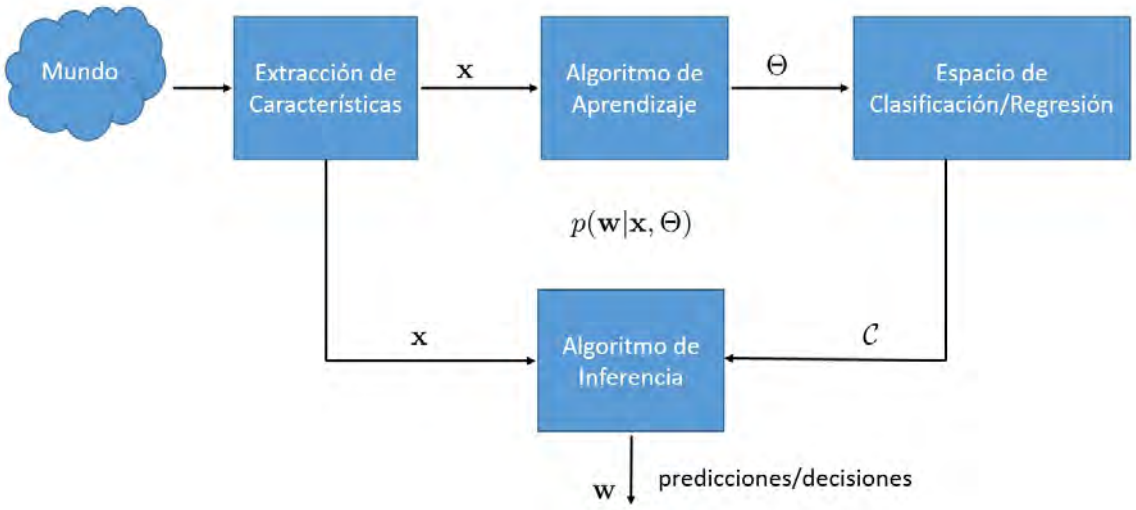


Figura 1: Reconocimiento de Patrones. Mediante mecanismos de percepción obtenemos información sobre el mundo en la forma de características $\mathbf{x}$. Durante la etapa de aprendizaje, las características nos permiten definir los valores de los parámetros $\boldsymbol{\Theta}$ tal que el modelo del mundo $p(\mathbf{w} \mid \mathbf{x}, \boldsymbol{\Theta})$ permita la inferencia del estado del mundo $\mathbf{w}$ en el espacio de clasificación $\boldsymbol{\Theta}$. Durante operación, una muestra no vista $\mathbf{x}$ es mapeada a un estado del mundo $\mathbf{w}$. Si la naturaleza de $\mathbf{w}$ es continua estamos hablando de un problema de regresión. Si $\mathbf{w}$ es discreta, el problema es de clasificación.

El Camino Adelante

En este documento nuestro objetivo es entender los fundamentos, explorar las técnicas, y adquirir dominio sobre herramientas que permitan generalizar la aplicación del reconocimiento de patrones a problemas de particulares. Para lograr lo anterior, procuramos obtener un conocimiento comprensivo de las técnicas más efectivas para realizar reconocimiento de patrones, buscamos implementar diferentes métodos de reconocimiento de patrones para la solución práctica de problemas y apuntamos a conocer, usar, y expandir algunas de las herramientas de desarrollo más importantes.

Parte I

Modelado de Datos

CAPÍTULO 1

Modelos Probabilísticos

Las técnicas derivadas de la probabilidad nos permiten asignar incertidumbre bajo principios quer permiten su manipulación y propagación. Aquí estudiamos algunos de los conceptos de la probabilidad y su uso en algunas condiciones experimentales.

1.1. Funciones de Probabilidad

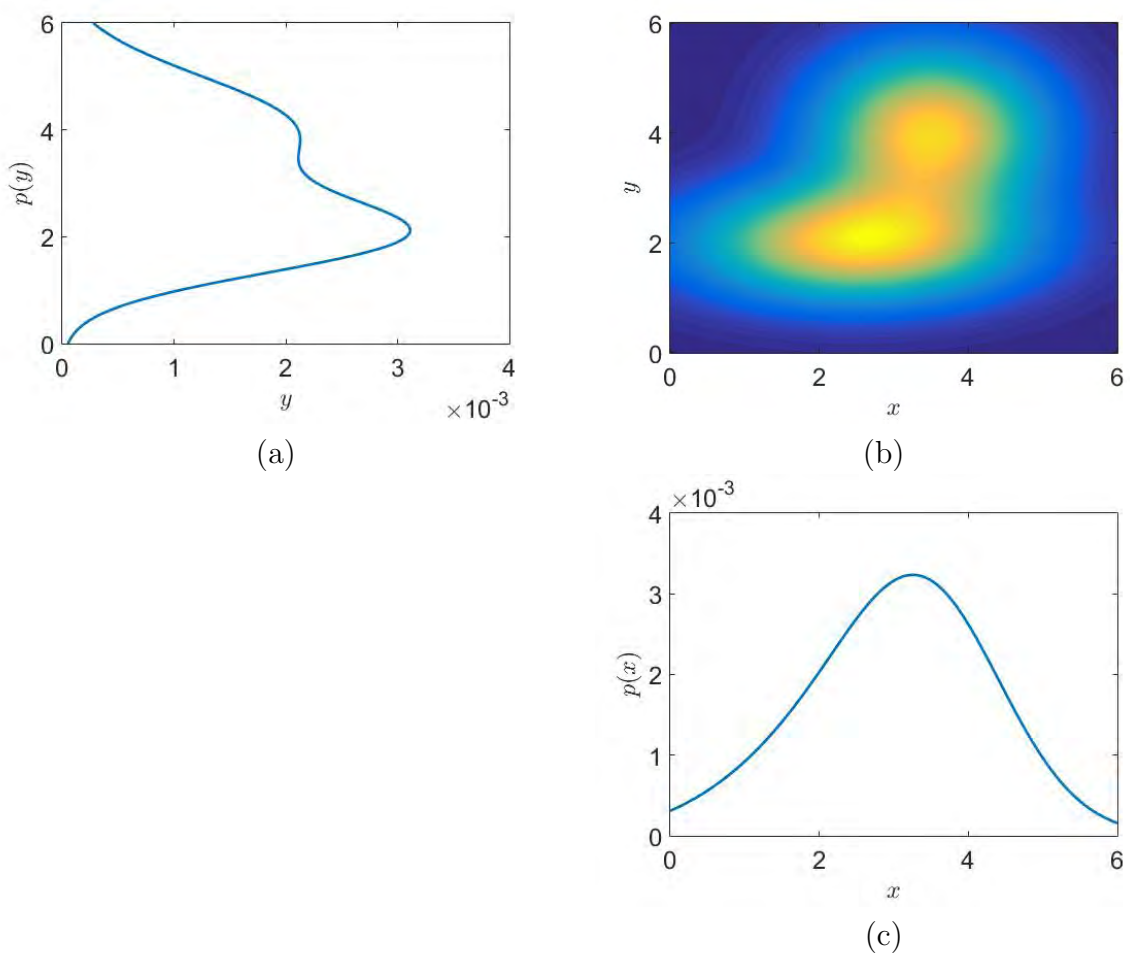
Normalmente cuando las variables se instancian contienen un valor, el cual permanence mientras no se realice con ella una operación que le altere. En contraposición, una variable aleatoria, x, cambia de valor debido al proceso azaroso al cual está ligada. En ese sentido el valor de una variable aleatoria es incierto. Las variables aleatorias pueden ser discretas o continuas. Ejemplos de valores discretos pueden ser el resultado de lanzar una moneda, *i.e.*, cara o cruz, o el resultado que observamos al lanzar un dado, *i.e.*, 1,2,..., 5,6. Algunos ejemplos de variables aleatorias continuas pudiera ser el peso o la altura de la siguiente persona con la que nos encontremos o la temperatura o velocidad del viento el día de mañana a medio día. En el caso de que la variable aleatoria sea continua esta puede ser finita o infinita.

Sin embargo, aun cuando es desconocido, la naturaleza de la variable aleatoria puede ser entendida en virtud de que hay una expectativa del conjunto de valores que puede tomar. A este conjunto de valores le conocemos como función de distribución de probabilidades (pdf), $p(x)$. Las pdf tienen dos propiedades básicas. Una de ellas es que sus valores siempre son mayores que cero. La segunda es que la suma, en el caso de que x se discreta, o su integral, cuando x es continua, suman uno. Al trabajar con probabilidades es importante mantener presente que estas reflejan una relación entre el número de eventos que nos interesa examinar y el número de eventos en el subconjunto dentro del cual esos eventos de interés se incluyen.

Una pdf puede ser multivariable, *e.g*, $p(x, y)$, en cuyo caso hace referencia a la *probabilidad conjunta* de ocurrencia de ambas x y y (ver Figura 1.1)(b). Esto nos lleva a la definición de marginalización, una operación que nos permite agrupar todo aquello sobre lo cual nuestro conocimiento es imperfecto (ver Figura 1.1). Esta operación puede ser definida como

$$p(x) = \int p(x, y)dy. \tag{1.1}$$

Cuando en una probabilidad conjunta suponemos conocido el valor de una variable, el resultado se convierte en una función de las variables que quedan. Esto es conocido como



(a) (b) (c)

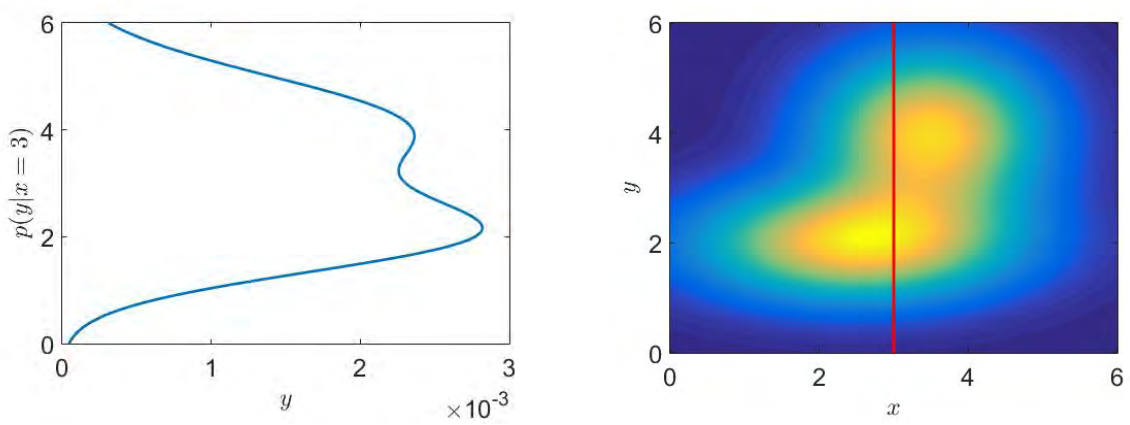
Figura 1.1: Probabilidad marginal de una pdf bivariada. En (b) se muestra la probabilidad conjunta $p(x, y)$. En (a) y (c) se muestra la probabilidad marginal $p(y) = \int p(x, y)dx$ y $p(x) = \int p(x, y)dy$, respectivamente.

probabilidad condicional, $p(x|y)$. La probabilidad condicional (ver Figura 1.2) puede ser definida como

$$p(y|x) = \frac{p(x, y)}{p(x)} = \frac{p(x, y)}{\int p(x, y)dy}. \tag{1.2}$$

En la anterior definición conviene notar que la probabilidad conjunta puede ser definida de cualquiera de las dos formas siguientes

$$\begin{aligned} p(x, y) &= p(y|x)p(x), \text{ ó} \\ &= p(x|y)p(y). \end{aligned} \tag{1.3}$$



(a) (b)

Figura 1.2: Probabilidad conditional de una pdf bivariada. En (b) se muestra que se conoce que $x = 3$. Como consecuencia, en (a) se muestra la probabilidad condicional $p(y|x = 3)$.

De donde puede definirse la regla de Bayes dada por

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\int p(y|x)p(x)dx}. \tag{1.4}$$

En la nomenclatura de Bayes, el término $p(y|x)$ es conocido como la verosimilitud, $p(x)$ es conocido como el término *a priori*, y $p(y)$ es conocida como la evidencia.

En ocasiones, el valor de y en la expresión de probabilidad condicional $p(x|y)$ no afecta la distribución de x, en cuyo caso se dice que x y y son independientes y se tiene la expresión

$$p(x|y) = p(x). \tag{1.5}$$

Las variables aleatorias pueden tomar cualquier valor, dentro del conjunto expresado por su función de distribución de probabilidades. Sin embargo, las variables tienen un valor esperado, *i.e.*, aquel valor que es más probable que ocurra. La defnición de valor esperado está dada por la expresión

$$\mathbb{E}[f(x)] = \int f(x)p(x)dx. \tag{1.6}$$

donde $f[x]$ representa la variable aleatoria y $p(x)$ su pdf. Las siguientes son algunas propiedades importantes del valor esperado.

- El valor esperado de una constante es la constante, $\mathbb{E}[k] = k$.
- El valor esperado de una constante por una variable aleatoria es la constante multiplicada por el valor esperado de la variable aleatoria, $\mathbb{E}[kf(x)] = \mathbb{E}[f(x)]$.
- El valor esperado de la suma de dos variables aleatorias es la suma del valor esperado de las variables aleatorias, $\mathbb{E}[f(x) + g(x)] = \mathbb{E}[f(x)] + \mathbb{E}[g(x)]$.
- Si x y y son independientes, eñ valor esperado de la multiplicación de dos variables aleatorias es la multiplicación del valor esperado de las variables aleatorias, $\mathbb{E}[f(x)g(x)] = \mathbb{E}[f(x)]\mathbb{E}[g(x)]$.

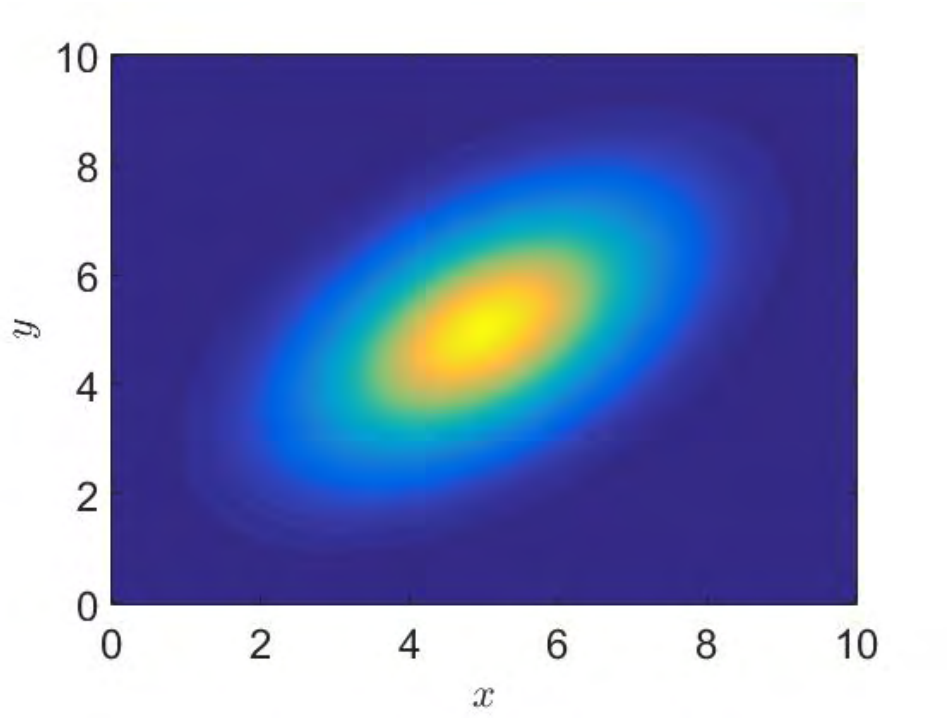


Figura 1.3: Ejemplo de distribución Gaussiana en dos dimensiones, con valor medio en μ y covarianza Σ.

1.2. Ajuste de Modelos

Una pdf describe el comportamiento de una variable aleatoria al proporcionar las probabilidad de ocurrencia de ciertos valores. En muchas ocasiones, las pdf's tienen formas muy complejas y no hay expresiones análiticas que las expliquen. En otras ocasiones, resulta conveniente aproximarlas por una expresión cerrada. Esto puede ser conveniente para propósitos de derivar propiedades, caracterizar un conjunto de datos, o simplemente por cuestiones estéticas. El ejemplo clásico es la distribución Gaussiana, cuya expresión mutivariada se da por

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left(-1/2(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right), \tag{1.7}$$

donde $\mathbf{x}$ es un vector en un espacio de D dimensiones, μ es un vector de $D \times 1$, y Σ es una matriz cuadrada, de dimensiones $D \times D$, simétrica positiva definida. El ser una matriz positiva definida implica, entre muchas otras cosas, que para cualquier vector $\mathbf{x}$, diferente de cero, el producto $\mathbf{x}^T\Sigma\mathbf{x} > 0$. La Figura 1.3 muestra un ejemplo de una distribución Gaussiana en $D = 2$ dimensiones, con valor medio en $\mu = (5,5)^T$ y covarianza $\Sigma = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$. Para describir todos los puntos bajo ella, únicamente requerimos $D + D(D+1)/2$ parámetros, D por la media y $D(D+1)/2$ por la covarianza. Por la misma razón que uno escoge una expresión cerrada para la pdf, uno puede seleccionar una expresión análitica para la incertidumbre de los parámetros de tal forma que la expresión resultante siga siendo análitica. Así tenemos la distribución de Bernoulli y su conjugada Beta, la Categórica y su conjugada Dirichlet, y

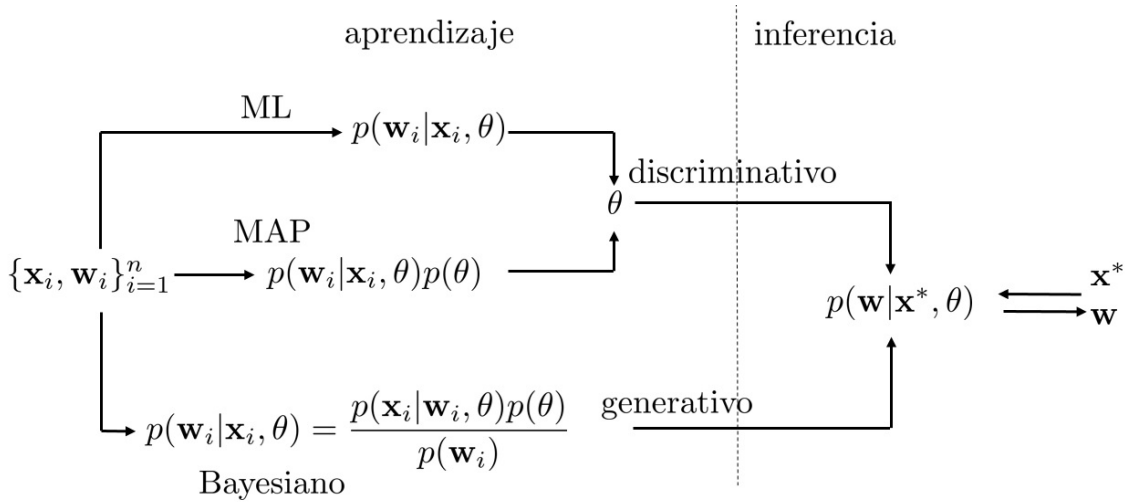


Figura 1.4: Aprendizaje Supervisado e inferencia. Dado un conjunto de datos y sus correspondientes estados del mundo, podemos seguir un enfoque discriminativo o generativo. En el enfoque discriminativo definimos la relación entre los datos del mundo y los estados $p(\mathbf{w}|\mathbf{x},\theta)$, y el problema se restringe a identificar los parámetros θ para lo cual podemos usar Máxima Verosimilitud (ML) o Máxima Probabiliad a Posteriori (MAP). En el enfoque generativo definimos una relación que identifica como las observaciones se dan en virtud del estado del mundo $p(\mathbf{x}|\mathbf{w},\theta)$. En adición, incorporamos conocimiento a priori sobre el problema. En ambos casos el objetivo es identificar el estado del mundo $\mathbf{w}$ dado un nuevo punto $\mathbf{x}^*$.

la Gaussiana y su conjugada la Normal Inversa Wishart. En todo caso, hay que seleccionar apropiadamente los parámetros de las pdf's de tal forma que describan adecuadamente los datos.

Así pues, una vez seleccionado el modelo $p(\mathbf{x}_{1,\ldots,n}|\theta)$ que describe al conjunto de datos $\{\mathbf{x}_i\}_{i=1}^n$, el problema es ahora como seleccionar el mejor valor de los parámetros θ. Para ello, hay diferentes métodos, de los cuales valos a estudiar el de máxima verosimilitud (ML), máxima probabilidad posterior (MAP), y el enfoque Bayesiano (ver Figura 1.4). En los dos primeros casos, obtenemos estimadores puntuales, mientras que en el último caso obtenemos una pdf. En todo caso, una vez aprendidos los parámetros θ, es posible hacer predicciones sobre una nueva muestra $\mathbf{x}^*$

ML. En ML buscamos obtener un estimado de los parámetros $\hat{\theta}$ que mejor interpretan los puntos $\mathbf{x}_i$ mediante la evaluación de la siguiente expresión.

$$\hat{\theta} = \arg\max_{\theta} \left(p(\mathbf{x}_{1,\ldots,n}|\theta)\right). \tag{1.8}$$

Si podemos asumir que los datos son obtenidos de forma independiente, la expresión anterior puede tomar la forma

$$\hat{\theta} = \arg\max_{\theta} \left(\prod_{i=1}^{n} p(\mathbf{x}_i|\theta)\right). \tag{1.9}$$

El procedimiento para solucionar esta expresión regularmente entraña obtener el logaritmo, lo cual convierte la multiplicatoria en sumatoria y puede también simplificar los términos

internos. Esto es posible por la observación de que el logaritmo es una función monotónicamente creciente que no cambia la posición de los puntos extremos. Enseguida, se sigue el procedimiento estándar de optimización en donde primero se la expresión se deriva con respecto a los parámetros, la expresión resultante se iguala a cero, y la expresión se resuelve para los parámetros θ. Para una nueva observación $\mathbf{x}^*$, la inferencia se realiza mediante la evaluación de la función $p(\mathbf{x}^*|theta)$.

MAP. En MAP formulamos la búsqueda directamente sobre $p(\theta|\mathbf{x}_{1,\ldots,n})$, una expresión que depende de los parámetros θ, de tal manera que tenemos

$$\hat{\theta} = \arg\max_{\theta} \left(p(\theta|\mathbf{x}_{1,\ldots,n}) \right). \tag{1.10}$$

La expresión anterior puede expandirse mediante la formulación de Bayes, de tal forma que resulta en

$$\hat{\theta} = \arg\max_{\theta} \left(\frac{p(\mathbf{x}_{1,\ldots,n}|\theta)p(\theta)}{p(\mathbf{x}_{1,\ldots,n})} \right). \tag{1.11}$$

Es interesante resaltar que el denominador no depende de θ, por lo que no afecta la posición del máximo y por tanto podemos precindir de él. Por otro lado, como en el caso de ML, si podemos asumir que las observaciones son independientes podemos evaluar cada termino $\mathbf{x}_i$ por separado. Tomando ambas observaciones en consideración, la expresión resultante es

$$\hat{\theta} = \arg\max_{\theta} \left(\prod_{i=1}^{n} p(\mathbf{x}_i|\theta)p(\theta) \right). \tag{1.12}$$

Esta expresión es interesante pues ahora se puede incluir conocimiento previo para la evaluación de la expresión, lo cual resulta en una mejor estimación. De no haber conocimiento previo, esta expresión se reduce a ML. Para obtener el máximo seguimos un proceso similar a ML. Primero, regularmente, obtenemos el logaritmo de la expresión. Luego, seguimos el proceso estándar para localizar la posición de un máximo en donde derivamos con respecto a θ, igualamos a cero, y resolvemos para la variable de interés. Para una nueva observación $\mathbf{x}^*$, la inferencia se realiza mediante la evaluación de la función $p(\mathbf{x}^*|\theta)$.

Bayesiano. En los casos anteriores, el proceso resulta en estimadores puntuales. En muchos casos, es interesante mantener la información sobre la incertidumbre y mantener la etapa de seleccioón definitiva de los valores de los parámetros hasta que esto es inevitable. Con ello fortalecemos el proceso de razonamiento evitando la toma de decisiones en pasos intemedios. En la aproximación Bayesiana, el resultado es precisamente una pdf. Esto es, asumiendo que las observaciones son independientes, los parámetros que buscamos siguen la forma

$$p(\theta|\mathbf{x}_{1,\ldots,n}) = \frac{\prod_{i=1}^{n} p(\mathbf{x}_{1,\ldots,n}|\theta)p(\theta)}{p(\mathbf{x}_{1,\ldots,n})}. \tag{1.13}$$

Luego, en el proceso de inferencia de una nueva observación $\mathbf{x}^*$, evaluamos la siguiente expresión

$$p(\mathbf{x}^*|\mathbf{x}_{1,\ldots,n}) = \int p(\mathbf{x}^*|\theta)p(\theta|\mathbf{x}_{1,\ldots,n})d\theta. \tag{1.14}$$

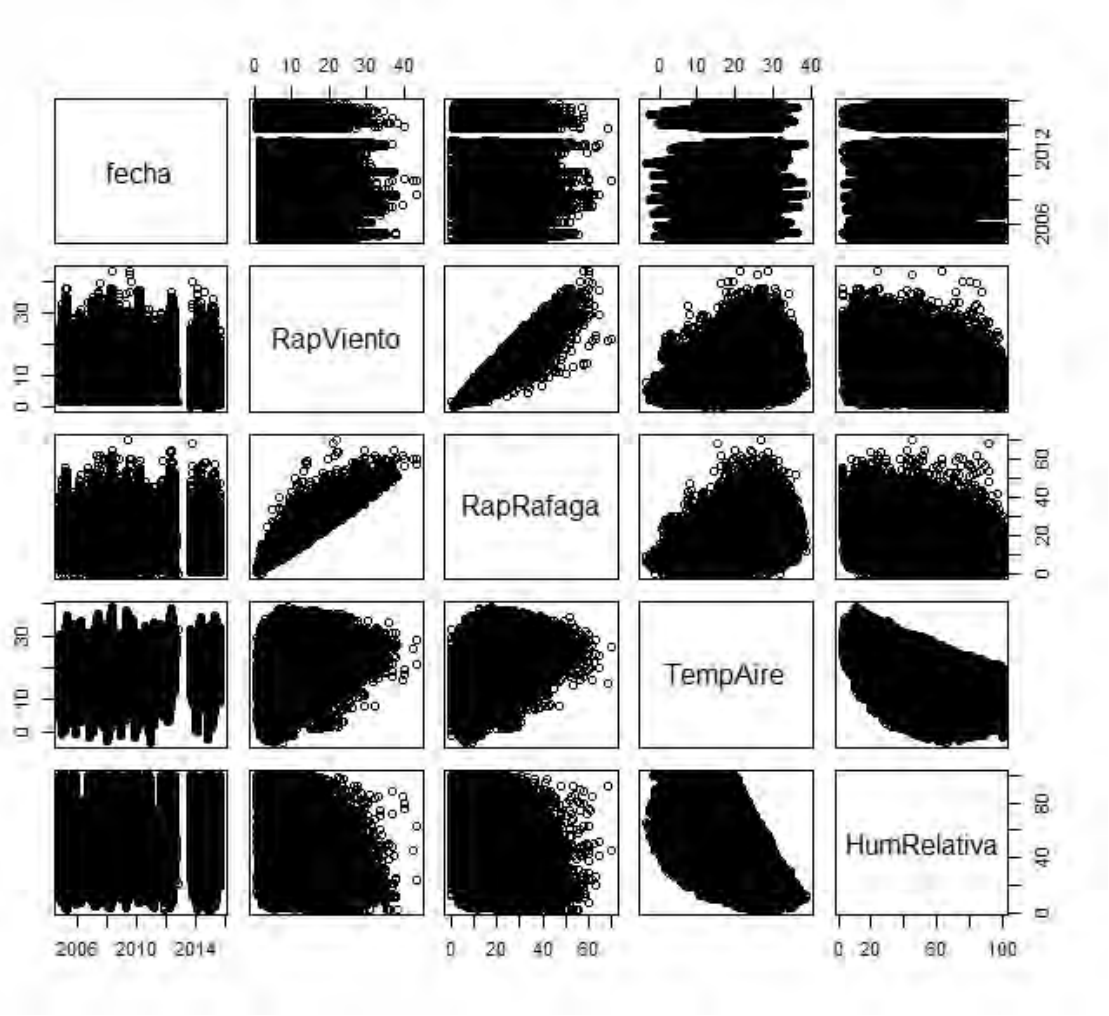


Figura 1.5: Pares de gráficas. Las gráficas muestran la relación entre pares de variables.

1.3. Ajuste de Parámetros a Datos Experimentales

El Servicio Meteorológico Nacional nos proporcionó los datos de la estación meteorológica Calvillo, localizada en Aguascalientes, México. La base de datos fue depurada por Juan Ramon Terven. El periodo de observación está ente Marzo 31 del 2006 y Septiembre del 2015, con periodos de no observación intermedios. Las observaciones incluyen: La fecha en UTC (fecha), la velocidad del viento (RapViento), la velocidad de las ráfagas de viento (RapRafaga), la temperatura del aire (TempAire), y la humedad relativa (HumRelativa). Las observaciones fueron tomadas, a excepción de los intervalos en donde no hay lecturas, cada 10 minutos. La Figura 1.5 presenta las diferentes relaciones que pueden establecerse entre pares de variables los datos. La figura es ilustrativa pero en algunas ocasiones se pierde la capacidad de apreciar la densidad de los datos en regiones. Para ello, exploramos más a detalle las diferentes relaciones.

La Figura 1.6 muestra las relaciones de las variables con respecto al tiempo. En adición, la figura también muestra la distribución de valores para la variable de interés. En adición la

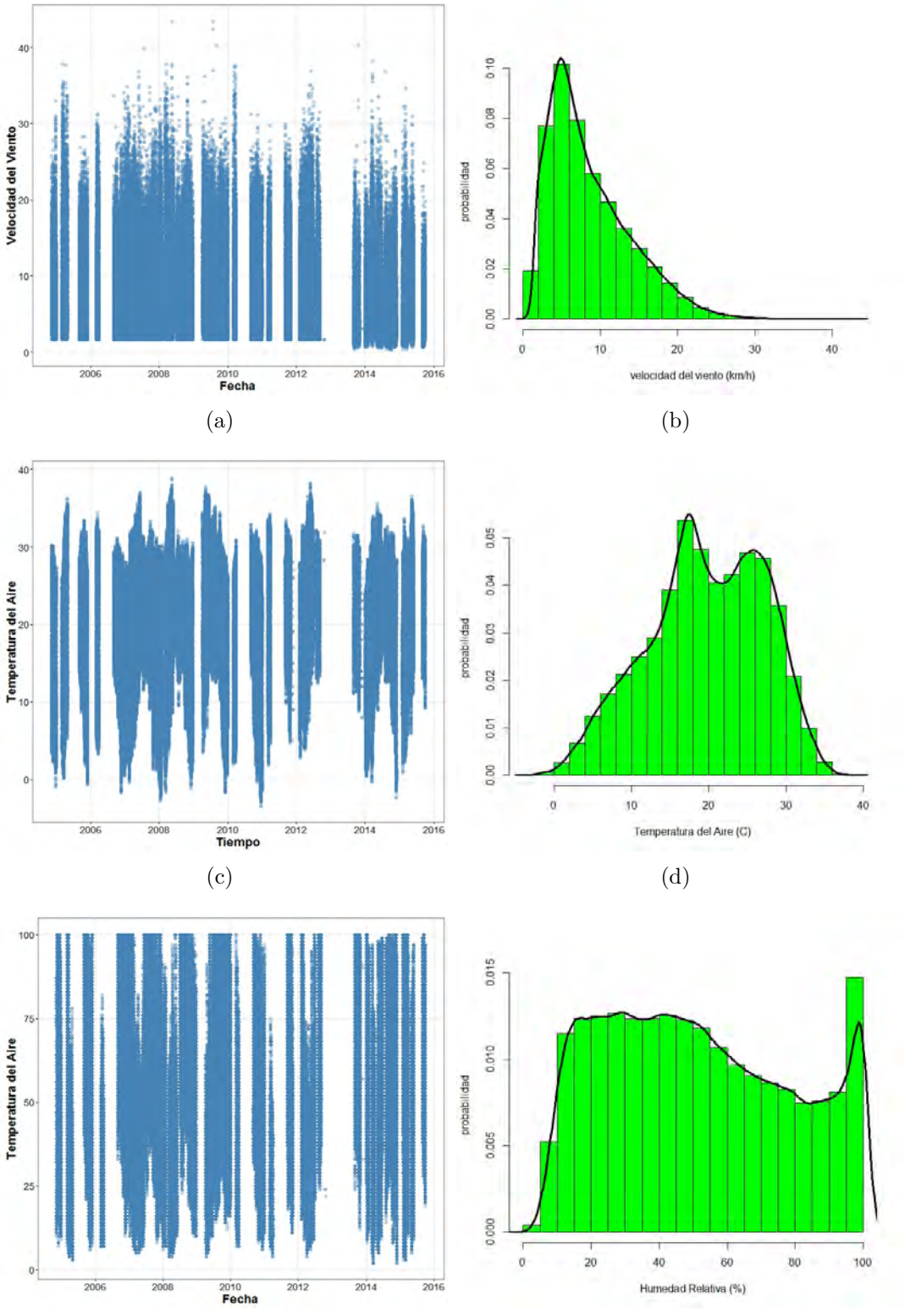
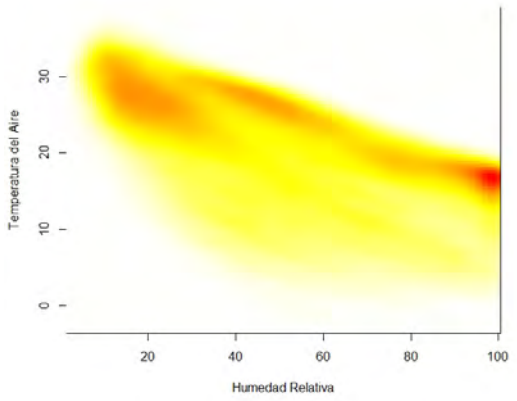
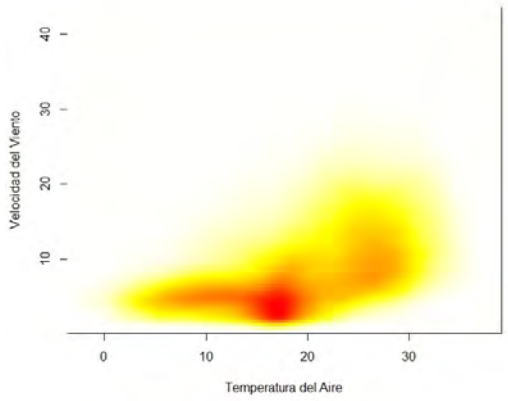


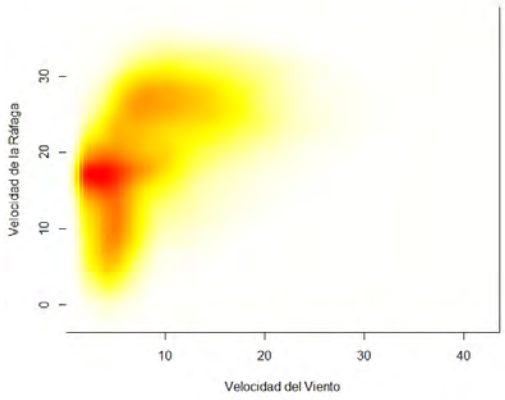
Figura 1.6: Tiempo contra velocidad del viento, temperatura del aire y humedad relativa. Las figuras son acompañadas por las frecuencias relativas, interpretadas como una probabilidad, de las variables de velocidad del viento, temperatura del aire y humedad relativa.



(a) Humedad Relativa vs Temperatura del Aire



(b) Temperatura del Aire vs Velocidad del Viento



(c) Velocidad del Viento vs Velocidad de las Rafagas

Figura 1.7: Pares de Variables. En estas gráficas se puede apreciar la relación entre pares de variables y la densidad de datos en ciertas regiones.

Figura 6.1 ilustra la relación entre pares de variables con énfasis especial en la densidad de datos en ciertas regiones. Carta *et al.*[13] hacen un recuento de las diferentes pdf utilizadas para caracterizar la velocidad del viento, siendo la más ampliamente utilizada la pdf de Weibull, dada por

$$p(x|\lambda,k) = \begin{cases} \dfrac{k}{\lambda}\left(\dfrac{x}{\lambda}\right)^{k-1} \exp\left(-x/\lambda\right)^{k} & x \geq 0, \\ 0 & x < 0 \end{cases} \tag{1.15}$$

La distribución de Weibull también es utilizada en otros casos, como por ejemplo el tiempo de falla.

ML. Para obtener los parámetros de la distribución podemos utilizar ML. Aquí seguimos parcialmente la derivación de Bhattacharya[4]. Primero, hay que notar que la expresión de verosimilitud de la muestra aleatoria formada por el conjunto $\mathbf{x} = \{x_i\}$ pueden estar dados por la expresión

$$L(\mathbf{x}) = \prod_{i=1}^{n} p\left(x_i|\lambda, k\right), \tag{1.16}$$

donde los parámetros pueden ser obtenidos resolviendo para las expresiones

$$\frac{\partial L}{\partial k} = 0 \text{ y } \frac{\partial L}{\partial \lambda} = 0. \tag{1.17}$$

La expresión de verosimilitud estaría dada por

$$L(\mathbf{x}) = \prod_{i=1}^{n} \frac{k}{\lambda}\left(\frac{x_i}{\lambda}\right)^{k-1} \exp\left(-x_i/\lambda\right)^k. \tag{1.18}$$

Aplicando logaritmos, derivando la expresión resultante con respecto a k y λ y resolviendo ambas con respecto a cero tenemos

$$\frac{\partial L}{\partial k} = \frac{n}{k} + \sum_{i=1}^{n} \log x_i - \frac{1}{\lambda}\sum_{i=1}^{n} x_i^k \log x_i = 0, \tag{1.19}$$

y

$$\frac{\partial L}{\partial \lambda} = -\frac{n}{\lambda} + \frac{1}{\lambda^2}\sum_{i=1}^{n} x_i^k = 0. \tag{1.20}$$

De (1.20) podemos deducir que el valor de λ está dado por

$$\lambda = \frac{\sum_{i=1}^{n} x_i^k}{n}. \tag{1.21}$$

Para encontrar el valor de k podemos eliminar λ de ambas ecuaciones. Ello resulta en la expresión

$$\frac{\sum_{i=1}^{n} x_i^k \log x_i}{\sum_{i=1}^{k}} x_i^k - \frac{1}{k} - \frac{1}{n}\sum_{i=1}^{n} \log x_i = 0, \tag{1.22}$$

la cual debe ser resuelta por métodos numéricos. Una forma de hacerlo es mediante el método de Newton-Raphson, el cual expresa que la raíz de una ecuación puede encontrarse iterativamente mediante la formulación

$$z_{j+1} = z_j - \frac{f(z_j)}{f'(z_j)}, \tag{1.23}$$

donde las expresiones para f y f' están dadas por

$$f(k) = \frac{\sum_{i=1}^{n} x_i^k \log x_i}{\sum_{i=1}^{k} x_i^k} - \frac{1}{k} - \frac{1}{n}\sum_{i=1}^{n} \log x_i, \tag{1.24}$$

y

$$f'(k) = \sum_{i=1}^{n} x_i^k \left(\log x_i\right)^2 - \frac{1}{k^2} \sum_{i=1}^{k} \left(k \log x_i - 1\right) - \left(\frac{1}{n} \sum_{i=1}^{n} \log x_i\right) \left(\frac{1}{n} \sum_{i=1}^{n} x_i^k \log x_i\right). \quad (1.25)$$

Para el caso particular de la muestra en la Figura 1.6(a), las variables resultan en $\lambda = 9.83$ y $k = 1.73$. El ajuste se muestra en la Figura 1.8.

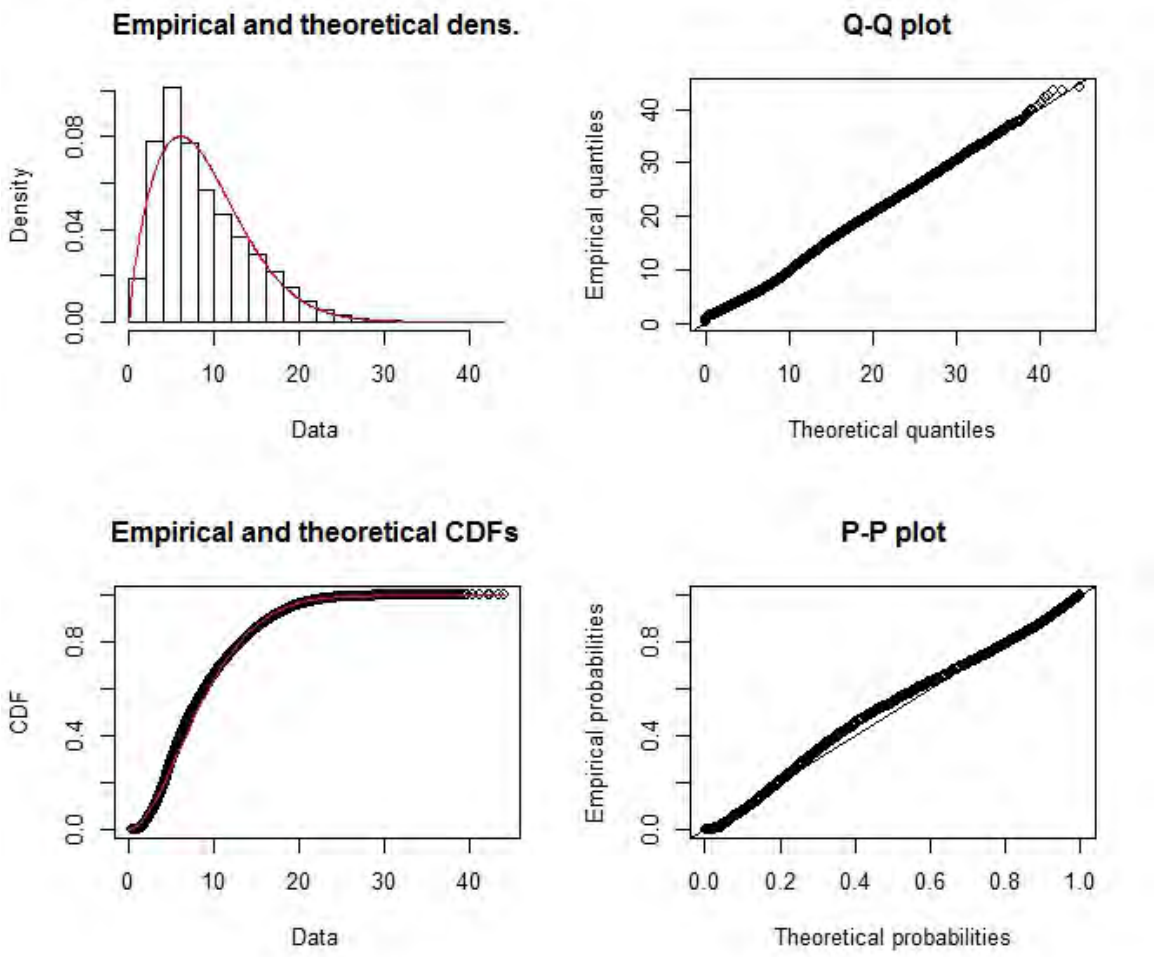


Figura 1.8: Ajuste mediante ML a los datos muestrales de la velocidad de viento en la estación meteorológica de Calvillo. El diagrama Q-Q muestra las velocidades teóricas contra las empíricas, la curva P-P muestra la evaluación de la pdf empírica, evaluada en cada punto, contra la teórica. La curva CDF muestra la distribución empírica contra la ajustada, finalmente, el ajuste se muestra contra el histograma empírico.

MAP. Otra forma de obtener los parámetros de la distribución es mediante MAP. Supongamos que aproximamos la temperatura del aire mediante una distribución normal. Asimismo,

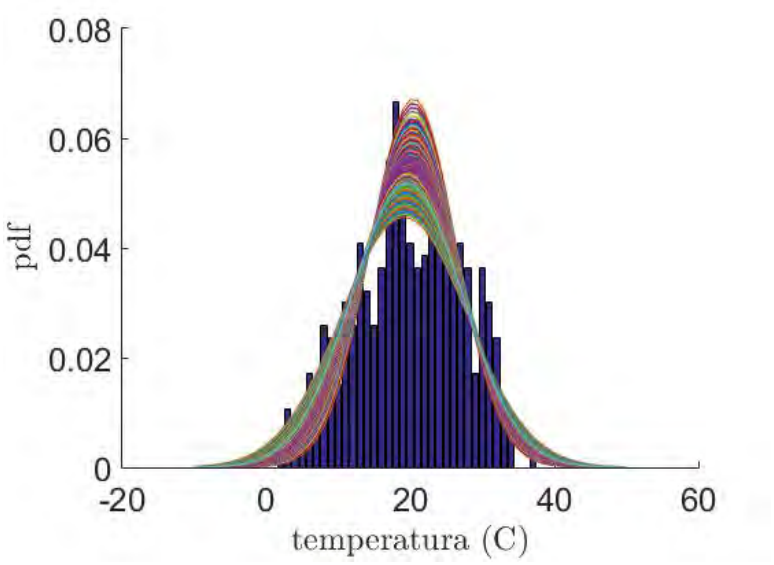


Figura 1.9: Aproximación de los datos mediante un modelo Gaussiano, usando MAP. La interpretación se ve ligeramente afectada por el valor de los hiperparámetros. En la gráfica, α, β, γ y δ fueron variados entre -10 y 10.

aproximamos el *prior* por su conjugado, la Normal Gamma Inversa. Con ello, la expresión de verosimilitud tiene la forma

$$L(\mathbf{x}) = \prod_{i=1}^{n} p(x_i|\mu, \sigma^2) p(\mu, \sigma^2), \tag{1.26}$$

donde la expresión para la verosimilitud tiene la forma

$$p(x_i|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-1/2 \left(\frac{x-\mu}{\sigma}\right)^2\right\}, \tag{1.27}$$

mientras para el prior, la expresión conjugada es

$$p(\mu, \sigma^2|\alpha, \beta, \gamma, \delta) = \frac{\sqrt{\gamma}}{\sigma\sqrt{2\pi}} \frac{\beta^\alpha}{\Gamma(\alpha)} \left(\frac{1}{\sigma^2}\right)^{\alpha+1} \exp\left(-\frac{2\beta + \gamma(\delta-\mu)^2}{2\sigma^2}\right), \tag{1.28}$$

donde la función $\Gamma(t)$ se define como

$$\Gamma(t) = \int_0^\infty x^{t-1} \exp(-x) dx. \tag{1.29}$$

Una expresión más familiar para la función Γ se da para valores de argumento n entero, en donde $\Gamma(n) = (n-1)!$. Los estimados MAP se obtienen al obtener el logaritmo de (1.26), derivando con respecto a las variables μ y σ, igualando a cero las expresiones y resolviendo igualmente para μ y σ. Las expresiones quedan en función de los valores de los hiperparámetros α, β y γ y están dados por las siguientes expresiones

$$\hat{\mu} = \frac{n\overline{x} + \gamma\delta}{n + \gamma}, \tag{1.30}$$

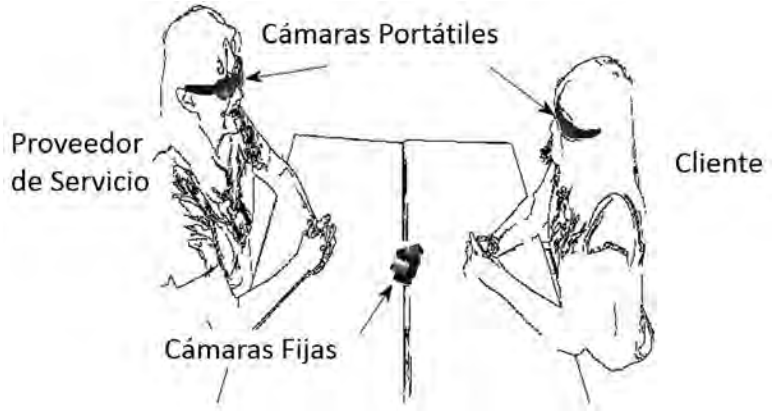


Figura 1.10: Interlocución diádica. Un sistema de Visión por Computadora es capaz de reconocer gestos realizados con la cabeza, tal como asentir, negar, mirar a la izquierda, derecha, arriba y abajo.

y

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n} (x_i - \hat{\mu})^2 + 2\beta + \gamma(\delta - \hat{\mu})^2}{n + 3 + 2\alpha}. \tag{1.31}$$

Para los datos de temperatura del aire del ejercicio, la media $\overline{x}$ es 19.86 $°C$ y desviación estándar $\overline{s}$ igual a 7.61. Con ello, podemos plantearnos diferentes modelos, los cuales tienen dependencia en los hiperparámetros $\alpha, \beta, \gamma, \delta$. La Figura 1.9 muestra la variación de la verosimilitud en función de la variación de los valores del *prior*. Los valores de los hiperparàmetros α, β, γ y δ fueron variados entre -10 y 10.

Bayesiano. A continuación desarrollo un ejemplo para la expresión de la probabilidad de los parámetros en función de los datos. Resulta que realizamos un experimento para probar la utilidad de un dispositivo para valorar la interacción entre dos interlocutores (ver Figura 1.10), un cliente y un proveedor de servicios. El dispositivo es capaz de reconocer gestos realizados con la cabeza. En particular, es capaz de reconocer asentimientos, negaciones, mirar hacia la izquierda, derecha, arriba y abajo[65]. Después de la conversación, se realiza una entrevista al cliente. Del análisis cualitativo de esta entrevista se deduce, para cada cliente, si su percepción fue que el proveedor de servicios era competente ($x_i = 1$) o no ($x_i = 0$). La Figura 1.11 muestra la distribución de Bernoulli tomando en cuenta esta medición empírica. Supongamos que queremos determinar la probabilidad λ de que el cliente haya tenido una percepción de competencia sobre el desempeño del proveedor de servicios. Para ello, utlizamos un enfoque Bayesiano cuyo formalismo puede ser expresado como

$$p(\lambda|x_{1,\ldots,n}) = \frac{p(x_{1,\ldots,n}|\lambda)p(\lambda)}{p(x_{1,\ldots,n})} \propto p(x_{1,\ldots,n}|\lambda)p(\lambda). \tag{1.32}$$

La verosimilitud nos expresa la probabilidad de que los clientes tenga una percepción sobre la competencia del proveedor de servicios. En particular para un solo cliente, la función puede tomar los siguientes valores

$$p(x_i|\lambda) = \begin{cases} \lambda & \text{si } x_i = 1, \\ 1 - \lambda & \text{si } x_i = 0. \end{cases} \tag{1.33}$$

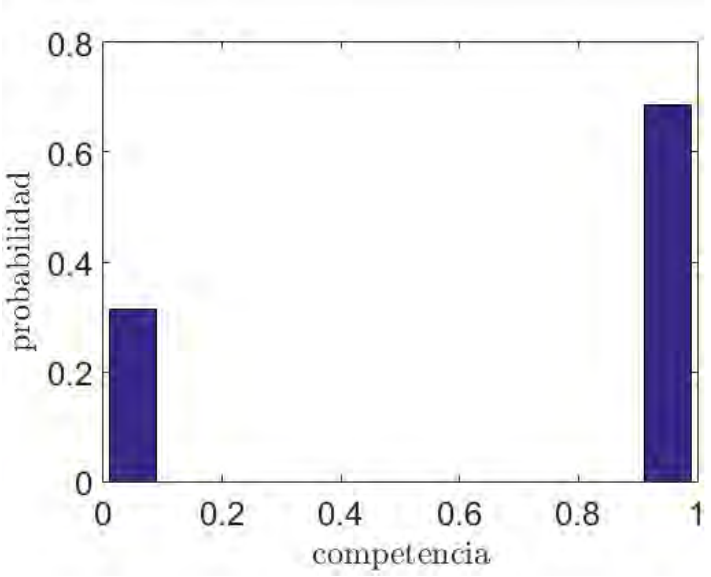


Figura 1.11: Se muestra la distribución Bernoulli para esta observación. La frecuencia de personas que encuentran competence al proveedor de servicios es 0.69.

Para un conjunto de n clientes, la función de verosimilitud toma los valores

$$p(x_{1,\ldots,n}|\lambda) \propto \lambda^X (1-\lambda)^{n-X}, \tag{1.34}$$

donde $X = \sum_{i=1}^{n} x_i$. Esta última pdf es conocida como Binomial.

Por su lado, puede ser conveniente expresar el prior de forma tal que nos facilite la solución de los productos. Para la pdf de Bernoulli o Binomial, la distribución conjunta toma la forma de una pdf Beta, la cual está dada por

$$p(\lambda) = \text{Beta}_\lambda(\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \lambda^{\alpha-1}(1-\lambda)^{\beta-1}. \tag{1.35}$$

Tal como lo discute Prince[56], la pdf Beta tiene un lóbulo principal que se mueve de izquierda a derecha conforme la fracción $\alpha/(\alpha+\beta)$. Todavía más, conforme el valor absoluto de α y β se incrementan, también se incrementa el tamaño del lóbulo.

El producto de una pdf Bernoulli y una pdf Beta puede expresarse como

$$\text{Bern}_x(\lambda)\text{Beta}_\lambda(\alpha, \beta) = \lambda^{x_i}(1-\lambda)^{1-x_i} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \lambda^{\alpha-1}(1-\lambda)^{\beta-1}. \tag{1.36}$$

Realizando los productos de los exponentes y desplazando las funciones Γ hacia la izquierda, tenemos

$$\text{Bern}_x(\lambda)\text{Beta}_\lambda(\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \lambda^{x_i+\alpha-1}(1-\lambda)^{1-x_i+\beta-1}. \tag{1.37}$$

El númerador de la expresión anterior es casi una pdf Beta, solo hay que incluirle los términos en función de α, β y x_i apropiados. El objetivo sería llegar a una expresión de la forma

$$\text{Beta}_\lambda(x_i+\alpha, 1-x_i+\beta) = \frac{\Gamma(x_i+\alpha+1-x_i+\beta)}{\Gamma(x_i+\alpha)\Gamma(1-x_i+\beta)} \lambda^{x_i+\alpha-1}(1-\lambda)^{1-x_i+\beta-1}. \tag{1.38}$$

Con lo cual, (1.37) podría expresarse como

$$\text{Bern}_x(\lambda)\text{Beta}_\lambda(\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\frac{\Gamma(x_i+\alpha)\Gamma(1-x_i+\beta)}{\Gamma(x_i+\alpha+1-x_i+\beta)}\text{Beta}_\lambda(x_i+\alpha,1-x_i+\beta), \quad (1.39)$$

o alternativamente, cercano a como lo expresa Prince[56],

$$\text{Bern}_x(\lambda)\text{Beta}_\lambda(\alpha,\beta) = \kappa(x_i+\alpha,1-x_i+\beta)\text{Beta}_\lambda(x_i+\alpha,1-x_i+\beta), \quad (1.40)$$

donde κ está dada por la siguiente expresión

$$\kappa(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}. \quad (1.41)$$

En (1.32), la expresión del Bayesiano, podemos sustituir las definiciones de la verosimilitud (1.34) y el prior (1.35) quedando como

$$\begin{aligned} p(\lambda|x_{1,\dots,n}) &\propto \lambda^X(1-\lambda)^{n-X}\lambda^{\alpha-1}(1-\lambda)^{\beta-1} \\ &\propto \lambda^{\alpha+X-1}(1-\lambda)^{n+\beta-X-1}. \end{aligned} \quad (1.42)$$

Haciendo $\tilde{\alpha} = \alpha + X$ y $\tilde{\beta} = n + \beta - X$, tenemos que el posterior puede definirse como

$$p(\lambda|x_{1,\dots,n}) = \kappa(\tilde{\alpha},\tilde{\beta})\lambda^{\tilde{\alpha}-1}(1-\lambda)^{\tilde{\beta}-1}. \quad (1.43)$$

En inferencia, nos interesa saber cual sería la probabilidad de observar un cierto valor para x^*. Para ello empleamos el siguiente formalismo

$$p(x^*|x_{1,\dots,n}) = \int_0^1 p(x^*|\lambda)p(\lambda|x_{1,\dots,n})d\lambda. \quad (1.44)$$

Utilizando los resultados anteriores tenemos la siguiente expresión

$$p(x^*|x_{1,\dots,n}) = \int_0^1 \text{Bern}_{x^*}(\lambda)\text{Beta}_\lambda(\tilde{\alpha},\tilde{\beta})d\lambda. \quad (1.45)$$

De las operaciones anteriores podemos verificar que el resultado se simplifica en virtud de que estamos utilizando pdf conjugadas. Con ello, el resultado es

$$p(x^*|x_{1,\dots,n}) = \int_0^1 \kappa(X,\hat{\alpha},\hat{\beta})\text{Beta}_\lambda(x^*+\tilde{\alpha},1-x^*+\tilde{\beta})d\lambda, \quad (1.46)$$

donde $\hat{\alpha} = x^* + \tilde{\alpha}$ y $\hat{\beta} = 1 - x^* + \tilde{\beta}$. El término κ no depende de λ y por tanto puede ser sacado de la integral, resultando en

$$p(x^*|x_{1,\dots,n}) = \kappa(x^*,\hat{\alpha},\hat{\beta})\int_0^1 \text{Beta}_\lambda(x^*+\tilde{\alpha},1-x^*+\tilde{\beta})d\lambda. \quad (1.47)$$

Por su lado, la función Beta es una pdf y por tanto su integral es uno. Por tanto la probabilidad de $p(x^*|x_{1,\dots,n})$ está dada por

$$p(x^*|x_{1,\dots,n}) = \kappa(x^*,\hat{\alpha},\hat{\beta}). \quad (1.48)$$

La Figura 1.12 muestra las curvas de λ para diferentes valores de α y β, y para diferentes tamaños de muestra.

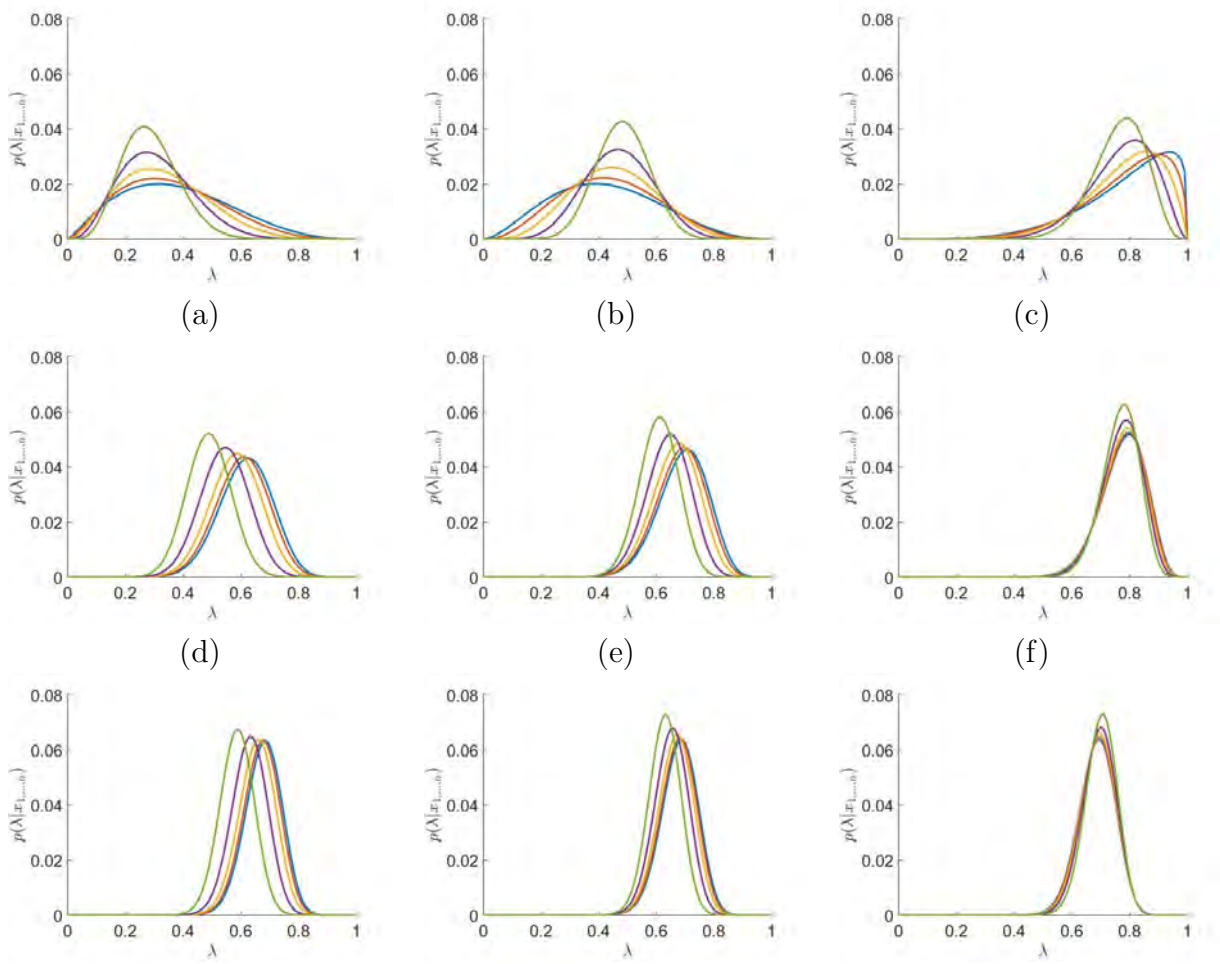


Figura 1.12: Funciones pdf para el proceso binomial-beta relacionado con la interacción diádica. De acuerdo al cociente $\alpha/(\alpha+\beta)$, en la primera columna el cociente es 0.25, en la segunda columna) el cociente es 0.5 y en tercera columna 0.75. Las diferentes curvas magnitudes crecientes de ambas α y β entre 0.25 y 0.75 creciendo al doble en cada curva. En las diferentes hileras se tienen el 10 % de las muestras para entrenamiento, en la segunda hilera el 50 % y en la tercera hilera el total de las 54 muestras.

CAPÍTULO 2

Modelos Multimodales

En este capítulo estudiamos el modelado de pdf complejas, que tienen más de una moda y que son capaces de interpretar modelos más elaborados. Un caso típico se presenta en la Figura 2.1. El ejemplo clásico en esta temática es considerar que esos modelos complejos resultan de la suma de otros más sencillos, donde la proporción de cada modelo es una variable que hay que determinar. La selección del componente que contribuye a nuestra observación es desconocida para nosotros, lo que da pie al concepto de *variable oculta*. Al mismo tiempo, esta variable oculta genera un proceso aleatorio mediante el cual determinamos la distribución básica. Por tanto, a lo más, aspiramos a tener un estimado del máximo. De ahí el nombre de la herramienta principal para la aproximación de modelos complejos: *Estimación-Maximización* (EM).

2.1. Variables Ocultas

Tal como se ilustra en la Figura 2.1, la expresión de una pdf $p(x)$ compleja puede en ocasiones visualizarse como la marginal en un espacio de probabilidad conjunto $p(x, h)$, tal como

$$p(x|\theta) = \int p(x, h|\theta)dh, \tag{2.1}$$

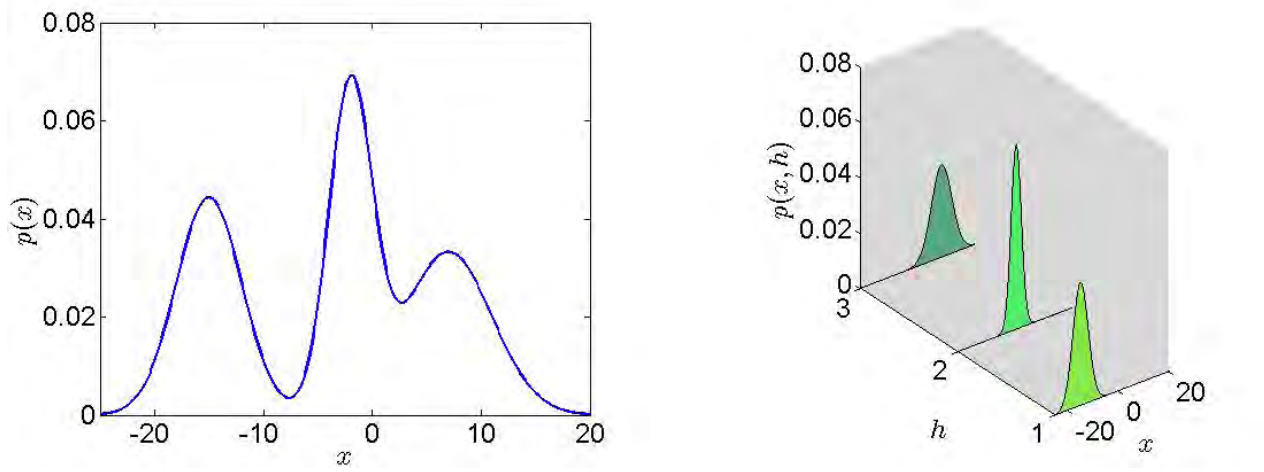
donde h es una variable que permanece oculta para nosotros y θ son los parámetros que debemos inferir. Tal como en ML, la idea es buscar los parámetros que permiten que el modelo interprete de la mejor manera los datos que han sido observados. En ese sentido, nos interesa evaluar una expresión de verosimilitud que tiene la forma

$$L(\theta) = \prod_{i=1}^{n} p(x|\theta) = \prod_{i=1}^{n} \int p(x, h|\theta)dh. \tag{2.2}$$

Tal como en ML, una estrategia para determinar la posición que maximiza la expresión de verosimilitud consiste en aplicar logaritmos en ambos lados de la ecuación. Esto resulta en

$$\ln L(\theta) = \sum_{i=1}^{n} \ln \int p(x, h|\theta)dh. \tag{2.3}$$

El problema ahora es que la aplicación de los logaritmos no nos ayuda más a simplificar la expresión anterior pues por la la presencia de la integral no puede aplicarse a los términos



(a) Una distribución compleja (b) Descomposición en distribuciones básicas

Figura 2.1: En ocasiones, una pdf compleja puede ser descompuesta en distribuciones más sencillas.

más internos. Parte del problema consiste en que la variable h es desconocida para nosotros. Sin embargo conocemos que es una variable aleatoria. Aprovechándonos de esto, buscamos estimar su valor esperado.

2.2. Maximización de la Expectativa (EM)

El método EM consiste en determinar un borde inferior $\mathcal{B}(\theta)$ a la superficie definida por la verosimilitud, seguido de un proceso de búsqueda del máximo sobre ese borde inferior. En la práctica, ese borde inferior también depende de un conjunto de pdfs sobre las variables ocultas $\{q_i(h_i)\}_{i=1}^{I}$, *i.e.*, se tiene una superficie [56] $\mathcal{B}(\{q(h_i)\}, \theta)$. Diferentes funciones $q_i(h_i)$ predicen diferentes bordes inferiores. Para explorar estos límites primero comenzamos explorando algunas propiedades de desigualdades y luego vemos como se obtienen los términos óptimos de esas desigualdades para nuestro problema.

Por ejemplo, la desigualdad de Jensen[48] nos dice que

$$\int \ln y \cdot p(y)dy \leq \ln \int yp(y)dy, \tag{2.4}$$

o que el valor esperado del logaritmo de una variable aleatoria es siempre menor o igual al logaritmo del valor esperado de la variable aleatoria. Es decir,

$$\mathbb{E}(\ln y) \leq \ln \mathbb{E}(y). \tag{2.5}$$

Esta propiedad nos permite obtener un límite inferior a la probabilidad conjunta $p(x, h|\theta)$ al

aplicar la desigualdad de Jensen sobre (2.3), tal que

$$\begin{aligned} \ln\left(\int p(x,h|\theta)dh\right) &= \ln\left(\int q(h)\frac{p(x,h|\theta)}{q(h)}dh\right), \\ &\geq \int q(h)\ln\frac{p(x,h|\theta)}{q(h)}dh. \end{aligned} \tag{2.6}$$

Es decir que se tiene un límite inferior en el proceso de maximización de la verosimilitud.

Por otro lado, la naturaleza de las funciones $q_i(h_i)$ puede ser explorado de la siguiente manera. Partiendo de (2.3) y aprovechando (2.6), tenemos que el límite inferior puede ser definido como

$$\mathcal{B}(\{q_i(h_i)\},\theta) = \sum_{i=1}^{n}\int q_i(h_i)\ln\frac{p(x_i,h_i|\theta)}{q_i(h_i)}dh_i. \tag{2.7}$$

Expandiendo el término de la pdf conjunta se tiene

$$\mathcal{B}(\{q_i(h_i)\},\theta) = \sum_{i=1}^{n}\int q_i(h_i)\ln\frac{p(h_i|x_i,\theta)p(x_i|\theta)}{q_i(h_i)}dh_i. \tag{2.8}$$

Luego, aplicando propiedades de logaritmos se llega a la expresión

$$\mathcal{B}(\{q_i(h_i)\},\theta) = \sum_{i=1}^{n}\int q_i(h_i)\ln p(x_i|\theta)dh_i + \sum_{i=1}^{n}\int q_i(h_i)\ln\frac{p(h_i|x_i,\theta)}{q_i(h_i)}dh_i. \tag{2.9}$$

El primer término puede integrarse con respecto a h_i, resultando en

$$\mathcal{B}(\{q_i(h_i)\},\theta) = \sum_{i=1}^{n}\ln p(x_i|\theta) + \sum_{i=1}^{n}\int q_i(h_i)\ln\frac{p(h_i|x_i,\theta)}{q_i(h_i)}dh_i. \tag{2.10}$$

Nuevamente, el primer término no requiere de las funciones q_i que definen el límite inferior. Por tanto puede ser ignorado en la búsqueda de la posición del máximo del límite inferior. Así que lo conducente es buscar la optimización del término

$$q_i(h_i) = \arg\max\left\{\int q_i(h_i)\ln\frac{p(h_i|x_i,\theta)}{q_i(h_i)}dh_i\right\}. \tag{2.11}$$

Ahora aplicamos la desigualdad $\ln y \leq y-1$ (ver Figura 2.2) para realizar la siguiente derivación. Sea

$$\begin{aligned} \int q_i(h_i)\ln\frac{p(h_i|x_i,\theta)}{q_i(h_i)}dh_i &\leq \int q_i(h_i)\left(\frac{p(h_i|x_i,\theta)}{q_i(h_i)}-1\right)dh_i, \\ &\leq \int \left(p(h_i|x_i,\theta)-q_i(h_i)\right)dh_i, \\ &\leq 0. \end{aligned} \tag{2.12}$$

Es decir, que ya cuando se incluye el signo la función de costo es positiva. Como lo que se busca es minimizar, el costo menor que se puede obtener en cero. Es decir que

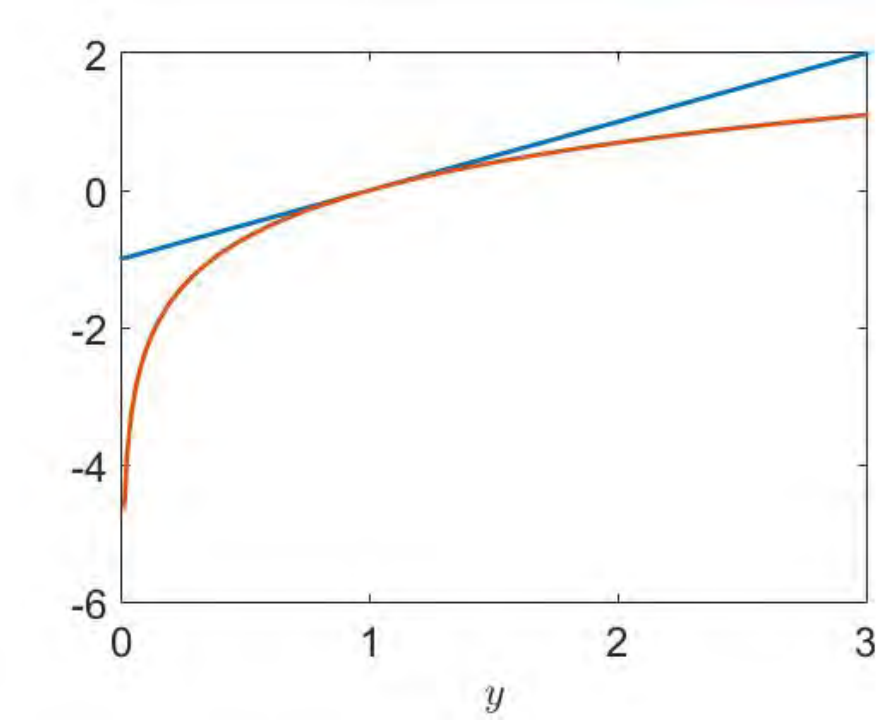


Figura 2.2: Ilustración de la desigualdad $\ln y \leq y - 1$, $y - 1$ se ilustra en azul y $\ln y$ en naranja.

$$\begin{aligned} \int q_i(h_i) \ln \frac{p(h_i|x_i,\theta)}{q_i(h_i)} dh_i &= \int p(h_i|x_i,\theta) \ln \left(\frac{p(h_i|x_i,\theta)}{p(h_i|x_i,\theta)} \right) dh_i \\ &= 0. \end{aligned} \tag{2.13}$$

En otras palabras, el valor óptimo para $q_i(h_i)$ es $p(h_i|x_i,\theta)$.

En resumen, el procedimiento de EM se basa en dos pasos. El primero consiste en la *Estimación* mediante el establecimiento de un límite inferior dado por el cálculo de la función

$$q_i(h_i) = p(h_i|x_i,\theta) = \frac{p(x_i|h_i,\theta)p(h_i|\theta)}{p(x_i)}. \tag{2.14}$$

Enseguida se realiza la *Maximización* obteniendo el valor máximo en la superficie estimada mediante el cálculo de los parámetros θ. Para ello, (2.7) puede ser utilizada, con la simplificación de que los términos que no dependen de θ pueden ser eliminados del proceso de búsqueda del máximo, resultando en

$$\theta = \arg \max_{\theta} \sum_{i=1}^{n} \int q_i(h_i) \ln p(x_i, h_i|\theta) dh_i. \tag{2.15}$$

2.3. Mezcla de Gaussianas

El esquema descrito anteriormente es de aplicación general. Como un ejemplo, en esta sección vamos a realizar la estimación de una distribución que tiene el efecto combinado de

varias distribuciones Gaussianas. La distribución que estamos buscando tiene la pdf marginal definida como

$$p(x|\theta) = \sum_{k=1}^{K} \lambda_k \text{Norm}_x(\mu_k, \Sigma_k). \tag{2.16}$$

En la estrategia a seguir se define una variable aleatoria oculta h a través de la cual escogemos una pdf Gaussiana en particular. Esto es, identificamos una pdf Gaussiana como

$$p(x|h, \theta) = \text{Norm}_x(\mu_h, \Sigma_h), \tag{2.17}$$

que es seleccionada en función de una variable aleatoria oculta que sigue su propia pdf, tal como

$$p(h|\theta) = \text{Cat}_h(\lambda), \tag{2.18}$$

donde Cat_h define una pdf discreta sobre la variable h. Así, (2.16) se expresa como

$$p(x|\theta) = \sum_{k=1}^{K} p(x, h = k|\theta), \tag{2.19}$$

o alternativamente

$$p(x|\theta) = \sum_{k=1}^{K} p(x|h = k, \theta) p(h = k|\theta). \tag{2.20}$$

Para el paso de *Estimación* resolvemos (2.14). Para el caso particular de la mezcla de Gaussianas, la expresión resultante es

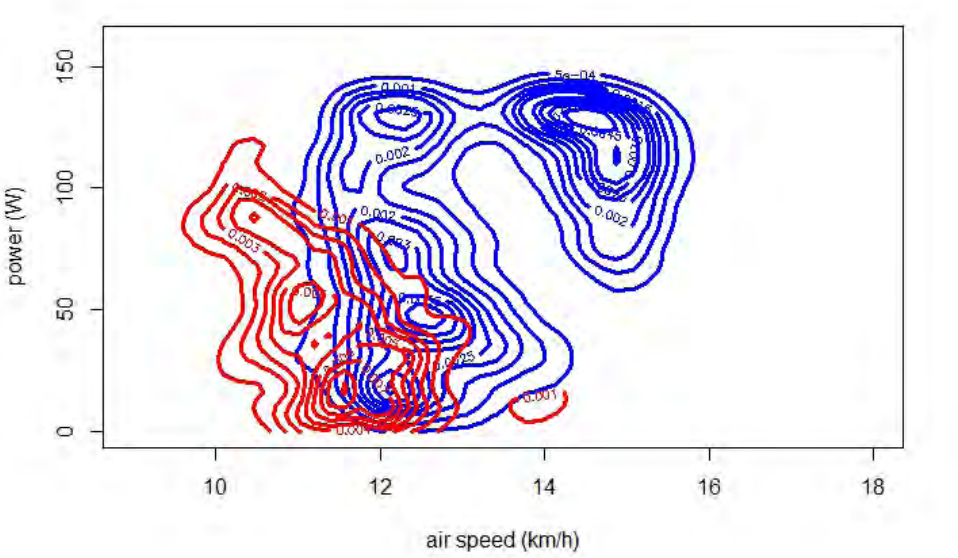
$$\begin{aligned} q_i(h_i) = p(h_i|x_i, \theta) &= \frac{p(x_i|h_i, \theta) p(h_i|\theta)}{p(x_i)}, \\ &= \frac{\lambda_k \text{Norm}_x(\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \lambda_j \text{Norm}_x(\mu_j, \Sigma_j)}, \\ &= r_{ik}. \end{aligned} \tag{2.21}$$

En el paso de maximización partimos de (2.15), expresada para el caso que nos ocupa

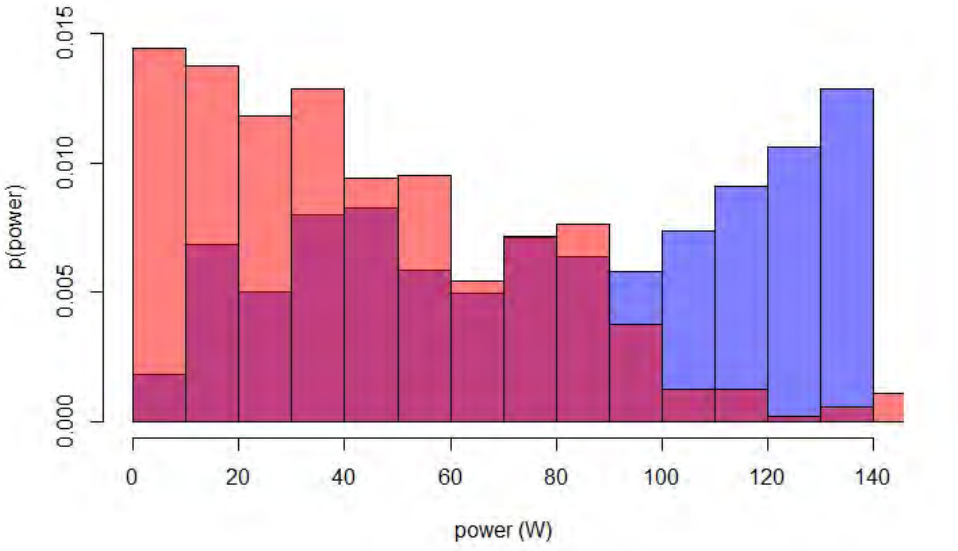
$$\begin{aligned} \theta &= \arg\max_{\theta} \sum_{i=1}^{n} \sum_{k=1}^{K} q_i(h_i = k) \ln p(x_i, h_i = k|\theta), \\ &= \arg\max_{\theta} \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \ln \left(\lambda_k \text{Norm}_x(\mu_k, \Sigma_k)\right), \end{aligned} \tag{2.22}$$

Con ello, la expresión de verosimilitud que se busca maximizar está dada por

$$L(\theta) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \ln \left(\lambda_k \text{Norm}_x(\mu_k, \Sigma_k)\right) + \gamma \left(\sum_{k=1}^{K} \lambda_k - 1 \right), \tag{2.23}$$

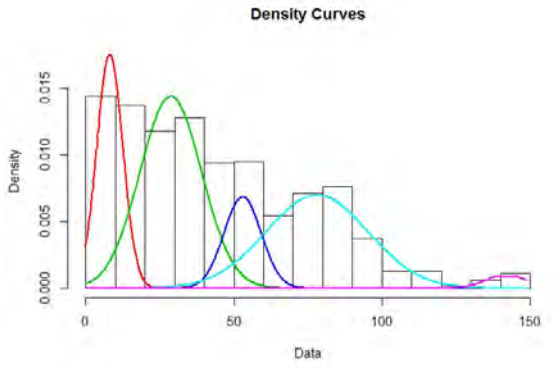


(a) Potencia contra velocidad

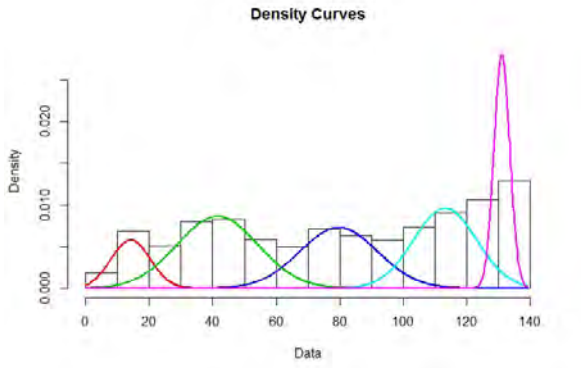


(b) Marginales con respecto a la potencia

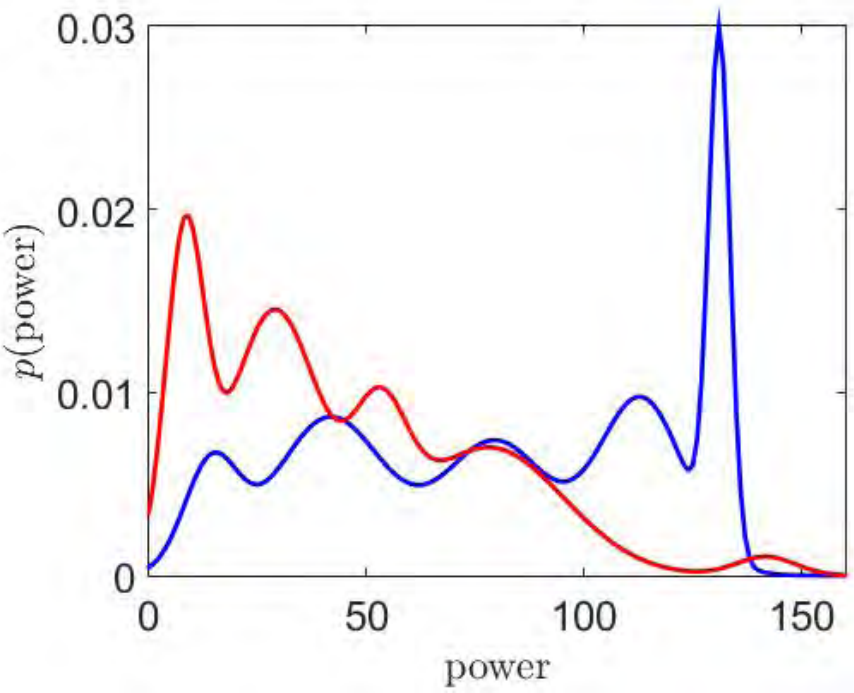
Figura 2.3: Datos de potencia contra velocidad recabados durante pruebas realizadas con alas de dos tipos de envergadura. En (a), las líneas rojas corresponden a curvas de nivel para alas largas y los contornos azules a alas cortas. En (b) se muestra la marginal de los datos con respecto a la potencia.



(a) Gaussinas alas largas



(b) Gaussianas alas Largas



(c) MOGs

Figura 2.4: Ilustración de las Gaussianas estimadas por clase y pdf resultante.

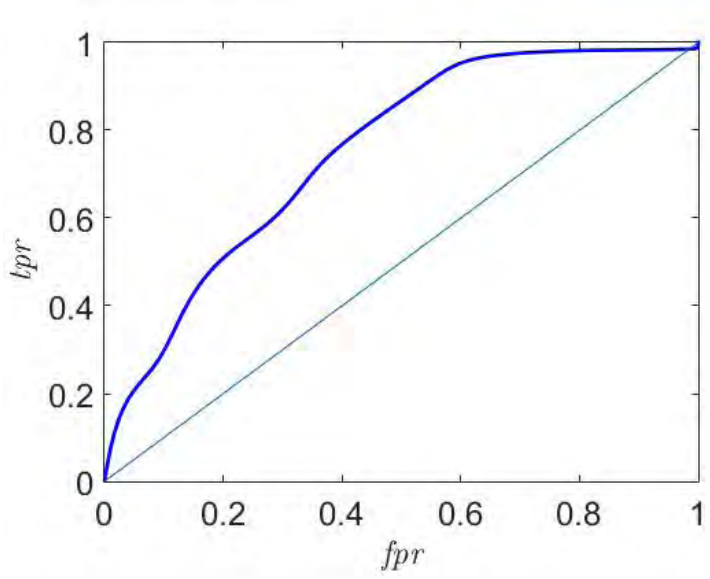


Figura 2.5: Análisis ROC. Esta es la curva resultante para diferentes criterios τ de decisión para la distinción entre alas cortas y largas basadas en la potencia aplicada. El área bajo la curva resultante es 0.75.

donde el segundo término contiene el operador de Lagrange y expresa la condición de que la suma de las componentes básicas debe ser igual a uno. Como es habitual, los parámetros que maximizan (2.23) se encuentran derivando $L(\theta)$ con respecto a θ e igualando a cero. En este caso resulta en

$$\partial L(\theta)/\partial\theta = \sum_{i=1}^{n} r_{ik}\frac{1}{\lambda_k} + \gamma = 0. \tag{2.24}$$

De esta forma se encuentra el valor del multiplicador de Lagrange como

$$\gamma = -\frac{\sum_{i=1}^{n} r_{ik}}{\lambda_k}, \tag{2.25}$$

y los valores de λ_k, μ_k y Σ_k están dados como [56]:

$$\lambda_k = \frac{\sum_{i=1}^{n} r_{ik}}{\sum_{k=1}^{K}\sum_{i=1}^{n} r_{ik}},\ \mu_k = \frac{\sum_{i=1}^{n} r_{ik}\mathbf{x}_i}{\sum_{i=1}^{n} r_{ik}},\ \Sigma_k = \frac{\sum_{i=1}^{n} r_{ik}(\mathbf{x}_i-\mu_k)(\mathbf{x}_i-\mu_k)^T}{\sum_{i=1}^{n} r_{ik}}. \tag{2.26}$$

2.4. Aplicación: Alas Deformables

Supongamos que se busca reducir el consumo energético de un UAV pequeño [1]. Para ello se ha diseñado la capacidad de modificar dinámicamente la envergadura de sus alas de

[1]Esta aplicación fue desarrollada por Othón González en su tesis de maestría[23].

acuerdo a la etapa de la misión que le toca realizar. El objetivo es corroborar si efectivamente es posible distinguir entre los tipos de ala `Corta` y `Larga` basados en la potencia aplicada al UAV. Para ello, se tienen las mediciones de velocidad y potencia para ambos regímenes ilustrados en la Figura 2.3. En la Figura 2.3(a) se ilustran curvas de nivel que expresan la densidad de observaciones. Los contornos rojos corresponden a la clase ala `Larga` y los contornos azules a la clase ala `Corta`. Por su lado, en la Figura 2.3(b) se muestra la marginal de los datos con respecto a la potencia. Una caracterización de las clases se puede dar mediante la determinación del grado de separabilidad que existen entre las clases ala `Corta` y `Larga`. Utilizando la implementación del algoritmo de EM en **MixTools** de **R**. Estimamos las mezclas de Gaussianas para los datos de ambas clases. Una visualización de las Gaussianas estimadas se encuentra en la Figura 2.4.

Enseguida, utilizando las pdf de las MOGs podemos evaluar el desempeño de un clasificador para diferentes valores de umbral τ. Para ello, evaluamos el área bajo la curva que resulta en verdaderos positivos, falsos positivos, falsos negativos y verdaderos negativos al establecer diferentes criterios de decisión. Utilizando esos valores se obtienen las tasas de falsos positivos y verdaderos positivos requeridos para realizar análisis ROC[62]. Lo anterior resulta en la curva mostrada en la Figura 2.5. El área bajo la curva es 0.75.

CAPÍTULO 3

Características

Uno de los problemas centrales del reconocimiento de patrones es la caracterización de los problemas. Las características de un problema permiten su expresión en términos que puedan ser calculador por una computadora. Aquí estudiamos aspectos relacionados con la selección de las características más importantes y su ordenamiento.

3.1. Selección de Características

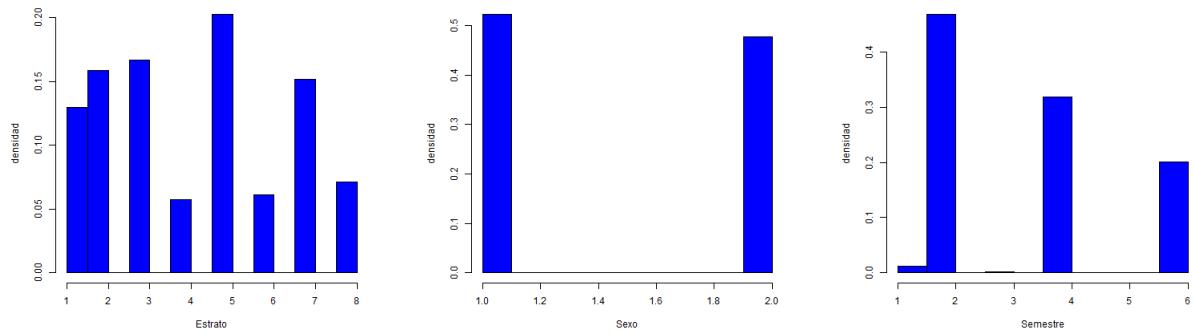
La selección de características es el problema de detectar los predictores que mejor nos ayudan a inferir el estado del mundo. El problema es importante por dos razones importantes. Por un lado, actualmente tenemos dispositivos que nos ayudan a capturar mucha información. Sin embargo, no necesariamente toda la información es relevante. Incorporla en nuestro proceso de inferencia puede resultar computacionalmente muy caro. Por otro lado, si las características no aportan al proceso de inferencia, en la práctica estarían introduciendo ruido que puede ser detrimental para el proceso de inferencia. Es decir, que tanto por la eficiencia como por la eficacia es conveniente realizar la selección de características.

El problema de la selección de las mejores características puede sin embargo ser complicado. Imáginese una base de datos $\mathbf{D}_{n\times m}$ con n observaciones de m características. En este conjunto hay un número de subconjuntos igual a

$$N = \sum_{k=1}^{m} \left(\begin{array}{c} m \\ k \end{array} \right), \tag{3.1}$$

que puede ser prohibitivo evaluar aun para valores de m pequeños. Por ejemplo, en un conjunto de $m = 20$ características hay 1,048,575 subconjuntos de al menos un elemento; mientras que para $m = 30$, hay 1,073,741,823 de subconjuntos. Ante este panorama, se ha buscado desarrollar heurísticas, las cuales se pueden agrupar en la selección de subconjuntos de predictores, regularización y la reducción de dimensiones. En este capítulo estudiamos algunas de estas.

Sopóngase que se tiene una base de datos correspondiente a una encuesta realizada sobre las adicciones en jóvenes. La base de datos contiene 1,838 entrevistas realizadas a jóvenes, donde a cada jóven se le pidió responder hasta 314 reactivos. En esta base de datos, la atención se centra en el consumo riesgoso de alcohol y los elementos del entorno que le propician. En la base de datos esta variable se deriva de AUDIT-C[11], un subconjunto de tres preguntas del cuestionario AUDIT. Por su lado el AUDIT es un cuestionario de 10



(a) Estrato (b) Edad (c) Semestre

Figura 3.1: La base de datos sobre consumo excesivo y/o abuso activo de alcohol.

preguntas, incluido en el Drinking Practice Questionnaire (DPQ)[12]. Si bien el AUDIT fue desarrollado para determinar consumo elevado y/o abuso activo de alcohol, se estimaba que dada su longitud, iba a ser muy difícil su aplicacion por cuidadores primarios. El AUDIT-C evalua la cantidad y frecuencia para decidir si un consumo dado de alcohol es abundante y/o hay abuso activo de alcohol o dependencia en él. Dependiendo del género del sujeto, una cierta medida de AUDIT-C asigna la categoría consumo riesgoso o consumo no riesgoso. Sin embargo, es conveniente notar que estudios han determinado que el uso de estas preguntas reporta un área bajo la curva de 0.891 para detectar consumo elevado de alcohol. Por su parte, el área bajo la curva para abuso de alcohol o dependencia fue de 0.786[11]. En nuestro estudio se quiere determinar si hay otras variables del contexto que determinen una situación de riesgo para el consumo de alcohol. El determinar estas variables es importante en tanto que, una vez conocidas, uno pudiera ayudar a determinar acciones de prevención. La edad de los encuestados es entre 13 y 28 años, con una media de 16.78. En la muestra hay un 53 % de mujeres y 48 % de hombres. Prácticamente todos los estudiantes asisten a semestres pares, dentro de los seis posibles, habiendo una disminución en el número de participantes conforme se avanza en los semestres (Figura 3.1).

La Figura 3.2 muestra algunos de los predictores agrupados por su relación con el consumo excesivo y/o abuso activo de alcohol. Considere por ejemplo, el mayor número de copas completas tomadas por una persona entrevistada en los últimos doce meses (Figura 3.2(c)). Ciertamente observamos que haber tomado seis copas al mes es un fuerte indicador sobre el riesgo de alcoholismo de una persona. Igualmente, un fuerte indicador negativo es cero copas de alcohol en el último mes. Sin embargo, hay una zona entre una y dos copas al mes en donde hay un gran traslape. Según la evidencia, las personas pueden haber tomado varias copas completas por ocasión durante su vida y aun así no presentar riesgo de convertirse en alcohólico (Figura 3.2(b)). Sin embargo, el riesgo aumenta significativamente cuando se tienen episodios en donde se han tomado cinco o más copas. Igualmente, se puede deducir que es prácticamente nula si solo se han tenido episodios en donde se han tomado a lo más dos copas de vino. El número de veces que una persona se ha emborrachado en los últimos doce meses es un fuerte indicador de su riesgo de ser alcoholico (Figura 3.2(x)). Esto es prácticamente cierto para cuatro o más veces, es casi seguro para tres, pero es un

poco incierto para una o dos veces. Hay casos en los que algunas personas se han no se han emborrachado en los últimos doce meses y aun así presentan riesgo. Igualmente, hay personas que se han emborrachado una, dos o hasta tres veces y no están en el grupo de personas con riesgo de alcoholismo. Esto es interesante pues destaca la complejidad de los problemas y la tendencia natural a tomar conclusiones con pocos datos. Por su parte, el número de copas completas también es un predictor importante (Figura 3.2(e)). Considere que si se tomaron tres o cuatro copas completas, es casi seguro que se tenga un riesgo hacia el alcoholismo. Por otro lado, si no se ha tomado copas completas es también poco, pero no nulo, el riesgo de alcoholismo. Es interesante también el número de copas entre uno y dos, donde las respuestas son dificiles de discernir. Tomar alcohol en casa de otro es un fuerte predictor respecto al riesgo de alcoholismo (Figura 3.2(a)). Sin embargo, el predictor tiene un grado de confusión que no es despreciable. Los datos muestran que son pocas las personas que no han probado alcohol en la vida (Figura 3.2(b)). Igualmente muestra que la probabilidad de tomar alcohol y no tener asociado un riesgo de alcoholismo es bajo y aproximadamente constante. Sin embargo, para las personas que presentan riesgo de alcoholismo, el tener un número de cinco o más veces de haber probado alcohol se traduce en un incremento significativo del riesgo. En esta visita a la distribuciones de probabilidades de los predictores dado que se conoce que la persona tiene o no el riesgo de ser alcoholica se tienen el número de veces que se ha fumado cigarro completo(Figura 3.2(f)). Aquí es notable que una buena parte de la población encuestada ha fumado cigarrillos completos. Igualmente, que el riesgo de alcoholismo se incrementa conforma el número de cigarros completos fumados lo hace. Sin embargo, en las regiones entre uno y dos cigarros completos la discriminación es bastante confusa. En la base de datos que se analiza se tiene un conjunto importante de descriptores que no predictores que contribuyen poco a la distinción de riesgo/no-riesgo en consumo de alcohol (Figura 3.2(g)-(i)). Para ejemplificar, a continuación se muestran los predictores para Te Felicitan, calificaciones, y a veces creo que no soy una buena persona. Es conveniente notar como los predictores más ricos nos dan una idea sobre como separar las clases. Sin embargo, los predictores pobres pueden llegar a ser informativos, sobre todo cuando se tratan de ideas preconcebidas. Aquí estamos dejando que los datos se manifiesten y nos proporcionen conclusiones sobre como las cosas son y no como creemos que son.

3.2. Algoritmo de Boruta

Boruta es un algoritmo para la selección de todas las características relevantes, en contraposición a algoritmos que buscan obtener un conjunto mínimo, que se se construye alrededor de la respuesta de un clasificador. Los algoritmos que aspiran a encontrar todas las características relevantes parten del principio de que si bien es cierto que una característica que degrada el nivel de desempeño de un clasificador puede considerarse importante, el hecho de que no degrade el desempeño no significa que no sea importante[49].

Boruta utiliza random forest aprovechando que la salida de ese clasificador proporciona una medida de importancia de los predictores usados en la clasificación. En random forest, cada árbol es desarrollado en base a un subconjunto de muestras diferentes del conjunto de entrenamiento[16]. La importancia asignada a una característica corresponde a la pérdida de precisión en la clasificación por la permutación de los valores de los atributos entre los

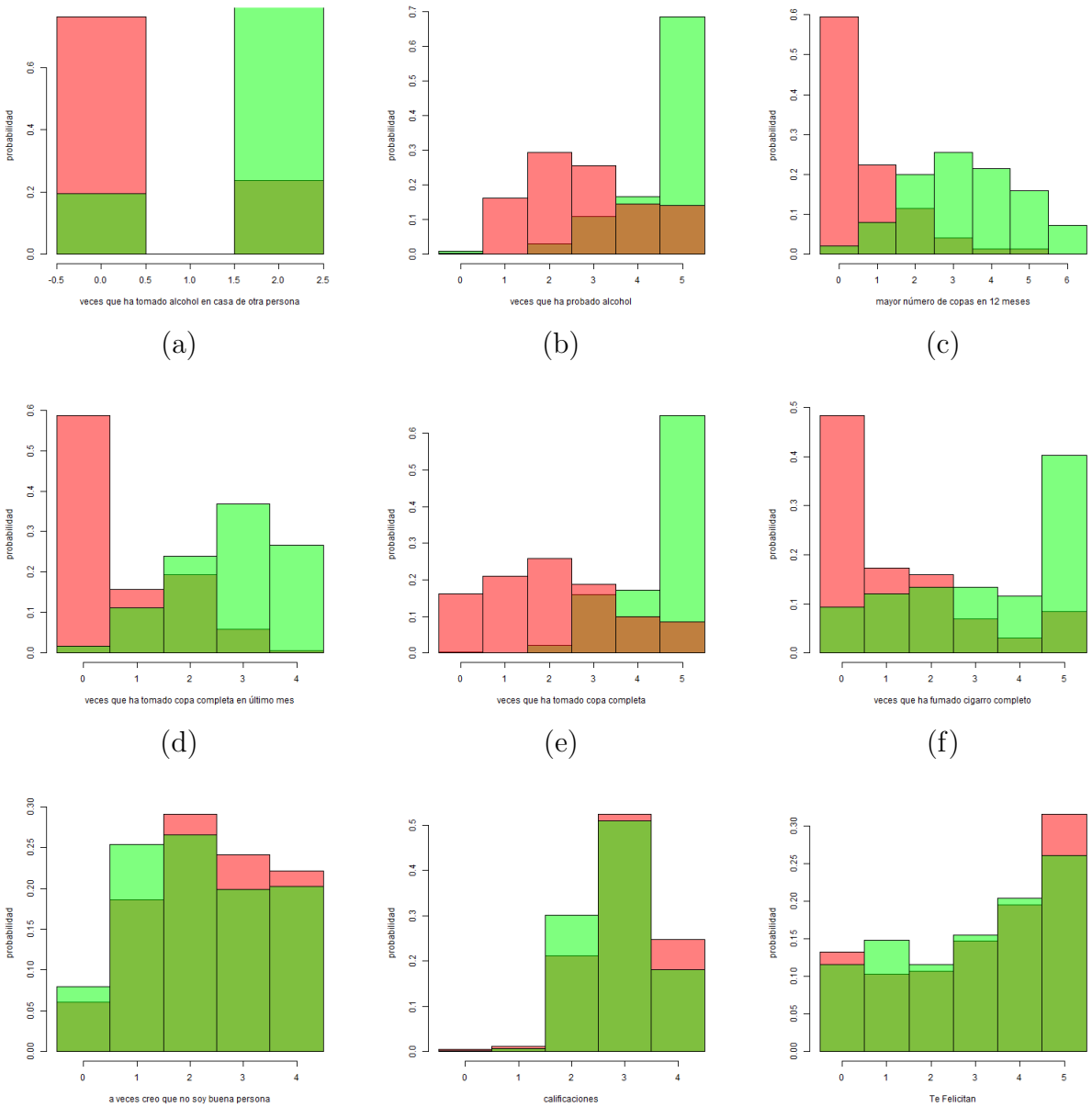


Figura 3.2: Distinción de variables por consumo elevado y/o abuso activo de alcohol.

objetos, calculada para todos los árboles del bosque en los cuales se usa un atributo dado. En el proceso, el valor medio y la desviación estándar de la pérdida es calculada. El llamado Z-score se define como el cociente entre la pérdida promedio y la deviación estándar.

El algoritmo de Boruta esencialmente determina si una importancia relativa entre los predictores y un conjunto de atributos extendidos que se distribuyen entre los datos. El algoritmo funciona como sigue (Figura 3.3). Primero, se extiende el conjunto de atributos agregando una copia de todos los predictores. Los valores de los nuevos atributos son mezclados entre los objetos con la finalidad de remover la correlación con la respuesta. Enseguida se ejecuta el algoritmo de clasificación por *random forest* en el conjunto extendido de atributos, calculandose el Z-score para los predictores. Para cada atributo de importancia que no ha sido determinada se realiza una prueba de igualdad de dos lados[5]. Cuando se tiene una importancia significativamente menor que el valor mediano del Z-score, el atributo es considerado no importante y es removido. De forma similar, los atributos que son significativamente superiores al valor mediano del Z-score son considerados importantes. Una vez hecho lo anterior, los predictores que se añadieron son removidos. Los pasos anteriores son repetidos hasta que se ha asignado la debida propiedad de importancia para todos los predictores.

Como paso extra, si se quisieran obtener predictores altamente relevantes y descorrelacionados, se puede ejecutar una prueba adicional para examinar correlación entre variables.

3.3. Asignación de Pesos

En AHP (Analytic Hierachy Process) los factores que llevan a una decisión son ordenados en una estructura jerárquica desde una meta hacia los criterios, subcriterios y alternativas. Supóngase que se tienen n alternativas, $A_1, \ldots, A_n$, con pesos $w_1, \ldots, w_n$. La matriz de cocientes de los pesos tendrá la forma[57]

$$\begin{pmatrix} w_1/w_1 & w_1/w_2 & \ldots & w_1/w_n \\ w_2/w_1 & w_2/w_2 & \ldots & w_2/w_n \\ \vdots & \vdots & & \vdots \\ w_n/w_1 & w_n/w_2 & \ldots & w_n/w_n \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = n \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}, \tag{3.2}$$

$$\mathbf{A}\mathbf{w} = n\mathbf{w}.$$

Una matriz con la estructura de $\mathbf{A}$ es llamada matriz reciproca. Notando que las hileras proporcionales unas con otras, podemos inferir que la matriz $\mathbf{A}$ es de rango (rank) uno. Así pues los pesos $\mathbf{w}$ corresponden al eigenvector asociado a su único eigenvalor diferente a cero. Por ejemplo, supóngase que se tienen los pesos $\mathbf{w} = (1, 6, 2)$. Así constituido, el sistema lineal anterior tendría la forma

$$\begin{pmatrix} 1/1 & 1/6 & 1/2 \\ 6/1 & 6/6 & 6/2 \\ 2/1 & 2/6 & 6/6 \end{pmatrix} \begin{pmatrix} 1 \\ 6 \\ 2 \end{pmatrix} = 3 \begin{pmatrix} 1 \\ 6 \\ 2 \end{pmatrix}. \tag{3.3}$$

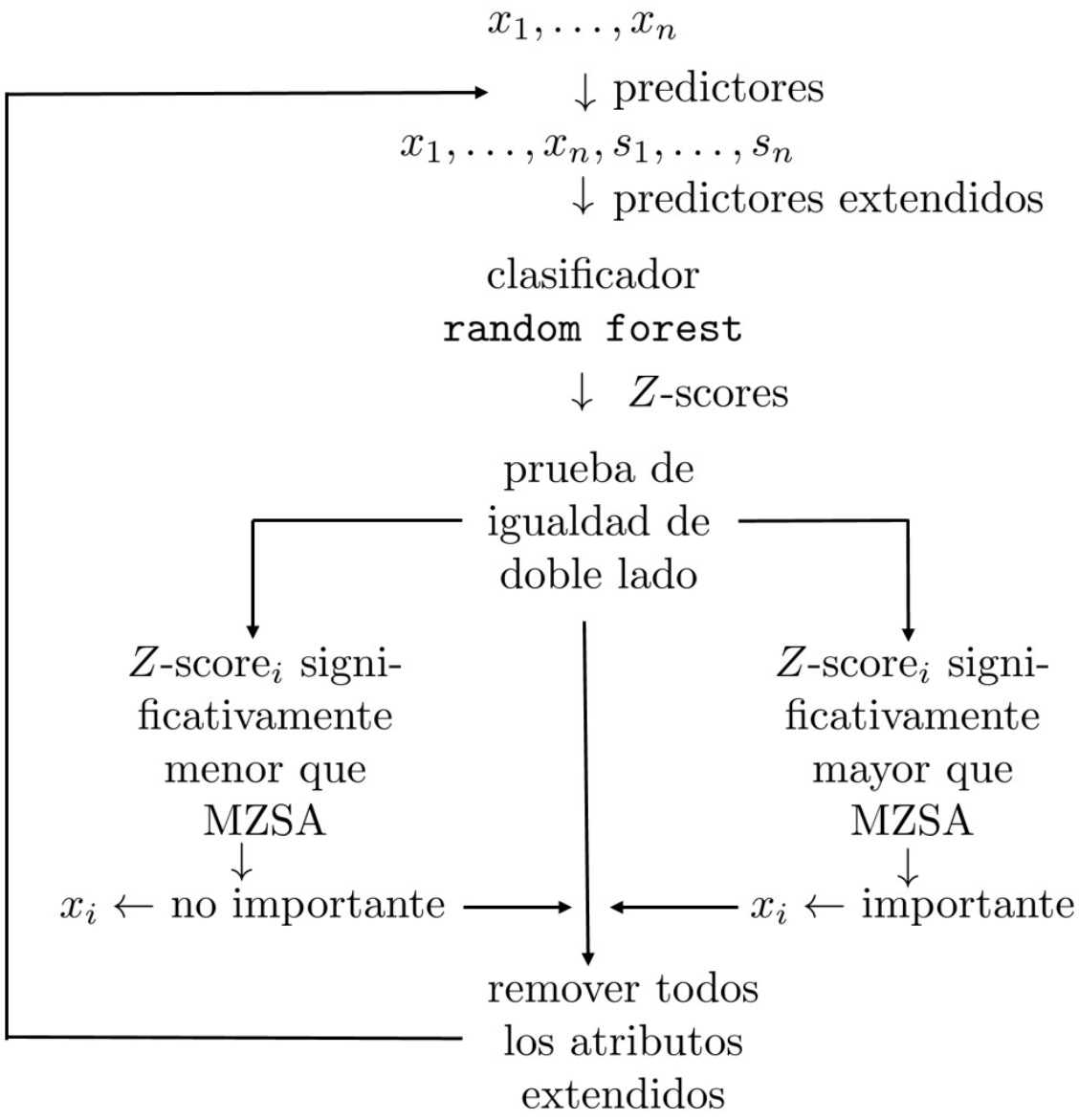


Figura 3.3: Diagrama de flujo para el algoritmo de Boruta. El algoritmo clasifica las variables entre importantes y no importantes basado en la comparación de la degradación del clasificador de las propias variables y variables *sombra* que se han insertado entre los predictores. Las variables sombra corresponden a los atributos originales pero sus valores han sido mezclados entre las observaciones.

La anterior igualdad se mantiene si multiplicamos ambos lados por un factor de escala. Una forma de eliminar esta ambigüuedad consiste en restringuir $\mathbf{w}$ a que tenga norma uno. Esto resulta en el vector $\mathbf{w}^T = (0.1562, 0.937, 0.3123)$.

El establecimiento de las prioridades es una tarea de gestión que define que cosas son importantes. Para facilitar el establecimiento de la prioridades Saaty [57] estableció un procedimiento en el cual se realizan comparaciones por pares entre los índices de desempeño, asignando un valor de importancia de uno con respecto al otro. La interpretación del valor de asignación es el siguiente:

- 1, igualmente importantes
- 3, algo más importante
- 5, más importante
- 7, mucho más importante
- 9, absolutamente más importante

donde los valores intermedios permiten dar una valoración más fina de la importancia relativa.

Para asignaciones arbitrarias de las prioridades, los cocientes w_i/w_j no necesariamente resulta en una matriz de rango uno. Considere el caso de la matriz

$$\mathbf{A} = \begin{pmatrix} 1 & 7 & 3 \\ 1/7 & 1 & 1 \\ 1/3 & 1/1 & 1 \end{pmatrix}, \tag{3.4}$$

cuya descomposición en valores singulares resulta en el producto (redondeado a las centésimas)

$$\begin{aligned} \mathbf{A} &= \mathbf{USV}^T, \\ &= \begin{pmatrix} 0.97 & 0.24 & 0.00 \\ 0.17 & 0.67 & -0.72 \\ 0.17 & 0.70 & 0.69 \end{pmatrix} \begin{pmatrix} 7.91 & & \\ & 0.73 & \\ & & 0.13 \end{pmatrix} \begin{pmatrix} 0.13 & 0.90 & 0.41 \\ -0.12 & 0.43 & 0.90 \\ 0.98 & -0.07 & -0.17 \end{pmatrix}, \\ &= \begin{pmatrix} 1.02 & 6.93 & 3.16 \\ 0.18 & 1.20 & 0.55 \\ 0.18 & 1.22 & 0.56 \end{pmatrix}. \end{aligned} \tag{3.5}$$

Utilizando el criterio de tomar como solución para los pesos el eigenvector correspondiente a valor singular mayor resulta en la siguiente aproximación a la matriz $\mathbf{A}$

$$\mathbf{A} \approx \begin{pmatrix} 0.97 \\ 0.17 \\ 0.17 \end{pmatrix} \begin{pmatrix} 7.91 \end{pmatrix} \begin{pmatrix} 0.13 & 0.90 & 0.41 \end{pmatrix} = \begin{pmatrix} 1.02 & 6.93 & 3.16 \\ 0.18 & 1.20 & 0.55 \\ 0.18 & 1.22 & 0.56 \end{pmatrix}. \tag{3.6}$$

Es interesante notar que el análisis que hemos hecho al introducir valores a la matriz $\mathbf{A}$ no obedece las propiedades del cociente entre rangos de valores. Por ejemplo, en la primera hilera el valor de 7 es un poco más del doble que el valor de 3, queriendo decir que el

primer atributo comparado con el segundo es más del doble de importante que el primero comparado con el tercero. Sin embargo, en la tercera hilera, el segundo atributo parece ser tan importante como el tercero. Lo anterior manifiesta una inconsistencia que es capturada por la descomposición en valores singulares. Saaty mide la consistencia como proporcional a la diferencia entre el eigenvalor máximo calculado y el teórico para una martiz consistente e inversamente proporcional al valor téorico, es decir

$$CI = \frac{\lambda_{\text{máx}} - n}{n-1}. \tag{3.7}$$

3.4. Establecimiento del Ranking

En el método de VIKOR[51] se busca establecer un compromiso en un proceso de decisión de múltiples criterios. El método se basa en la métrica L_p definida como

$$L_{pj} = \left\{ \sum_{i=1}^{n} \left(w_i \frac{\mid f_i^* - f_{ij} \mid}{\mid f_i^* - f_i^- \mid} \right) \right\}^{1/p}, \tag{3.8}$$

donde j corresponde a cada alternativa e i corresponde a los índices de desempeño. Por su lado, f_i^* y f_i^- corresponden a las mejores y peores valores del índice de desempeño, f_{ij} es el valor del índice i para la alternativa j y p puede tomar algun valor entre 1 e ∞. El método de VIKOR calcula los índices S_j y R_j asociados con la útilidad grupal máxima y el mínimo rechazo individual de las alternativas, respectivamente. Los valores corresponden con L_1 y L_∞. Estos valores son calculados como

$$S_j = \sum_{i=1}^{n} \left(w_i \frac{f_i^* - f_{ij}}{f_i^* - f_i^-} \right), \tag{3.9}$$

y

$$R_j = \max_i \left(w_i \frac{f_i^* - f_{ij}}{f_i^* - f_i^-} \right). \tag{3.10}$$

Es decir, el valor S_j recoge la suma de las diferencias de los atributos respecto al valor máximo para ese atributo, mientras que R_j recoge el atributo más diferente. En ambos casos, las variables son normalizadas para dar valores entre cero y uno.

Enseguida, con los valores de S_j y R_j se calcula un valor de compromiso Q_j entre las estrategias grupales e individuales, sesgadas por un valor de peso v como

$$Q_j = v \left(\frac{S_j - S^*}{S^- - S^*} \right) + (1-v) \left(\frac{R_j - R^*}{R^- - R^*} \right), \tag{3.11}$$

donde nuevamente R_j^- y S_j^-, y R_j^* y S_j^*, corresponden a los valores más altos y más bajos de las medidas S_j y R_j, respectivamente.

El ranking se obtiene ordenando decrecientemente S, R y Q. La alternativa d es seleccionada como la mejor si se cumple que

$$Q(a) - Q(d) \geq 1/(J-1), \tag{3.12}$$

Cuadro 3.1: Ejemplo de alternativas y atributos. Supóngase que se tienen cuatro alternativas, cada una de ellas caracterizada por tres atributos. El problema es ordenar las alternativas por orden de importancia.

		alternativas			
atributos		1	2	3	4
	a	0.50	0.25	0.50	0.37
	b	0.51	0.69	0.33	0.34
	c	1.72	1.33	1.03	2.00

donde J es el número de alternativas y a es una alternativa, y si el proceso muestra suficiente estabilidad al tener d mejor valor de R y S que a.

Para ilustrar el esquema anterior, supóngase que se tienen las alternativas y atributos descritas en la Tabla 3.4 y los pesos resultantes del ejercicio en la sección anterior.

Parte II

Clasificación y Regresión

CAPÍTULO 4

Clasificación

Ahora enfocamos nuestro interés a las tareas de aprendizaje supervisado e inferencia, tal como se ilustra en la Figura 1.4. Dado un conjunto de datos muestrales $\{\mathbf{x}_i, \mathbf{w}_i\}$, nuestro problema consiste en encontrar el valor del estado del mundo $\mathbf{w}$ para un dato observado $\mathbf{x}^*$. Cuando el valor de $\mathbf{w}$ se da en valores continuos tenemos un problema de *regresión*. Aquí la cuestión principal consiste en realizar el ajuste de los datos a una curva que tomamos como modelo del mundo. Con ello, el propósito es determinar el valor del estado del mundo para un cierto valor $\mathbf{x}^*$. Cuando por el contrario, $\mathbf{w}$ toma valores discretos tenemos un problema de *clasificación*. En este sentido la cuestión es la asignación de $\mathbf{x}^*$ a una clase.

Así pues, los componentes de nuestro problema son por un lado un modelo del mundo $p(\mathbf{w}|\mathbf{x})$ que nos proporciona el valor del estado dada una observación $\mathbf{x}$. Por otro lado, tenemos un algoritmo de aprendizaje, el cual puede ser discriminativo o generativo. En el *modelo discriminativo* $p(\mathbf{w}|\mathbf{x})$, tenemos una expresión que nos dice como se relaciona el estado del mundo con las entradas. En el *modelo generativo* $p(\mathbf{x}|\mathbf{w})$, definimos una relación que nos dice como se producen los puntos que observamos dados estados particulares del mundo. En el modelo discriminativo debemos obtener los parámetros θ que ajustan el modelo para relacionar de la mejor manera los datos $\mathbf{x}$ con el estado del mundo $\mathbf{w}$. En el modelo generativo incorporamos conocimiento del mundo en forma de *priors*. El último componente de nuestro sistema de solución es el modelo de inferencia $p(\mathbf{w}|\mathbf{x}, \theta)$, mediante el cual podamos saber el valor del estado $\mathbf{w}$ a partir de los datos $\mathbf{x}$ y los parámetros aprendidos θ.

A continuación exploramos ejemplos de procesos de clasificación con modelos discriminativos y generativos. El problema de la clasificación consiste en asignar la pertenencia a una clase w, dentro de un conjunto discreto $\{1, \ldots, K\}$, a una observación $\mathbf{x}$. Siguiendo con la aproximación que hemos desarrollado en el curso, nos interesa obtener la probabilidad de una cierta clase dada una observación $p(w|\mathbf{x})$.

Estas notas están basadas en el libro de Simon Prince[56].

4.1. Modelo Discriminativo

Para ejemplificar el desarrollo de clasificadores binarios lineales, nos situamos en el escenario en donde hay una aeronave no tripulada que necesita distinguir entre el cielo y la tierra. Para ello, vuelos anteriores han tomado imágenes, las cuales han sido cuidadosamente marcadas para distinguir entre ambas clases. Un ejemplo de estas imágenes se encuentra en la Figura 4.8. En total se tienen 12,389 de estas imágenes. Las imágenes tienen diferentes

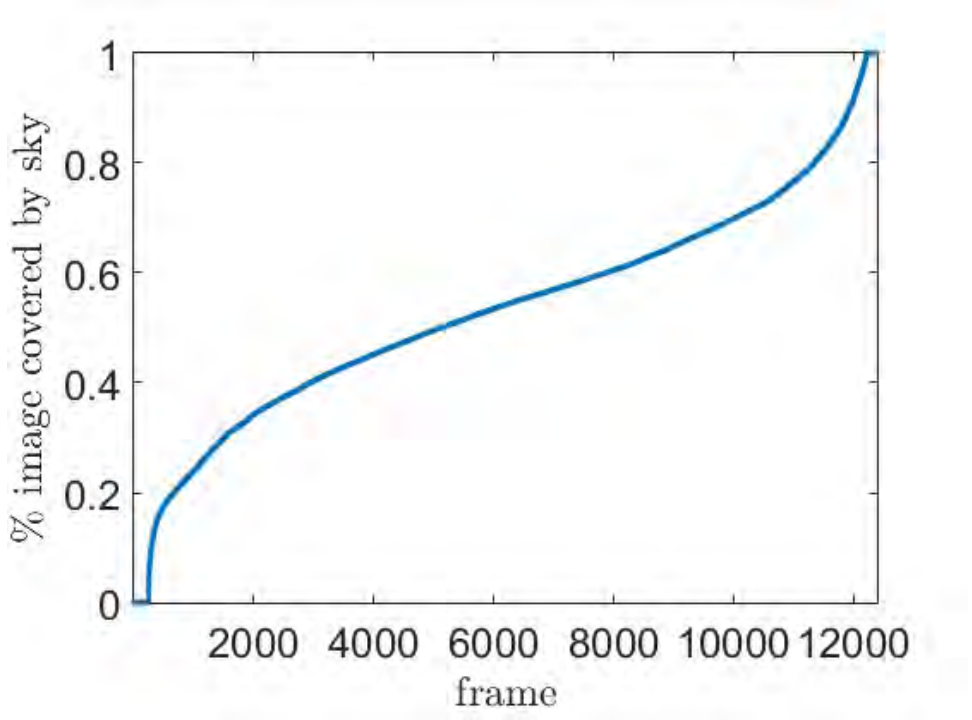


Figura 4.1: Cobertura de cielo. Algunas imágenes no contienen porciones de cielo, algunas otras no tienen porciones de tierra. En general, si ordenamos por porcentaje de cobertura tenemos la expresión dada por la curva.

proporciones de cobertura de cielo/tierra, tal como lo ilustra la Figura 4.1. Para este ejercicio caracterizaremos los pixeles de la imagen por su color. Para ello, la representación RGB será cambiada a La*b*, pues hay algunos estudios que parecen indicar que este espacio de color mapea la percepción humana a un espacio con una métrica Euclidiana[15]. Para ello, las imágenes primero pasadas a través de un filtro bilateral[68], el cual reduce el ruido tomando en cuenta tanto la distribución espacial como el rango de representación de los colores. Para ganar un poco de intuición respecto a la clasificación que realizaremos primero observamos la distribución de las bandas de color para todas las imágenes. Para ello, obtenemos el histograma normalizado para cada una de las bandas de color por separado. El resultado se muestra en la Figura 4.2(a)-(c). Debido a esta separación, escogemos hacer nuestra clasificación sobre las bandas L y b*, dejando de lado la banda a*. En la Figura 4.2(d) se muestra la representación de las bandas Lb*.

4.1.1. Error Mínimo de Clasificación

En una función de densidad de probabilidad (pdf) como las que se presentan en la Figura 4.2(a), la división óptima se da en el punto central en donde ambas pdfs tienen el mismo valor. Para ver por qué note que la probabilidad de cometer un error p_e es igual a la suma de los errores de clasificar erroneamente un dato. Es decir, supóngase que en un problema se tiene la posibilidad de escoger un valor $x_0 \in \mathcal{R}$ que distinga dos clases (ver Figura 4.3). Esto genera dos regiones R_1 y R_2 sobre las clases w_1 y w_2 respectivamente. El error de clasificación

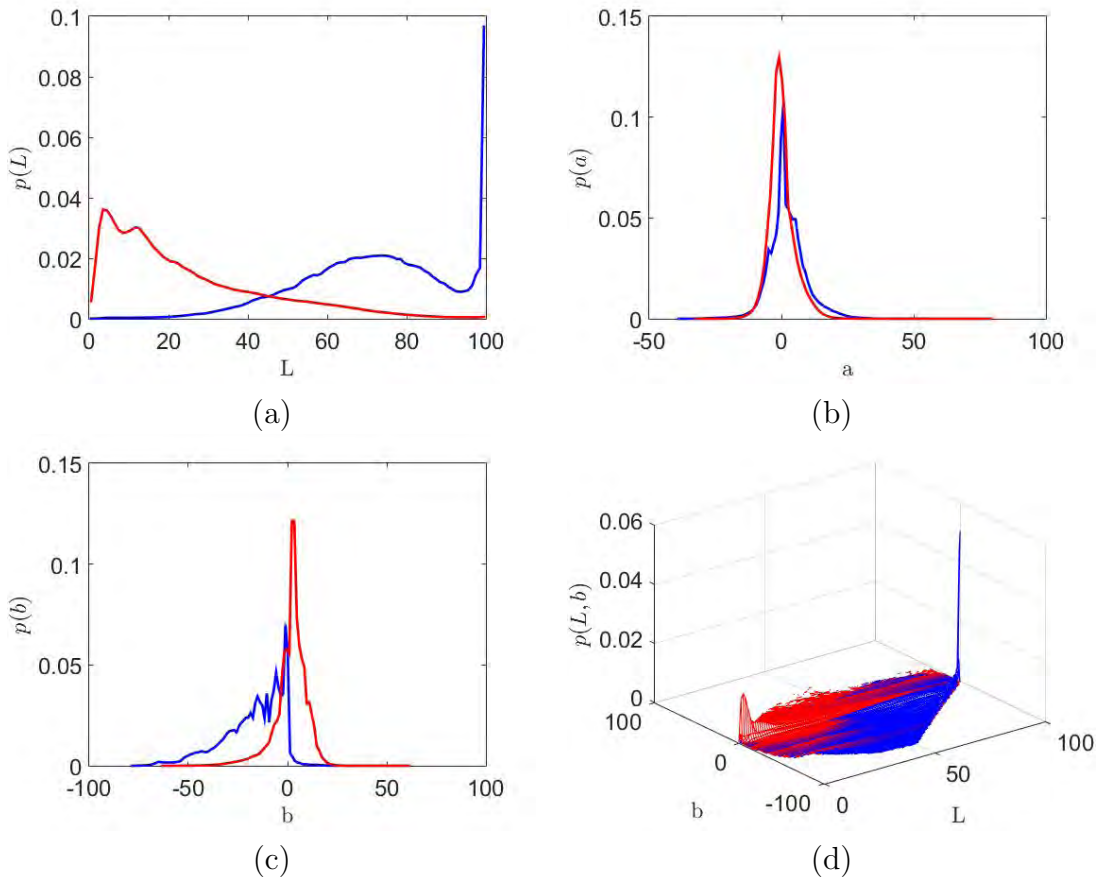


Figura 4.2: Histogramas normalizados para las bandas de color en el modelo La*b*. Notese que mientras que sobre la proyección en L y en b* hay una cierta distinción entre las clases, la distinción sobre a* no lo es tanto.

puede definirse como[66]

$$p_e = p(x \in R_2, w_1) + p(x \in R_1, w_2). \tag{4.1}$$

Es decir, un error es que, al establecer un criterio de clasificación x_0, un punto quede en la región R_2 siendo que el punto pertenece a la clase w_1 o alternativamente que el punto quede en la región R_1 cuando pertenece a la clase w_2. Aplicando probabilidad condicional las expresiones pueden cambiar a

$$p_e = p(x \in R_2|w_1)p(w_1) + p(x \in R_1|w_2)p(w_2). \tag{4.2}$$

Alternativamente, tomando en consideración las regiones

$$p_e = p(w_1)\int_{R_2} p(x|w_1)dx + p(w_2)\int_{R_1} p(x|w_2)dx. \tag{4.3}$$

Aplicando la regla de Bayes tenemos

$$p_e = p(w_1)\int_{R_2} p(w_1|x)p(x)/p(w_1)dx + p(w_2)\int_{R_1} p(w_2|x)p(x)/p(w_2)dx, \tag{4.4}$$

la cual puede simplificarse a

$$p_e = \int_{R_2} p(w_1|x)p(x)dx + \int_{R_1} p(w_2|x)p(x)dx. \tag{4.5}$$

Para continuar avanzado, hay que notar que la probabilidad de la clase w_1 puede evaluarse como

$$p(w_1) = \int_{R_1} p(w_1|x)p(x)dx + \int_{R_2} p(w_1|x)p(x)dx, \tag{4.6}$$

de donde podemos obtener la expresión

$$\int_{R_2} p(w_1|x)p(x)dx = p(w_1) - \int_{R_1} p(w_1|x)p(x)dx. \tag{4.7}$$

Sustituyendo (4.7) en (4.5) tenemos

$$p_e = p(w_1) - \int_{R_1} p(w_1|x)p(x)dx + \int_{R_1} p(w_2|x)p(x)dx = p(w_1) - \int_{R_1} (p(w_1|x) - p(w_2|x))p(x)dx. \tag{4.8}$$

Una derivación similar puede llevarse a cabo con $p(w_2)$, partiendo de la expresión equivalente a (4.6), para llegar a la expresión

$$p_e = p(w_2) + \int_{R_2} (p(w_1|x) - p(w_2|x))p(x)dx. \tag{4.9}$$

Igualando (4.8) y (4.9) llegamos a la conclusión que el error mínimo se da cuando $p(w_1) = p(w_2)$. Con ese antecedente, la localización de la posición del mínimo se facilita, pues podemos construir la relación

$$\int_{R} p(w_1|x)p(x)dx = \int_{R} p(w_2|x)p(x)dx, \tag{4.10}$$

de tal manera que el objetivo se convierte en maximizar el cociente

$$\frac{p(w_1|x)p(x)}{p(w_2|x)p(x)} = \frac{p(w_1, x)}{p(w_2, x)}. \tag{4.11}$$

En el caso de la Figura 4.3, donde tenemos dos Gaussianas. El cociente que se quiere resolver está dado por

$$\frac{p(w_1, x)}{p(w_2, x)} = \frac{\pi \mathrm{Norm}_x(\mu_1, \sigma)}{(1-\pi)\mathrm{Norm}_x(\mu_2, \sigma)}. \tag{4.12}$$

En general, es posible probar[58], que para espacios con características $\mathbf{x}$, la relación

$$\frac{p(w_1, \mathbf{x})}{p(w_2, \mathbf{x})} = \frac{\pi \mathrm{Norm}_{\mathbf{x}}(\mu_1, \mathbf{\Sigma})}{(1-\pi)\mathrm{Norm}_{\mathbf{x}}(\mu_2, \mathbf{\Sigma})}, \tag{4.13}$$

puede resolverse en

$$\ln\left\{\frac{p(w_1, \mathbf{x})}{p(w_2, \mathbf{x})}\right\} = \theta^T \mathbf{x} + \theta_0. \tag{4.14}$$

donde θ contiene los parámetros y θ_0 es un término conocido como *bias*.

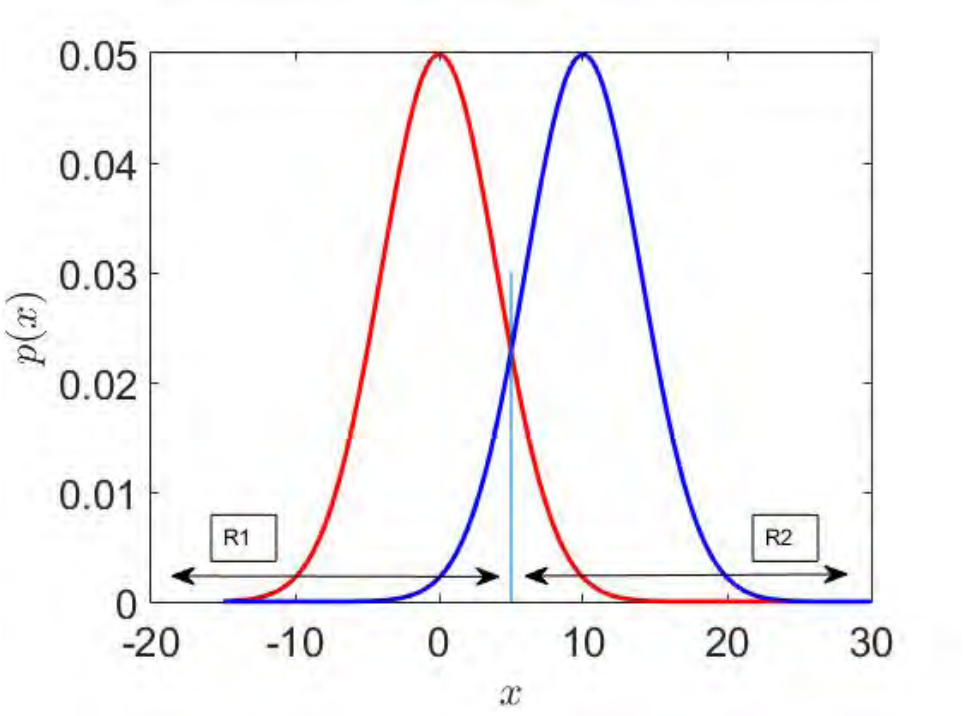


Figura 4.3: Clasificación erronea. Una vez establecido un criterio de clasificación, el error de clasificación es la suma de los errores de asignar a la clase equivocada.

4.1.2. Regresión Logística

El cociente en (4.14) puede ser representado expandido usando marginzalización. Esto resulta en una expresión en términos de condicionales, tal como (haciendo un abuso de notación agregamos un uno a $\mathbf{x}$ y absorbemos θ_0 en θ)

$$\ln \frac{p(w_1|\mathbf{x})p(\mathbf{x})}{p(w_2|\mathbf{x})p(\mathbf{x})} = \ln \frac{p(w_1|\mathbf{x})}{p(w_2|\mathbf{x})} = \theta^T \mathbf{x}, \tag{4.15}$$

o de otra forma, usando propiedades de los logaritmos, tenemos

$$\ln \frac{p(w_2|\mathbf{x})}{p(w_1|\mathbf{x})} = -\theta^T \mathbf{x}. \tag{4.16}$$

Aplicando exponenciales en ambos lados de la ecuación, tenemos la expresión

$$\frac{p(w_2|\mathbf{x})}{p(w_1|\mathbf{x})} = \exp\{-\theta^T \mathbf{x}\}, \tag{4.17}$$

o alternativamente

$$p(w_2|\mathbf{x}) = (1 - p(w_1, \mathbf{x})) = p(w_1|\mathbf{x}) \exp(-\theta^T \mathbf{x}). \tag{4.18}$$

Resolviendo para $p(w_1, \mathbf{x})$ tenemos

$$p(w_1|\mathbf{x}) = \frac{1}{1 + \exp(-\theta^T \mathbf{x})} = g(\theta, \mathbf{x}), \tag{4.19}$$

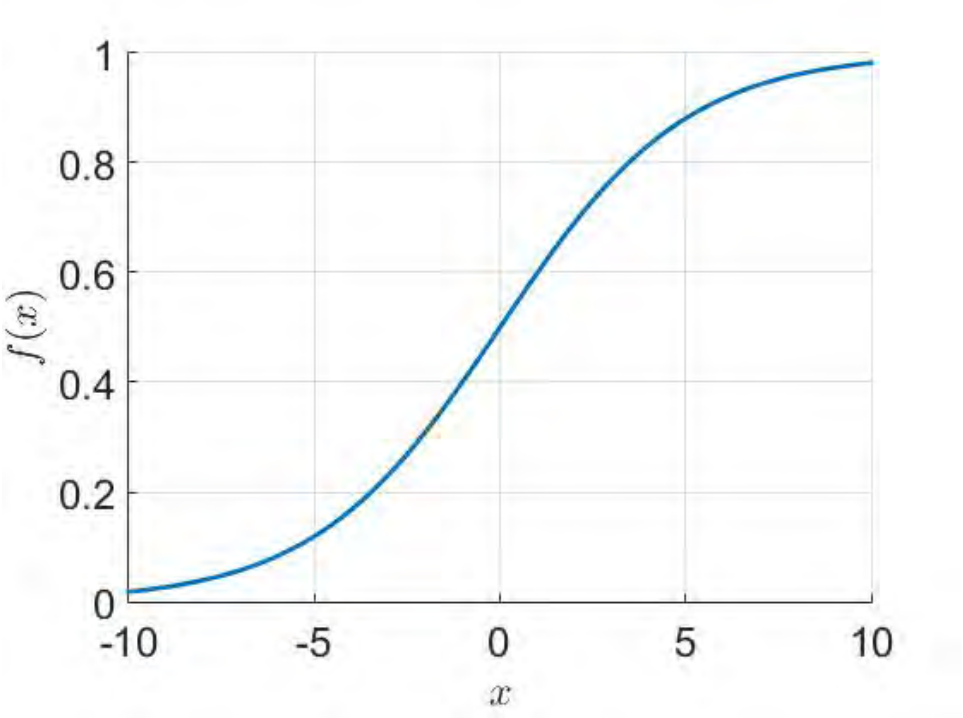


Figura 4.4: Función sigmoide. Entre los usos que esta función tiene es la establecer un mapeo entre el conjunto de los reales y valores entre cero y uno. Note que en los extremos la densidad de puntos es menor, haciendo que las cotas varien menos cuando cambia el argumento.

la cual es conocida como función sigmoide (ver Figura 4.4). La función sigmoide mapea valores en el rango $(-\infty, \infty)$ al rango $(0, 1)$. En general encuentra gran aplicación en áreas tan diversas como las redes neuronales, el mapeo no lineal, y el estudio de crecimiento poblacional, entre otras.

Con lo anterior podemos entonces definir una función de ML para el proceso que nos permita distinguir entre las clases cielo y tierra. Este proceso puede ser descrito por una pdf Bernoulli tal que

$$L(\mathbf{w}) = \prod_{i=1}^{n} p(w_i|\mathbf{x}_i) = \prod_{i=1}^{n} g(\theta, \mathbf{x})^{w_i}(1 - g(\theta, \mathbf{x}))^{1-w_i}. \tag{4.20}$$

Aplicando logaritmos sobre (4.20) llegamos a la expresión

$$\ln L(\mathbf{w}) = \sum_{i=1}^{n} w_i \ln g(\theta, \mathbf{x}) + (1 - w_i) \ln(1 - g(\theta, \mathbf{x})). \tag{4.21}$$

Para obtener los parámetros θ derivamos (4.21) e igualamos con cero. La derivada resulta en la expresión

$$\partial \ln L(\mathbf{w})/\partial\theta = \sum_{i=1}^{n} \left\{ \frac{w_i}{g(\theta, \mathbf{x})} \nabla_\theta g(\theta, \mathbf{x}) - (1 - w_i) \frac{\nabla_\theta g(\theta, \mathbf{x})}{(1 - g(\theta, \mathbf{x}))} \right\}, \tag{4.22}$$

donde el gradiente de $g(\theta, \mathbf{x})$ resulta en

$$\nabla_\theta g(\theta, \mathbf{x}) = \frac{\mathbf{x} \exp\left(\theta^T \mathbf{x}\right)}{(1 - \exp\left(\theta^T \mathbf{x}\right))^2} = g(\theta, \mathbf{x})(1 - g(\theta, \mathbf{x})). \tag{4.23}$$

Sustituyendo (4.23) en (4.24) tenemos

$$\partial \ln L(\mathbf{w})/\partial\theta = \sum_{i=1}^{n} \left\{ \frac{w_i}{g(\theta, \mathbf{x})} g(\theta, \mathbf{x})(1 - g(\theta, \mathbf{x})) - (1 - w_i) \frac{g(\theta, \mathbf{x})(1 - g(\theta, \mathbf{x}))}{(1 - g(\theta, \mathbf{x}))} \right\}. \tag{4.24}$$

Simplificando, llegamos a la expresión

$$\partial \ln L(\mathbf{w})/\partial\theta = \sum_{i=1}^{n} \left\{ (w_i - g(\theta, \mathbf{x}))\mathbf{x} \right\}, \tag{4.25}$$

el cual puede ser resuelto mediante gradiente ascendente como

$$\theta = \theta + \eta \nabla_\theta \ln L, \tag{4.26}$$

donde ∇ es una constante de aprendizaje. El resultado de la clasificación se muestra en la Figura 4.5. Los valores de θ para el clasificador son $\phi_L = 0.086$, $\phi_b = -0.1315$, y la intersección con el eje $\phi_0 = -5.20$.

4.2. Modelo Generativo

Dada la expresión sobre la asignación de una clase w a una observación $\mathbf{x}$, dada por

$$p(w|\mathbf{x}) = \frac{p(\mathbf{x}|w)p(w)}{p(\mathbf{x})}, \tag{4.27}$$

en el modelo generativo comenzamos con la expresión de la verosimilitud $p(\mathbf{x}|w)$. A ella, le incluimos conocimiento *a priori* sobre el estado del mundo y normalizamos por la evidencia $p(\mathbf{x})$. En particular ahora estamos interesados en problemas en donde hay dos clases. En esas condiciones la expresión anterior toma la forma

$$p(w|\mathbf{x}) = \frac{p(\mathbf{x}|w)p(w)}{p(\mathbf{x}|w = 0)p(w = 0) + p(\mathbf{x}|w = 1)p(w = 0)}. \tag{4.28}$$

4.2.1. Distribuciones de Probabilidad Marginales

Regresando a nuestro problema de detección de las clases `Cielo` y `Tierra` , notamos que la variable aleatoria L puede ser aproximada por una función Weibull, tal como visualmente se puede apreciar en la Figura 4.9. Para la clase `Cielo` , los parámetros de forma y escala tienen valores 4.46 y 78.93, respectivamente. Correspondientemente, la clase `Tierra` tiene parámetros de forma y escala con valores 1.23 y 26.32, respectivamente. Con respecto a la componente b, hay varias posibles distribuciones. Después de varias posibilidades una que

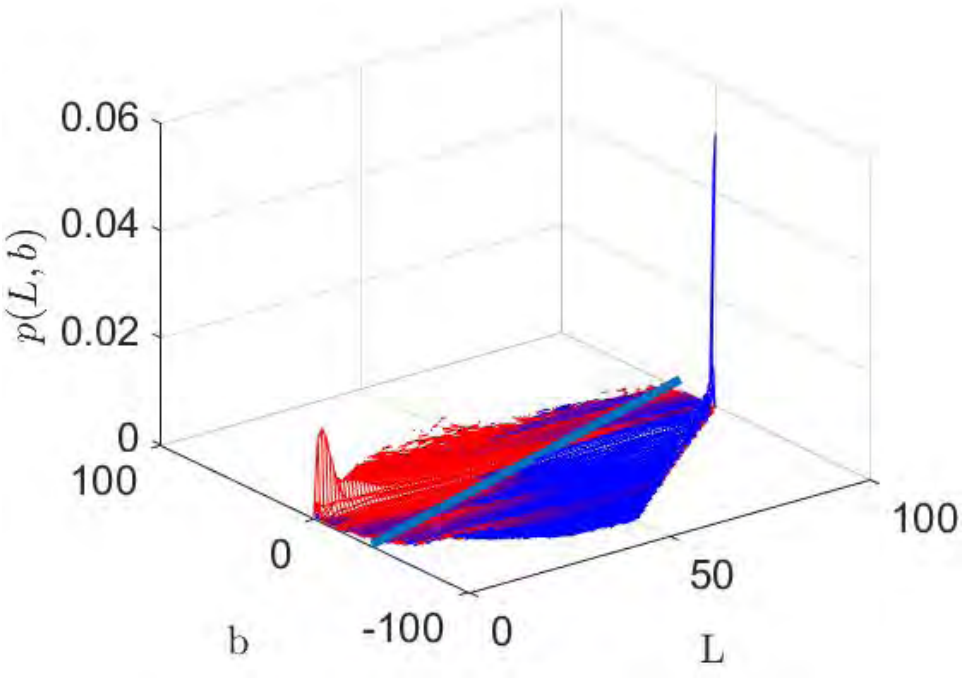


Figura 4.5: Resultado del Clasificador. Una línea se despliega para distinguir las clases cielo/tierra a partir de la información L y b*, en la representación de color CIELab.

parece apropiada podría ser la Gaussiana sesgada, pues tiene soporte en el rango de los números reales y sigue aproximadamente la distribución de datos. Su forma analística está dada de la siguiente manera. Primero, sea la forma normalizada de la Gaussiana (media cero y desviación estándar uno) dada por

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-x^2/2\right), \tag{4.29}$$

y su distribución acumulada está dada por

$$\Phi(x) = \int_{-\infty}^{x} \phi(t)dt = \frac{1}{2}\left(1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right). \tag{4.30}$$

Con esas definiciones la pdf Gaussiana sesgada está dada por la expresión

$$f(x) = 2\phi(x)\Phi(\alpha x). \tag{4.31}$$

Si aplicamos la transformación de localización y escala $x \to (x - \xi)/\omega$, la función de probabilidad puede ser escrita como

$$f(x) = \frac{2}{\omega}\phi\left(\frac{x-\xi}{\omega}\right)\Phi\left(\alpha\frac{x-\xi}{\omega}\right). \tag{4.32}$$

En el caso de nuestro problema el ajuste de los parámetros por ML (ver Figura 4.10) resulta, para la clase `Cielo` en los valores de localización ξ, escala ω y sesgo α, -17.13, 14.63 y -0.9

respectivamente. Correspondientemente, la clase `Tierra` tiene parámetros de localización, escala y sesgo 1.44, 7.85 y -0.57, respectivamente.

Como vemos, aun cuando el ajuste de las marginales es satisfactorio, no existe una pdf conjunta que describa el comportamiento complejo de los datos. La siguiente sección trata sobre una herramienta que permite el manejo de la interacción a partir del conocimiento de las marginales y su correlación.

4.2.2. Modelado Multivariable con Cópulas

La función acumulada de probabilidad (cdf) de una pdf es la acumulación de los valores de probabilidad de un rango inferior hasta el punto de interés. Esta se define, para una variable aleatoria continua, como

$$\mathbb{P}(x) = \int_{-\infty}^{x} p(t)dt. \tag{4.33}$$

Para p dimensiones, la cdf se define como

$$\mathbb{P}(x_1, \ldots, x_p) = \mathbb{P}(t_1 \leq x_1, \ldots, t_p \leq x_p). \tag{4.34}$$

Por su naturaleza, una cdf es una función monotónicamente creciente. Esto es muy relevante porque para funciones continuas, sin intervalos en donde la probabilidad sea cero, implica que la inversa de la cdf existe. Una propiedad interesante es que si se tiene la cdf y se quiere obtener valores de su correspondiene pdf, basta con obtener muestras aleatorias de la distribución uniforme y evaluarlas en la inversa de la cdf. Es decir, si x es una variable aleatoria con pdf arbitraria y correspondiente $\mathbb{P}$ y u es una variable aleatoria con pdf uniforme, entonces la siguiente relación se cumple

$$u = \mathbb{P}(x) \therefore x = \mathbb{P}^{-1}(u). \tag{4.35}$$

En 1959, Sklar[61] planteo que la probabilidad conjunta $\mathbb{P}(x_1, \ldots, x_p)$ podía estimarse a partir de las distribuciones marginales y la relación entre ellas, a lo que llamó cópula. La cópula C es una función de distribución acumulada en términos de las variables aleatorias. Una cópula puede ser definida como

$$\begin{aligned} C(u_1, \ldots, u_p) &= \mathbb{P}(x_1, \ldots, x_p) \\ &= \mathbb{P}(\mathbb{P}_1^{-1}(u_1), \ldots, \mathbb{P}_p^{-1}(u_p)) \end{aligned} \tag{4.36}$$

donde $\mathbb{P}_i$ corresponde a la marginal de la i-ésima variable. La pdf de la cópula puede ser calculada derivando con respecto a las variables $u_1, \ldots, u_p$ y suponiendo independencia como

$$\begin{aligned} \frac{\partial C(u_1, \ldots, u_p)}{\partial u_1} &= \frac{\partial \mathbb{P}(\mathbb{P}_1^{-1}(u_1), \ldots, \mathbb{P}_p^{-1}(u_p))}{\partial u_1} \frac{\partial \mathbb{P}_1^{-1}(u_1)}{\partial u_1} \\ \frac{\partial^2 C(u_1, \ldots, u_p)}{\partial u_1 \partial u_2} &= \frac{\partial \mathbb{P}_1^{-1}(u_1)}{\partial u_1} \frac{\partial^2 \mathbb{P}(\mathbb{P}_1^{-1}(u_1), \ldots, \mathbb{P}_p^{-1}(u_p))}{\partial u_1 \partial u_2} \frac{\partial \mathbb{P}_2^{-1}(u_2)}{\partial u_2} \\ &\vdots \\ c(u_1, \ldots, u_p) &= \frac{p(x_1, \ldots, x_p)}{\prod_{i=1}^{p} p(x_i)}, \end{aligned} \tag{4.37}$$

donde finalmente, la conjunta $p(x_1, \ldots, x_p)$ está dada por

$$p(x_1, \ldots, x_p) = c(u_1, \ldots, u_p) \prod_{i=1}^{p} p(x_i). \tag{4.38}$$

La importancia de (4.38) es que la cópula puede ser obtenida independientemente de los datos, suponiendo algún tipo de relación entre las variables. Algunas de las cópulas más comunes incluyen las elípticas, las cuales corresponden a densidades de la forma $\phi(t) = \Psi(\mathbf{t}^T \mathbf{\Sigma} \mathbf{t})$[55]; y las Arquímedas, populares porque permiten establecer la asociación entre las variables usando un solo parámetro.

Un ejemplo de una cópula elíptica es la cópula normal, la cual puede definir como

$$C(u_1, \ldots, u_p) = \phi_\Sigma \left(\phi^{-1}(u_1), \ldots, \phi^{-1}(u_p) \right), \tag{4.39}$$

donde ϕ es la cdf de una distribución normal y ϕ_Σ es la cdf de una distribución conjunta multivariable con parámetros $\phi_i^{-1}(u_i)$.

En general, las cúpulas Arquimedeas están dadas por la fórmula general

$$C(u_1, \ldots, u_p) = \varphi^{-1} \left(\varphi(u_1) + \varphi^{-1}(u_p) \right), \tag{4.40}$$

donde φ puede estar definida como

$$\varphi(t) \quad = \quad \begin{cases} t^{-\alpha} - 1 & \text{Clayton,} \\ \ln \dfrac{\exp(\alpha t) - 1}{\exp(\alpha t) - 1} & \text{Frank,} \\ (-\ln t)^{\alpha} & \text{Gumbel.} \end{cases} \tag{4.41}$$

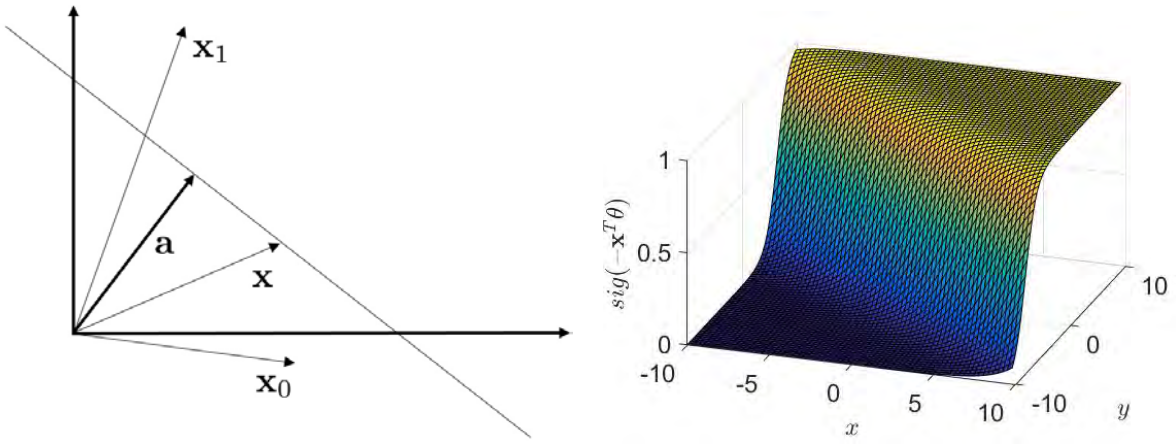
4.3. Regresión Logística

Siguiendo la notación ilustrada en la Figura 4.6, una línea recta puede ser definida por la ecuación

$$\begin{aligned} ax + by - c &= 0, \\ \begin{pmatrix} x & y & -1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} &= 0 \\ \mathbf{x}^T \theta &= 0, \end{aligned} \tag{4.42}$$

donde si imponemos la restricción de que $\sum_{i=1}^{n} \phi_i^2 = 1$, ϕ_0 es la distancia entre el origen y la línea recta. En esta interpretación, para un punto cualquiera en el espacio $\mathbf{x}$, el producto $\mathbf{x}^T \theta$ es la distancia con signo a la línea recta. Entre mayor sea el número se encontrarán más arriba y entre menor se encontrará más abajo. En términos probabilísticos esta noción puede ser capturada por la curva sigmoide, la cual puede ser definida como

$$p(\mathbf{x}|\theta) = \frac{1}{1 + \exp(-\mathbf{x}^T \theta)}. \tag{4.43}$$



(a) Línea recta (b) Curva sigmoide

Figura 4.6: Una línea recta es el lugar geométrico en donde $\mathbf{a}^T\mathbf{x} = 0$. Por ejemplo en esta ilustración con $\mathbf{a} = (\phi_1, \phi_2, \phi_0)^T$, $\mathbf{x} = (x, y, 1)^T$. Puntos donde $\mathbf{a}^T\mathbf{x}_1 > 0$ se encuentran en la parte superior y alternativamente $\mathbf{a}^T\mathbf{x}_0 < 0$ se encuentran en la parte superior. Imponiendo la restricción $\phi_1^2 + \phi_2^2 = 1$, ϕ_0 es la distancia entre el origen y la línea recta. La noción de distancia en un marco acotado a cero y uno puede ser capturado por la curva sigmoide. Entre mayor sea el número más cercano será a uno. Alternativa,ente entre menor la distancia más cercano será a cero.

La distinción que resulta de esta evaluación da pie a la clasificación dada por la expresión

$$p(w|\mathbf{x}, \theta) = \lambda^w(1-\lambda)^{1-w} = \text{Bern}_w(\lambda), \tag{4.44}$$

donde $\lambda = \text{sig}\left(-\theta^T\mathbf{x}\right)$.

Dado un conjunto de observaciones y etiquetas de la forma $\{\mathbf{x}_i, w\}_{i=1}^I$, los parámetros de la mejor línea que separa las clases puede ser obtenida maximizando la expresión

$$\begin{aligned} p(w|\mathbf{X}, \phi) &= \prod_{i=1}^{I} \lambda^{w_i}(1-\lambda)^{1-w_i}, \\ &= \prod_{i=1}^{I} \left(\frac{1}{1+\exp\left(-\mathbf{x}_i^T\phi\right)}\right)^{w_i} \left(\frac{\exp\left(-\phi^T\mathbf{x}_i\right)}{1+\exp\left(-\mathbf{x}_i^T\theta\right)}\right)^{1-w_i}, \\ &= \prod_{i=1}^{I} \text{Bern}_{w_i}(\text{sig}(\mathbf{x}_i^T\phi)). \end{aligned} \tag{4.45}$$

La forma estándar de obtener los parámetros es aplicando logaritmos para simplificar la expresión, derivando con respecto a las variables de interés, igual a cero y resolver para las

incógnitas. La ecuación para minimizar, después de aplicar logaritmos, podría tener la forma

$$\begin{aligned}\ln p(w|\mathbf{X},\phi) &= \sum_{i=1}^{I}\left\{w_i \ln\left(\frac{1}{1+\exp\left(-\mathbf{x}_i^T\phi\right)}\right)+(1-w_i)\ln\left(\frac{\exp\left(-\phi^T\mathbf{x}_i\right)}{1+\exp\left(-\mathbf{x}_i^T\phi\right)}\right)\right\},\\ &= \sum_{i=1}^{I}\left\{-w_i\ln\left(1+\exp\left(-\mathbf{x}_i^T\phi\right)\right)+(1-w_i)\left(-\phi^T\mathbf{x}_i-\ln\left(1+\exp\left(-\mathbf{x}_i^T\phi\right)\right)\right)\right\},\\ &= L(\phi|\mathbf{X},\mathbf{w}).\end{aligned} \tag{4.46}$$

Después de alguna manipulación algebraica, la primera derivada resulta en la expresión

$$\partial L(\phi|\mathbf{X},\mathbf{w})/\partial\phi = -\sum_{i=1}^{I}\left(\frac{1}{1+\exp\left(-\mathbf{x}_i^T\phi\right)}-w_i\right)\mathbf{x}_i. \tag{4.47}$$

Para resolver el valor de los parámetros ϕ que maximizan esta expresión, Prince[56] plantea el uso de el método de Newton, el cual requiere un esquema iterativo donde el valor nuevo se obtiene a partir del valor anterior, tal como

$$\phi^t = \phi^{t-1} + \alpha\left(\partial^2 L/\partial\phi^2\right)^{-1}\partial L/\partial\phi. \tag{4.48}$$

donde α es un parámetro de aprendizaje que se utiliza para determinar la magnitud del paso que se dará en cada iteración. Para obtener la segunda derivada de L con respecto a ϕ, el Hessiano, derivamos (4.47) con respecto a ϕ. Esto da como resultado la expresión

$$\partial^2 L(\phi|\mathbf{X},\mathbf{w})/\partial\phi^2 = -\sum_{i=1}^{I}\left(\frac{1}{1+\exp\left(-\mathbf{x}_i^T\phi\right)}\right)\left(\frac{\exp\left(-\mathbf{x}_i^T\phi\right)}{1+\exp\left(-\mathbf{x}_i^T\phi\right)}\right)\mathbf{x}_i\mathbf{x}_i^T, \tag{4.49}$$

El Algoritmo 1 presenta el pseudo-código para el cálculo de los parámetros del hiperplano que mejor distingue dos clases usando regresión logística.

4.4. Regresión Logística Bayesiana

La regresión lógistica Bayesiana reconoce la incertidumbre que se tiene en la estimación de los parámetros ϕ y busca cuantificarla. Para ello, se define la medida de incertidumbre

$$p(\phi|\mathbf{X},\mathbf{w}) = \frac{p(\mathbf{w}|\mathbf{X},\phi)p(\phi)}{p(\mathbf{w}|\mathbf{X})}, \tag{4.50}$$

donde en la sección anterior se ha definido $p(\mathbf{w}|\mathbf{X},\phi) = \text{Bern}_w(\text{sig}(\phi^T\mathbf{x}))$ y a falta de algo mejor se define el prior $p(\phi)$ por $\text{Norm}_\phi(0,\sigma_p^2\mathbf{I})$. Dado que la verosimilitud y el prior no son conjugados, no hay solución cerrada. Prince[56] propone utilizar lo que es conocido como la aproximación Laplaciana del posterior. Supóngase que θ es la posición del máximo para una

Llamada : $\phi \leftarrow$ `Regresión-Lógistica` $(\mathbf{X}, \mathbf{w})$
Entradas: Los datos $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$ y las etiquetas $\mathbf{w} = (w_1, \ldots, w_N)$.
Salidas : Los valores de los parámetros ϕ que definen el hiperplano que mejor distingue las clases.

```
// Inicializa las variables
```
$\phi_{\text{old}} \leftarrow \phi_0$; $\alpha \leftarrow \alpha_0$; $c \leftarrow$ `FALSE`;
while $c \equiv$ `FALSE` **do**
```
    // Inicializa acumuladores
```
    $\mathbf{g} \leftarrow \mathbf{0}_{2\times 1}$; $\mathbf{H} \leftarrow \mathbf{0}_{2\times 2}$;
```
    // Calcula el gradiente y el Hessiano
```
    for $i \leftarrow 1$ **to** n **do**
```
        // Calcula el sigmoide
```
        $\lambda \leftarrow \dfrac{1}{1+\exp\left(-\mathbf{x}_i^T\phi\right)}$;
```
        // Actualiza el gradiente y el Hessiano
```
        $\mathbf{g} \leftarrow \mathbf{g} + \lambda\mathbf{x}_i$; $\mathbf{H} \leftarrow \mathbf{H} + \lambda(1-\lambda)\mathbf{x}_i\mathbf{x}_i^T$;
    end
```
    // Actualiza el valor de φ
```
    $\phi_{\text{new}} \leftarrow \phi_{\text{old}} + \alpha\mathbf{H}^{-1}\mathbf{g}$;
```
    // Verifica convergencia
```
    $c = (|\phi_{\text{new}} - \phi_{\text{old}}| \leq \epsilon)$;
```
    // Continua iterando
```
    $\phi_{\text{old}} \leftarrow \phi_{\text{new}}$;
end

Algorithm 1: Algoritmo para la estimación de los parámetros de regresión logística.

pdf. Entonces, θ es también la posición del máximo para el logaritmo de esa pdf, $q(\theta)$. Esta función puede expandirse en términos de la serie de Taylor como[67]

$$\begin{aligned} q(\theta) &\approx q(\hat{\theta}) + (\theta-\hat{\theta})\dot{q}(\theta) + \frac{1}{2}(\theta-\hat{\theta})^2\ddot{q}(\theta), \\ &= q(\hat{\theta}) + 0 + \frac{1}{2}(\theta-\hat{\theta})^2\ddot{q}(\theta), (\text{ pues } \dot{q}(\theta)=0), \\ &= \kappa + \frac{1}{2}(\theta-\hat{\theta})^2\ddot{q}(\theta), \\ &= \kappa + \frac{(\theta-\tilde{a})^2}{2\tilde{b}^2}, \end{aligned} \tag{4.51}$$

donde $\tilde{a} = \hat{\theta}$ y $\tilde{b} = -\left(\ddot{q}(\theta)\right)^{-1}$ ($\ddot{q}(\theta) < 0$ pues $\hat{\theta}$ está en el máximo).

Para obtener el valor óptimo de los parámetros ϕ hay que evaluar la expresión

$$p(\mathbf{w}|\mathbf{X}, \phi) = \prod_{i=1}^{I} p(w_i|\mathbf{x}_i, \phi)p(\phi). \tag{4.52}$$

Llamada : $< \mu, \mathbf{\Sigma} >\leftarrow$ `Regresión-Lógistica-Bayesiana-Aprendizaje` $(\mathbf{X}, \mathbf{w})$
Entradas: Los datos $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$ y las etiquetas $\mathbf{w} = (w_1, \ldots, w_N)$.
Salidas : Los valores de los parámetros μ y $\mathbf{\Sigma}$ que definen la distribución de probabílidad posterior sobre los parámetros ϕ.

`// Inicializa las variables`
$\phi_{\text{old}} \leftarrow \phi_0$; $\alpha \leftarrow \alpha_0$; $c \leftarrow$ `FALSE`;
while $c \equiv$ `FALSE` **do**
  `// Inicializa acumuladores`
  $\mathbf{g} \leftarrow \mathbf{0}_{2\times 1}$; $\mathbf{H} \leftarrow \mathbf{0}_{2\times 2}$;
  `// Calcula el gradiente y el Hessiano`
  for $i \leftarrow 1$ **to** n **do**
    `// Calcula el sigmoide`
    $\lambda \leftarrow \dfrac{1}{1 + \exp\left(-\mathbf{x}_i^T \phi\right)}$;
    `// Actualiza el gradiente y el Hessiano`
    $\mathbf{g} \leftarrow \mathbf{g} + \lambda \mathbf{x}_i$; $\mathbf{H} \leftarrow \mathbf{H} + \lambda(1-\lambda)\mathbf{x}_i \mathbf{x}_i^T$;
  end
  `// Incluye el bias`
  $\mathbf{g} \leftarrow \mathbf{g} - \phi/\sigma_p^2$; $\mathbf{H} \leftarrow \mathbf{H} - 1/\sigma_p^2$;
  `// Actualiza el valor de` ϕ
  $\phi_{\text{new}} \leftarrow \phi_{\text{old}} + \alpha \mathbf{H}^{-1}\mathbf{g}$;
  `// Verifica convergencia`
  $c \leftarrow (|\phi_{\text{new}} - \phi_{\text{old}}| \leq \epsilon)$;
  `// Continua iterando`
  $\phi_{\text{old}} \leftarrow \phi_{\text{new}}$;
end
`// Obtén los parámetros de la aproximación de Laplace al posterior`
$\mu \leftarrow \phi_{\text{new}}$; $\mathbf{\Sigma} \leftarrow \mathbf{H}^{-1}$;

Algorithm 2: Algoritmo para la estimación de los parámetros de la aproximación Laplaciana del posterior en regresión logística considerando la incertidumbre en la obtención de los parámetros del hiperplano.

Aplicando logaritmos para simplificar la expresión, tenemos

$$\ln p(\mathbf{w}|\mathbf{X},\phi) = L(\phi|\mathbf{X},\mathbf{w}) = \sum_{i=1}^{I} \left\{ \ln p(w_i|\mathbf{x}_i,\phi) + \ln p(\phi) \right\}. \tag{4.53}$$

Para encontrar la posición del máximo derivamos, igualamos a cero y resolvemos para los términos de interés. En este caso, el gradiente y el Hessiano resultan en las ecuaciones[56]

$$\partial L(\phi|\mathbf{X},\mathbf{w})/\partial\phi = -\sum_{i=1}^{I} \left(\frac{1}{1+\exp\left(-\mathbf{x}_i^T\phi\right)} - w_i \right) \mathbf{x}_i - \frac{\phi}{\sigma_p^2}, \text{ y} \tag{4.54}$$

$$\partial^2 L(\phi|\mathbf{X},\mathbf{w})/\partial\phi^2 = -\sum_{i=1}^{I} \left(\frac{1}{1+\exp\left(-\mathbf{x}_i^T\phi\right)} \right) \left(\frac{\exp\left(-\mathbf{x}_i^T\phi\right)}{1+\exp\left(-\mathbf{x}_i^T\phi\right)} \right) \mathbf{x}_i\mathbf{x}_i^T - \frac{1}{\sigma_p^2}. \tag{4.55}$$

Un esquema iterativo de solución basado en el método de Newton nos proporciona los parámetros ϕ y los resultados parciales de gradiente y Hessiano. Este esquema se presente en el Algoritmo 2. Con ello, la aproximación Laplaciana al posterior permite obtener los parámetros

$$\mu = \hat{\phi} \text{ y } \mathbf{\Sigma} = -\left(\frac{\partial^2 L}{\partial\phi^2} \right)^{-1} \Bigg|_{\phi=\hat{\phi}}. \tag{4.56}$$

Para inferir la etiqueta de una nueva observación evaluamos la marginal con respecto a los parámetros del hiperplano ϕ, tal que

$$\begin{aligned} p(w^*|\mathbf{x}^*,\mathbf{X},\mathbf{w}) &= \int p(w^*|\mathbf{x}^*,\phi)p(\phi|\mathbf{X},\mathbf{w})d\phi, \\ &= \int p(w^*|\mathbf{x}^*,\phi)q(\phi)d\phi, \\ &= \int \text{Bern}_w(\text{sig}(\phi^T\mathbf{x}))\text{Norm}_\phi(\mu,\mathbf{\Sigma})d\phi. \end{aligned} \tag{4.57}$$

Para facilitar la evaluación de la integral, Prince[56] propone realizar el cambio de variable $a = \phi^T\mathbf{x}$. En una Gaussiana, el cambio de variable $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ sobre una pdf $\text{Norm}_\mathbf{x}(\mu,\mathbf{\Sigma})$ resulta en la pdf $\text{Norm}_\mathbf{y}(\mathbf{A}\mu+\mathbf{b}, \mathbf{A}\mathbf{\Sigma}\mathbf{A}^T)$[56]. En nuestro caso, la pdf $\text{Norm}_\phi(\mu,\mathbf{\Sigma})$ se transforma en $\text{Norm}_a(\mathbf{x}^{*T}\mu, \mathbf{x}^{*T}\mathbf{\Sigma}, \mathbf{x}^*)$. Con este cambio de variable la integral puede representarse y aproximarse[56] como

$$\begin{aligned} p(w^*|\mathbf{x}^*,\mathbf{X},\mathbf{w}) &= \int p(w^*|a)p(a)da, \\ &\approx \frac{1}{1+\exp\left(-\mu_a/\sqrt{1+\phi\sigma_a^2/8}\right)}, \end{aligned} \tag{4.58}$$

con $\mu_a = \mu^T\mathbf{x}^*$ y $\sigma_a^2 = \mathbf{x}^{*T}\mathbf{\Sigma}\mathbf{x}^*$.

4.5. Regresión Logistica no Lineal

El esquema desarrollado previamente puede ser extendido a problemas donde la frontera de decisión debe tomar una decisión no lineal. Para ello, una estrategia consiste en proyectar las características originales a espacios no lineales de mayor dimensión utilizando la transformación $\mathbf{z} = \mathbf{f}(\mathbf{x})$. Es decir, la probabilidad de observar un estado particular w puede expresarse como

$$p(w|\mathbf{x}, \phi) = \text{Bern}_w(\text{sig}(\phi^T \mathbf{z})) = \text{Bern}_w(\text{sig}(\phi^T \mathbf{f}(\mathbf{x})). \tag{4.59}$$

Algunas de las formas que la función $\mathbf{f}(\mathbf{x})$ puede tomar incluyen

- la función escalón: $z_k = \text{Escalón}(\alpha^T \mathbf{x})$, que es cero para valores del argumento menores que cero y uno en otro caso,
- la función arco tangente: $z_k = \arctan(\alpha^T \mathbf{x})$, y
- la función Gaussiana: $z_k = \exp\left(-\frac{1}{\lambda_0}(\mathbf{x} - \alpha_k)^T(\mathbf{x} - \alpha_k)\right)$,

donde z_k es la k-ésima entrada de $\mathbf{z}$ y $\{\alpha\}_{k=1}^K$ son direcciones de proyección. De esta forma, si el conjunto de parámetros se forma con las variables $\theta = (\phi^T, \alpha_1^T, \ldots, \alpha_K^T)^T$, el gradiente y el Hessiano para obtener su valor óptimo, por el método de Newton, se forman mediante[56]

$$\frac{\partial L}{\partial \theta} = -\sum_{i=1}^{I} (w_i - \text{sig}(a_i)) \frac{\partial a_i}{\partial \theta}, \text{ y} \tag{4.60}$$

$$\frac{\partial^2 L}{\partial \theta^2} = -\sum_{i=1}^{I} \left\{ (\text{sig}(a_i)) (\text{sig}(a_i) - 1) \frac{\partial a_i}{\partial \theta} \frac{\partial a_i}{\partial \theta}^T - (w_i - \text{sig}(a_i)) \frac{\partial^2 a_i}{\partial \theta^2} \right\}, \tag{4.61}$$

donde $a_i = \phi^T \mathbf{f}(\mathbf{x}_i)$.

4.6. Formulación Dual de la Regresión Logística

De forma similar a la regresión, la formulación en base a las características puede ser formulada en términos de las observaciones. Para ello, los parámetros son representados en términos de combinaciones lineales de hileras de $\mathbf{X}$ tal como

$$\phi = \mathbf{X}\psi, \tag{4.62}$$

donde ψ es un vector de $I \times 1$. De esta forma el modelo de regresión lineal puede representarse como

$$\begin{aligned} p(w|\mathbf{X}, \psi) &= \prod_{i=1}^{I} \text{Bern}_{w_i}(\text{sig}(\phi^T \mathbf{x}_i)), \\ &= \prod_{i=1}^{I} \text{Bern}_{w_i}(\text{sig}(\psi^T \mathbf{X}^T \mathbf{x}_i)). \end{aligned} \tag{4.63}$$

Para aprender los parámetros ψ podemos seguir el procedimiento de ML o si nos interesa introducir la incertidumbre de los parámetros mediante una formulación Bayesiana. En el primer caso aplicamos logaritmos, derivamos con respecto a los parámetros, igualamos a cero y resolvemos para los parámetros. Como en el caso primal, los parámetros se obtienen por el método de Newton. Para ello, el gradiente y el Hessiano corresponden a[56]

$$\frac{\partial L}{\partial \phi} = -\sum_{i=1}^{I} \left(\text{sig}(a_i) - w_i\right) \mathbf{X}^T \mathbf{x}_i, \text{ y} \tag{4.64}$$

$$\frac{\partial^2 L}{\partial \phi^2} = -\sum_{i=1}^{I} \left(\text{sig}(a_i)\right) \left(1 - \text{sig}(a_i)\right) \mathbf{X}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{X}. \tag{4.65}$$

Para la formulación Bayesiana nos interesa obtener

$$p(\psi|\mathbf{X}, \mathbf{w}) = \frac{p(\mathbf{w}|\mathbf{X}, \psi) p(\psi)}{p(\mathbf{w}|\mathbf{X})}, \tag{4.66}$$

donde a falta de algo mejor se define el prior $p(\psi)$ por $\text{Norm}_\psi(0, \sigma_p^2 \mathbf{I})$. Dado que la verosimilitud y el prior no son conjugados, no hay solución cerrada. Prince[56] propone utilizar lo que es conocido como la aproximación Laplaciana del posterior, tal que

$$p(\psi|\mathbf{X}, \mathbf{w}) \approx q(\psi) = \text{Norm}_\psi(\mu, \mathbf{\Sigma}). \tag{4.67}$$

Como en el caso primal, se requiere evaluar el gradiente y el Hessiano, cuyas expresiones están dadas por[56]

$$\frac{\partial L}{\partial \phi} = -\sum_{i=1}^{I} \left(\text{sig}(a_i) - w_i\right) \mathbf{X}^T \mathbf{x}_i - frac\psi\sigma_p^2, \text{ y} \tag{4.68}$$

$$\frac{\partial^2 L}{\partial \phi^2} = -\sum_{i=1}^{I} \left(\text{sig}(a_i)\right) \left(1 - \text{sig}(a_i)\right) \mathbf{X}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{X} - \frac{1}{\sigma_p^2}. \tag{4.69}$$

Con ello, se tiene que

$$\mu = \hat{\psi} \text{ y } \mathbf{\Sigma} = -\left(\frac{\partial^2 L}{\partial \psi^2}\right)^{-1}\Bigg|_{\psi=\hat{\psi}}. \tag{4.70}$$

Para inferir la etiqueta de una nueva observación se sigue un procedimiento similar al de la formulación primal en donde evaluamos la marginal con respecto a los parámetros ψ, tal que

$$\begin{aligned} p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) &= \int p(w^*|a) p(a) da, \\ &\approx \frac{1}{1 + \exp\left(-\mu_a / \sqrt{1 + \phi\sigma_a^2/8}\right)}, \end{aligned} \tag{4.71}$$

con $\mu_a = \mu^T \mathbf{X}^T \mathbf{x}^*$ y $\sigma_a^2 = \mathbf{x}^{*T} \mathbf{X} \mathbf{\Sigma} \mathbf{X}^T \mathbf{x}^*$.

4.7. Regresión con Kernels

La formulación dual permite la expresión de la regresión lógistica no lineal en términos de productos punto. Esto nos permite introducir de forma natural los kernels, toda vez que estos representan la operación

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{z}_i^T \mathbf{z}_j, \tag{4.72}$$

donde $\mathbf{z}_k = \mathbf{f}(\mathbf{x}_k)$ es la proyección del punto $\mathbf{x}_k$ a un espacio no lineal multidimensional. De esta forma el modelo de regresión no lineal puede representarse como

$$\begin{aligned} p(w|\mathbf{X}, \psi) &= \prod_{i=1}^{I} \text{Bern}_{w_i}(\text{sig}(\psi^T \mathbf{X}^T \mathbf{x}_i)), \\ &= \prod_{i=1}^{I} \text{Bern}_{w_i}(\text{sig}(\psi^T \mathbf{K}(\mathbf{X}, \mathbf{x}_i))), \end{aligned} \tag{4.73}$$

donde $\mathbf{K}(\mathbf{X}, \mathbf{Y})$ representa la matriz de $m \times k$ resultado de proyectar a espacios multidimensionales no lineales y realizar el producto punto entre los vectores de las matrices $\mathbf{X}_{n\times m}$ y $\mathbf{Y}_{m\times k}$. De forma análoga, la solución deberá encontrarse por el método de Newton. Para ello, las expresiones del gradiente y del Hessiano se convierten en [56]

$$\frac{\partial L}{\partial \phi} = -\sum_{i=1}^{I} (\text{sig}(a_i) - w_i)\, \mathbf{K}(\mathbf{X}, \mathbf{x}_i), \text{ y} \tag{4.74}$$

$$\frac{\partial^2 L}{\partial \phi^2} = -\sum_{i=1}^{I} \text{sig}(a_i)\,(1 - \text{sig}(a_i))\, \mathbf{K}(\mathbf{X}, \mathbf{x}_i)\mathbf{K}(\mathbf{x}_i, \mathbf{X}). \tag{4.75}$$

Tal como anteriormente, las funciones que cumplan con el teorema de Mercer pueden ocuparse como kerneles.

4.8. Clasificación con Vectores Relevantes

La técnica de clasificación mediante máquinas de vectores relevantes (RVM) sumariza los conceptos de clasificación lógistica no lineal mediante el uso de kerneles. Incluye el uso de priors para los parámetros ψ que promueven soluciones dispersas, la aproximación de Laplace al posterior, el uso de variables auxiliares para los parámetros y la solución iterativa del método de Newton a la representación por kerneles de la formulación dual. Para comenzar, se identifica un prior sobre los valores de los parámetros ψ que promueva una solución dispersa. Prince[56] propone lograr esto mediante el uso de la siguiente expresión

$$p(\psi) = \prod_{i=1}^{M} \text{Stud}_{\psi_i}(0, 1, \nu). \tag{4.76}$$

donde $\text{Stud}(0, 1, \nu)$ es una función de distribución Student de media cero, covarianza uno y grados de libertad ν. Para los propósitos siguientes será útil la interpretación de la pdf

de Student como la suma del producto entre una distribución Gamma y una Normal. Lo anterior queda expresado como

$$p(\psi) = \prod_{i=1}^{I} \int \text{Norm}_{\phi}(0, 1/h_i)\text{Gam}_{h_i}(\nu/2, \nu/2)dh_i, \tag{4.77}$$

donde h_i es un hiperparámetro que se define para cada una de las observaciones. Dado que la Gaussiana es separable, podemos conjuntar los efectos de las multiplicaciones en un solo factor como

$$p(\psi) = \int \text{Norm}_{\psi}(0, \mathbf{H}^{-1}) \prod_{i=1}^{M} \text{Gam}_{h_i}(\nu/2, \nu/2)dh_i, \tag{4.78}$$

donde $\mathbf{H}$ contiene las variables $\{h_i\}_{i=1}^{I}$ en su diagonal. La característica esencial del ajuste es la asignación de un hiperparámetro a cada peso o función base de tal forma de promover una solución dispersa.

La probabilidad de observar un valor particular de $\mathbf{w}$ puede evaluarse mediante la siguiente marginal

$$\begin{aligned} p(\mathbf{w}|\mathbf{X}) &= \int p(\mathbf{w}|\mathbf{X}, \psi)p(\psi)d\psi, \\ &= \int \prod_{i=1}^{I} \text{Bern}_{w_i}(\text{sig}(\psi^T \mathbf{K}(\mathbf{X}, \mathbf{x}_i)) \int \text{Norm}_{\phi}(0, \mathbf{H}^{-1}) \prod_{i=1}^{I} \text{Gam}_{h_i}(\nu/2, \nu/2)d\mathbf{H}d\psi, \\ &= \int \int \prod_{i=1}^{I} \text{Bern}_{w_i}(\text{sig}(\psi^T \mathbf{K}(\mathbf{X}, \mathbf{x}_i))\text{Norm}_{\phi}(0, \mathbf{H}^{-1}) \prod_{i=1}^{I} \text{Gam}_{h_i}(\nu/2, \nu/2)d\mathbf{H}d\psi. \end{aligned} \tag{4.79}$$

Tal como anteriormente, los primeros dos términos son aproximados por una normal con media μ y covarianza $\mathbf{\Sigma}$ de la aproximación de Laplace. Para los dos términos de la expresión anterior esto resulta en

$$\begin{aligned} \int q(\psi)d\psi \approx q(\mu) \int \exp\left(-\frac{1}{2}(\psi - \mu)^T \mathbf{\Sigma}^{-1}(\psi - \mu)\right) d\psi, \\ = q(\mu)(2\phi)^{D/2} \|\mathbf{\Sigma}\|^{1/2}. \end{aligned} \tag{4.80}$$

Con esta operación (4.79) se convierte en

$$p(\mathbf{w}|\mathbf{X}) = \int \prod_{i=1}^{I} (2\phi)^{D/2} \|\mathbf{\Sigma}\|^{1/2} \text{Bern}_{w_i}(\text{sig}(\mu^T \mathbf{K}(\mathbf{X}, \mathbf{x}_i))\text{Norm}_{\mu}(0, \mathbf{H}^{-1})\text{Gam}_{h_i}(\nu/2, \nu/2)d\mathbf{H}. \tag{4.81}$$

La expresión anterior es difícil de evaluar. Sin embargo, una buena aproximación es obtener su valor máximo. Es decir, se busca la aproximación

$$p(\mathbf{w}|\mathbf{X}) = \max_{\mathbf{H}} \left(\prod_{i=1}^{I} (2\phi)^{D/2} \|\mathbf{\Sigma}\|^{1/2} \text{Bern}_{w_i}(\text{sig}(\mu^T \mathbf{K}(\mathbf{X}, \mathbf{x}_i))\text{Norm}_{\mu}(0, \mathbf{H}^{-1})\text{Gam}_{h_i}(\nu/2, \nu/2) \right). \tag{4.82}$$

Dado que el término $\mathbf{H}$ no se encuentra en la distribución Gam, la función de verosimilitud puede ser

$$L = \sum_{i=1}^{I} \ln\left(\text{Bern}_{w_i}(\text{sig}(\mu^T \mathbf{K}(\mathbf{X}, \mathbf{x}_i))\right) + \ln\left(\text{Norm}_{\mu}(0, \mathbf{H}^{-1})\right). \tag{4.83}$$

Para obtener el valor de ψ por el método de Newton se obtienen las expresiones para el gradiente y el Hessiano, las cuales están dadas por[56]

$$\frac{\partial L}{\partial \psi} = -\sum_{i=1}^{I} \left(\text{sig}(a_i) - w_i\right) \mathbf{K}(\mathbf{X}, \mathbf{x}_i) - \mathbf{H}\psi, \text{ y} \tag{4.84}$$

$$\frac{\partial^2 L}{\partial \psi^2} = -\sum_{i=1}^{I} \text{sig}(a_i)\left(1 - \text{sig}(a_i)\right) \mathbf{K}(\mathbf{x}_i, \mathbf{X}) - \mathbf{H}, \tag{4.85}$$

donde se tiene que

$$\mu = \hat{\psi} \text{ y } \mathbf{\Sigma} = -\left(\frac{\partial^2 L}{\partial \psi^2}\right)^{-1}\Bigg|_{\psi=\hat{\psi}}. \tag{4.86}$$

En forma similar al Algoritmo 3, las variables auxiliares se obtienen de forma iterativa usando la expresión

$$h_i = \frac{(1 - h_i \Sigma_{i,i} + \nu)}{\mu_i^2 + \nu}. \tag{4.87}$$

Finalmente, para hacer inferencia se resuelve la expresión la aproximación[56]

$$\begin{aligned} p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) &= \int p(w^*|a)p(a)da, \\ &\approx \frac{1}{1 + \exp\left(-\mu_a / \sqrt{1 + \phi\sigma_a^2/8}\right)}, \end{aligned} \tag{4.88}$$

con $\mu_a = \mu^T \mathbf{K}(\mathbf{X}, \mathbf{x}^*)$ y $\sigma_a^2 = \mathbf{K}(\mathbf{x}^*, \mathbf{X})\mathbf{\Sigma}\mathbf{K}(\mathbf{X}, \mathbf{x}^*)$.

4.9. Evaluación del Desempeño

Para poder clasificar un pixel requerimos evaluar la expresión (4.43) para valores de $\mathbf{x}$. Enseguida, tenemos que establecer un criterio de clasificación τ tal que la siguiente decisión pueda ser tomada:

$$\mathcal{L}(\mathbf{x}) = \begin{cases} \text{cielo} & p(w_1|\mathbf{x}) > \tau, \\ \text{tierra} & \text{otro caso.} \end{cases} \tag{4.89}$$

Cada vez que se tomamos una decisión con respecto a una clase w_1 puede ocurrir lo siguiente. Puede ser que asignemos el pixel $\mathbf{x}$ a la clase w_1 y que por otro lado el pixel sea efectivamente de la clase w_1. En ese caso estamos ante la presencia de un verdadero positivo. Por otro lado, puede ser que asignemos el pixel $\mathbf{x}$ a la clase w_1 y pero que el pixel no pertenezca a la clase w_1. En ese caso estamos ante la presencia de un falso positivo. Inversamente, puede ser que no

Cuadro 4.1: Matriz de Confusión. La matriz de confusión sumariza los posibles resultados de tomar una decisión sobre la pertenencia de un elemento $\mathbf{x}$ a una clase w. Las hileras corresponden a la decisón que tomamos. Las columnas corresponden a la verdadera clase de $\mathbf{x}$.

	x es tipo w	**x es tipo** w
x es tipo w	Verdadero Positivo (TP)	Falso Positivo (FP)
x no es tipo w	Falso Negativo (FN)	Verdadero Negativo (TN)

asignemos el pixel $\mathbf{x}$ a la clase w_1 cuando el pixel sea la clase w_1. En ese caso tenemos un falso negativo. Finalmente, puede ser que no asignemos el pixel $\mathbf{x}$ a la clase w_1 y efectivamente ese pixel no pertenezca a la clase w_1. En ese caso estamos ante la presencia de un verdadero negativo. Todos estos resultados se sumarizan en la llamada *matriz de confusión*, que se sumariza en la Tabla 4.1.

La sumarización del desempeño para diversos valores $\mathbf{x}_i$ puede ayudarnos a determinar el desempeño de la estrategia de clasificación que hemos desarrollado. Un ejemplo clásico de esta sumarización es el análisis ROC (*Receiver Operating Characteristic*)[21]. En el análisis ROC usamos dos medidas, la tasa de verdaderos positivos (TPR) y la tasa de falsos positivos (TPR). Ambas cantidades se definen como

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \text{ y } \text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{4.90}$$

Luego la variación del criterio τ proporciona diversos valores cuyo resultado es la curva ROC. La Figura 4.7 muestra el resultado para el ejemplo de separación entre cielo y tierra desarrollado. Una medida estándar de desempeño es el área bajo la curva (AUC). En el caso de este ejmplo el AUC es 0.97. La línea en la diagonal de la curva ROC corresponde al nivel de desempeño cuando se toman decisiones aleatorias. La curva ROC es monotónicamente creciente. Por ejemplo, para un valor FPR de 0.08, el valor correspondiente de TPR es 0.91. Esto ocurre cuando el valor τ es 0.6.

La Figura 4.8 presenta un ejemplo de utilizar el clasificador en una imagen del horizonte. Primero se presenta la imagen a la cual se le ha aplicado un filtro bilateral. Esta imagen es representada en el espacio de color CIELab con la finalidad de obtener sus componentes L y b. Enseguida se presenta el resultado de aplicar la función de evaluación (4.43). Luego se presenta el resultado de tomar la decisión sobre el valor de umbral $\tau = 0.6$. Cualitativamente y tomando como base la clase que describe al cielo, se observa una región de falsos negativos en la esquina superior izquierda de la imagen.

4.10. El Viaje del Titanic

La noche del 15 de Abril del 1912, el Titanic chocó con un iceberg en su viaje inaugural entre Southampton y Nueva York[25]. Del total de 2,224 pasajeros y tripulación, más de

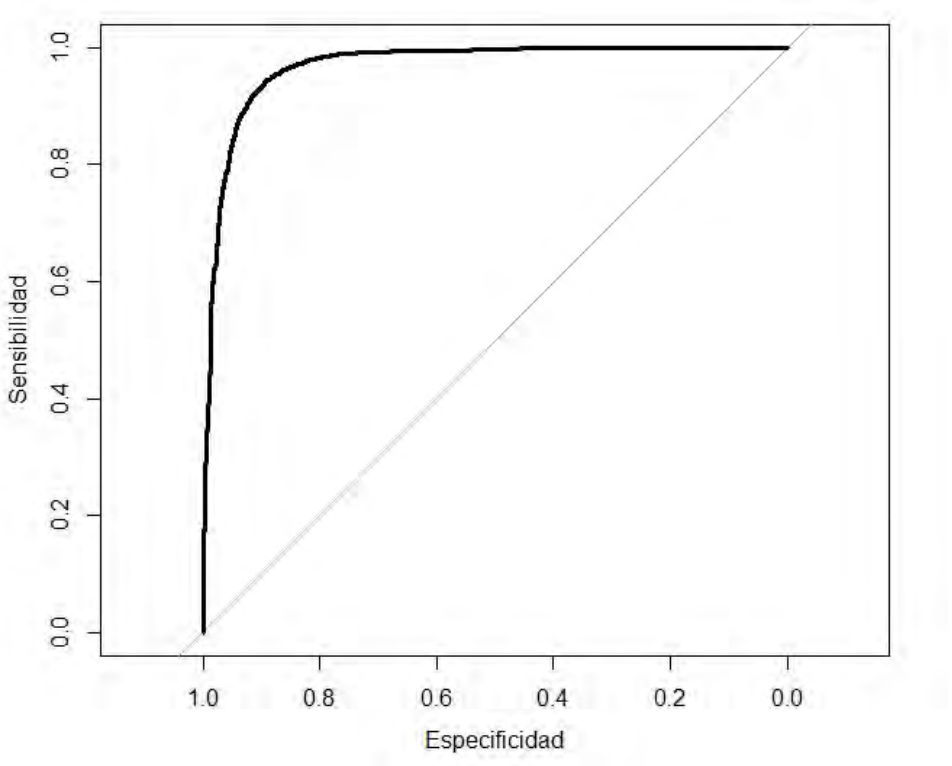


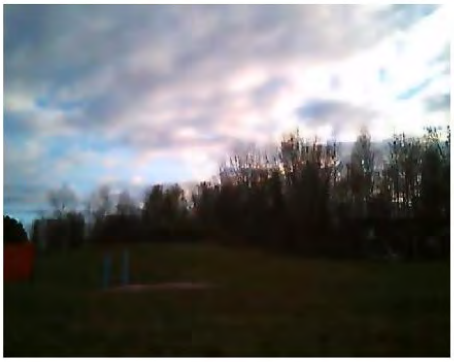
Figura 4.7: Análisis ROC. La curva ROC muestra como varían los TP en función de los FP. Una medida de desempeño estándar es el AUC, el cual es 0.97 en este caso.

1,500 personas murieron. El sitio `Kaggle`[1] pone disponible, una base de datos que contiene información sobre 852 pasajeros del Titanic. Esta información incluye si el pasajero sobrevivió o no (0 = no, 1 = sí), la clase en la que viajaba (1 = 1era; 2 = 2nda; 3 = 3ra), su nombre, su sexo, su edad, el número familiares a su mismo nivel generacional (hermanos o esposos), el número de familiares una generación anterior o una posterior (padres e hijos), el número de boleto, el monto del pago por su pasaje, el número de cabina y el puerto de embarcación (C = Cherbourg; Q = Queenstown; S = Southampton). El problema es predecir si un pasajero dado sobrevivió o no al hundimiento.

4.10.1. Visualización de los Datos

Una parte importante en el análisis de los datos es la comprensión del problema. Hacia ello, un paso adelante lo constituye su visualización. Para el caso del problema planteado, la Figura 4.11 ilustra las funciones de probabilidad de diversos predictores dado que utilizamos la información sobre la sobrevivencia del pasajero. Vale la pena hacer algunas observaciones. Por ejemplo, es notable la diferencia entre la sobrevivencia entre pasajeros de primera o tercera clase. Claramente, pasajeros de primera clase tenían una probabilidad más alta de sobrevivir que no hacerlo. Lo contrario aplicaba para pasajeros de tercera clase. Por otro lado,

[1] `https://www.kaggle.com/c/titanic/data`



(a)

(b)

(c)

Figura 4.8: Ejemplo de aplicación del clasificador lineal binario discriminativo a las imágenes del horizonte. En (a) se presenta la imagen producto del filtro bilateral. En (b) se muestra el resultado de evaluar la función discriminativa correspondiente al sigmoide en (4.43. Luego en (c) se presenta el resultado de aplicar el umbral de binarización con $p(w = cielo|\mathbf{x}) > 0.6$.

también es notable que siendo mujer se tenían una más alta probabilidad de sobrevivir que no. También se puede decir que en general para los niños menores de ocho años era más probable sobrevivir que no hacerlo. Mientras que para casi todos los grupos de edad, era más probable no sobrevivir, con la notable excepción de un grupo alrededor de los 34 años. Es interesante también notar que para las personas que viajaban solas era más probable no sobrevivir. Por otro lado, las personas que viajaban con una persona de su mismo nivel generacional (como un esposo o esposa, o hermano o hermana), las probabilidades de sobrevivir eran más altas que no hacerlo. Igualmente, era más probable no sobrevivir si se viajaba solo que con uno o dos familares de diferente nivel generacional, tales como padres o hijos. Tal vez poco sorprendente también es confirmar que el precio del boleto estaba relacionado con la probabilidad de sobrevivir. En general, entre más se pagaba por el boleto parece ser que la probabilidad de sobrevivir era más alta que si se tenía un boleto de precio bajo. Tal vez más sorprendente es confirmar que el valor del precio del boleto para el que se podía establecer esta generalización no era tan alto. Finalmente, puede observarse que si se salía

del puerto de Cherbourg, las probabilidades eran más altas de sobrevivir que no hacerlo. Mientras que lo contrario era si el puerto de salida era Southampton. Tal vez esto tenía que ver con las cabinas que los pasajeros ocupaban al ser los últimos que abordaron. Es decir, uno podría decir que las personas que tenían más probabilidades de sobrevivir eran las niñas que viajaban en primera clase que viajaban con uno de sus hermanos y uno de sus padres, habían pagado un alto precio precio por su boleto. Mientras que las mayores probabilidades de no sobrevivir se asociaban a ser un hombre que viajaba solo en tercera clase, había pagado muy poco por su boleto y había abordado en el puerto de Southampton.

4.10.2. Inferencia sobre la Sobrevivencia

Así planeado, nuestro problema tiene dos clases: sobrevivió y no-sobrevivió. Aquí exploramos la idea de construir un hiperplano que distinga entre ambas clases. Para leer los datos se puede emplear la instrucción

```
trainSet <- read.csv('train.csv', header=TRUE)
nrows = nrow(trainSet)
```

En términos del problema se tiene un conjunto de datos de entrenamiento y otro de prueba. En la práctica, necesitamos un conjunto de validación. El conjunto de prueba será empleado una sola vez. Para ello, los datos publicados por `Kaggle` como datos de entrenamiento son divididos en dos, uno que en realidad utilizaremos para realizar el entrenamiento y otro que será empleado para estimar el nivel de desempeño. El primero continuará siendo llamado como conjunto de entrenamiento y el segundo será llamado conjunto de validación. Primero calculamos el 70 % del número de registros en nuestra base de datos

```
smp_size <- floor(0.70 * nrows)
```

Enseguida seleccionamos ese número de índices de forma aleatoria de nuestra base de datos

```
train_ind <- sample(seq_len(nrows), size = smp_size)
```

Del número original de datos de entrenamiento, se toma el 70 % para entrenar y el resto para validar, tal como

```
train <- trainSet[train_ind, ]
validation <- trainSet[-train_ind, ]
```

A continuación definimos el clasificador de regresión lineal basado en los campos asociados a la clase donde se viajaba y el sexo del pasajero. Esto puede escribirse en `R` como

```
fit <- vglm(Survived ~  Pclass + Sex,
       family=multinomial, data=train)
```

Para comprobar este resultado uno puede verificar con el conjunto de validación. La instrucción podría ser

```
attach(validation)
probabilities <- predictvglm(fit,
      data.frame(Pclass = Pclass,Sex = Sex), type="response"))
```

Finalmente, la curva ROC de desempeño, incluyendo el área bajo la curva, podría obtenerse mediante el siguiente código

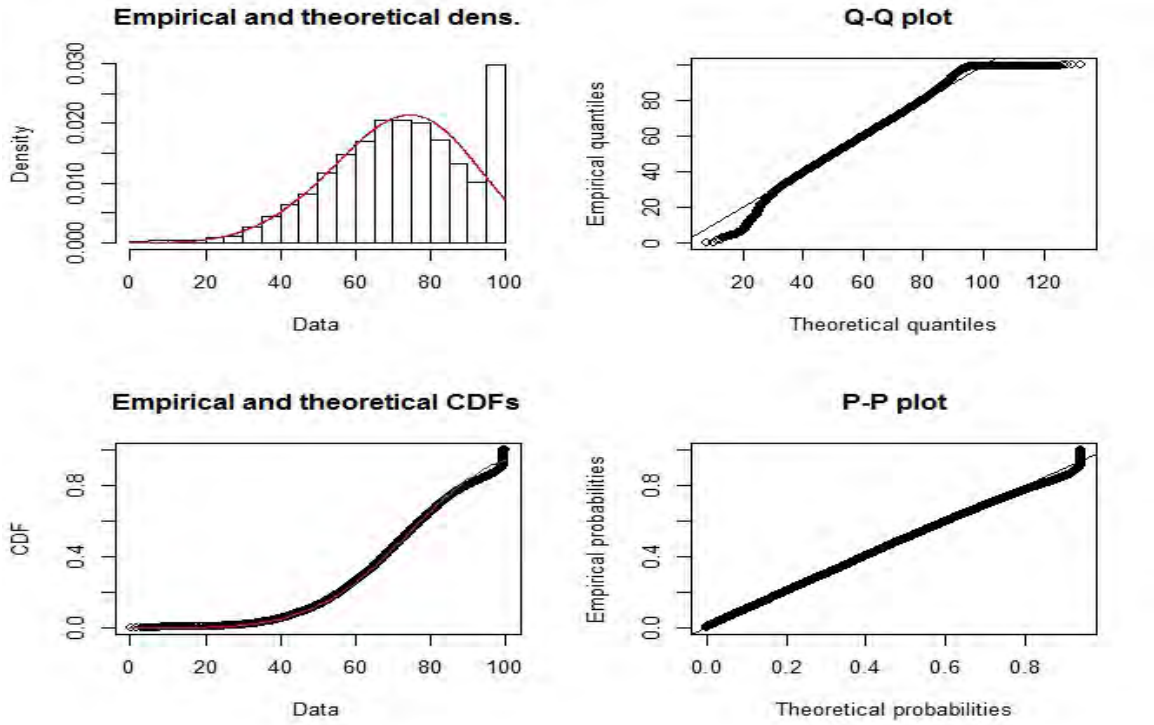
```
plt <- plot.roc(validation$Survived,probabilities[,1],
           xlab="Especificidad", ylab="Sensibilidad", lwd=3)
auc(plt)
```

El resultado de la curva ROC se muestra en la Figure 4.12. Para este ejemplo, el área bajo la curva es 0.86.

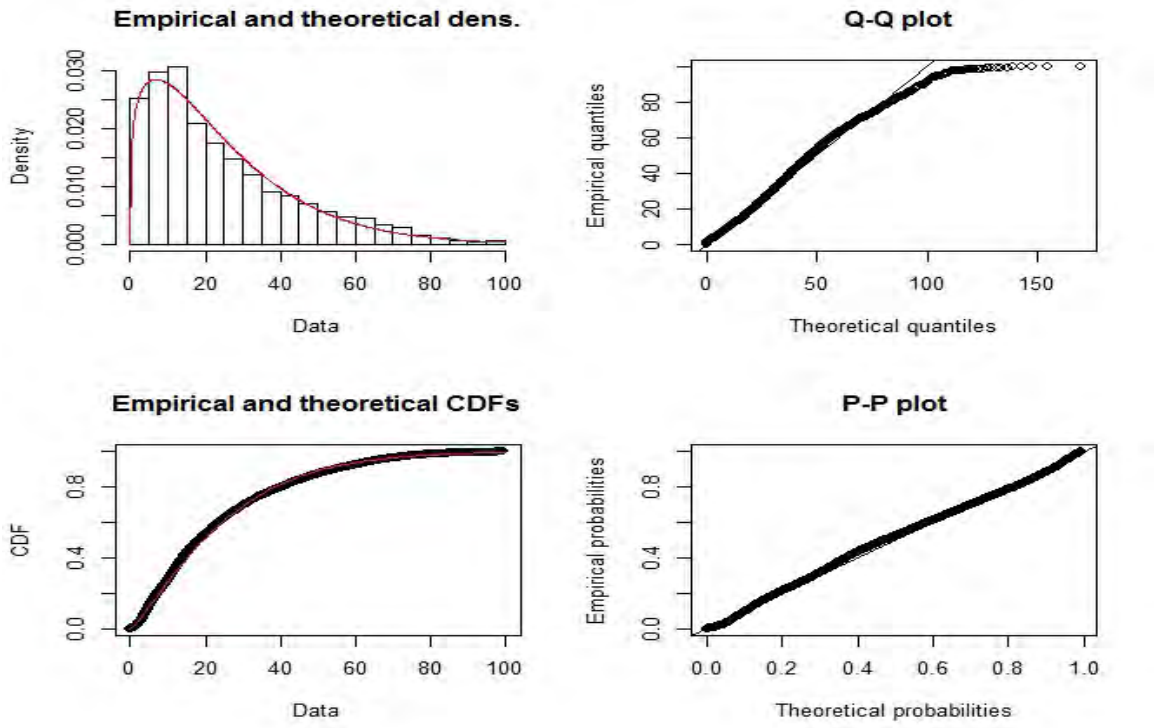
Ejercicios

Para la base de datos del Titanic,

1. Identifique las entradas de la tabla de confusión para un valor de umbral dado.
2. ¿Cuáles serán los campos que pueden ser utilizados para maximizar el desempeño del clasificador lineal?

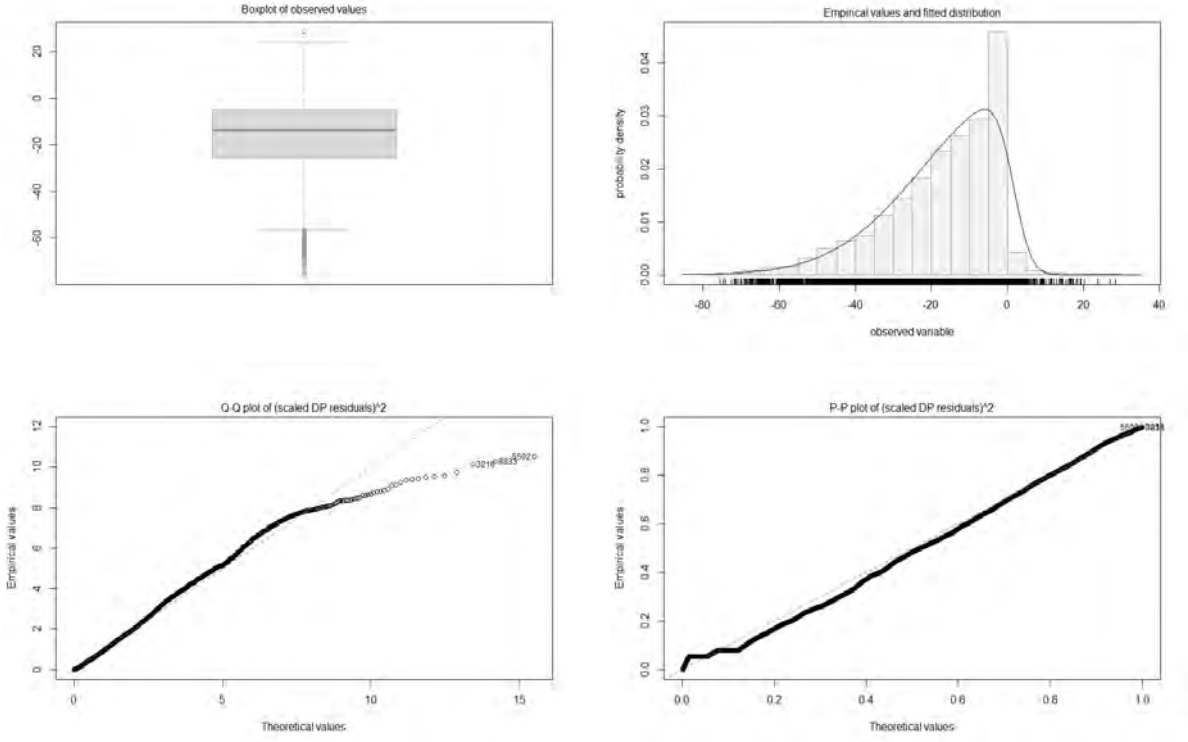


(a) Clase Cielo

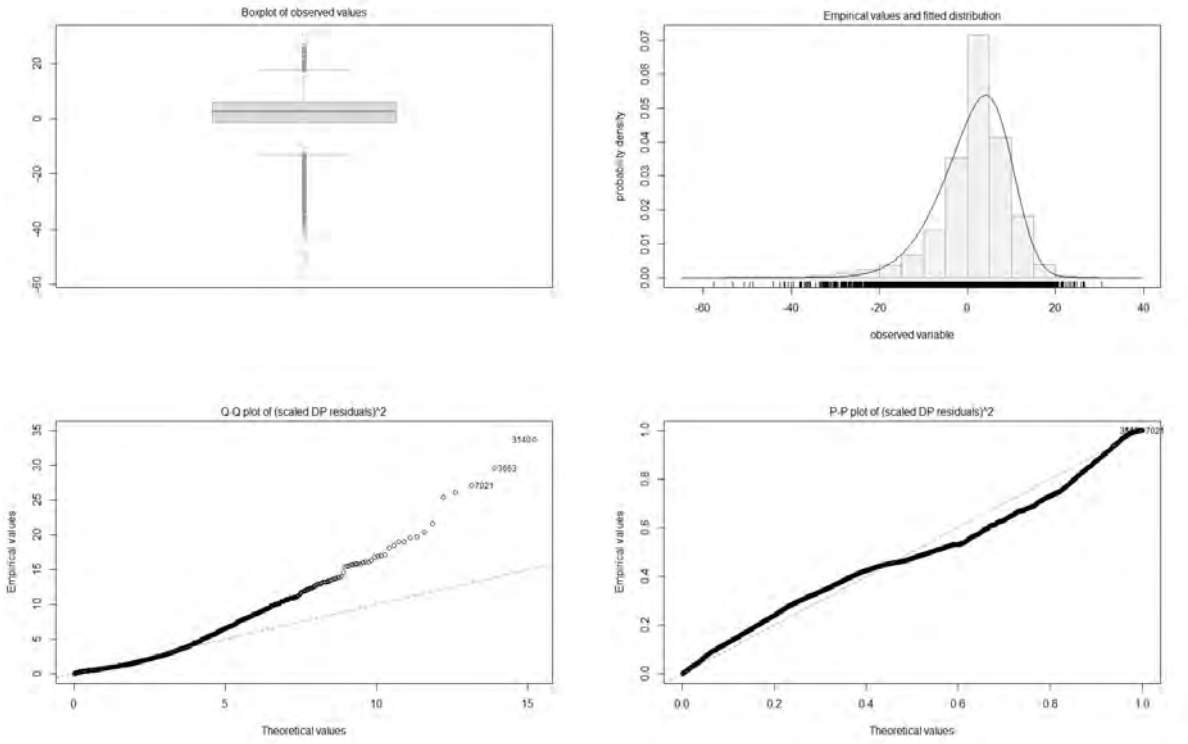


(b) Clase Tierra

Figura 4.9: Ajuste del componente L de la distribución de color $p(L, b)$ para la detección de las clases Cielo y Tierra.

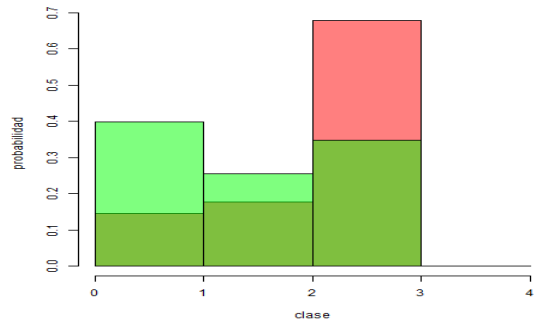


(a) Clase Cielo

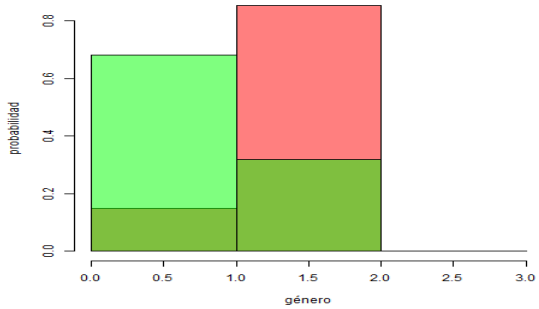


(b) Clase Tierra

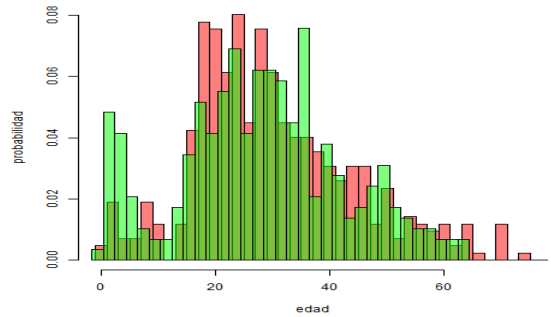
Figura 4.10: Ajuste del componente b de la distribución de color $p(L, b)$ para la detección de las clases Cielo y Tierra.



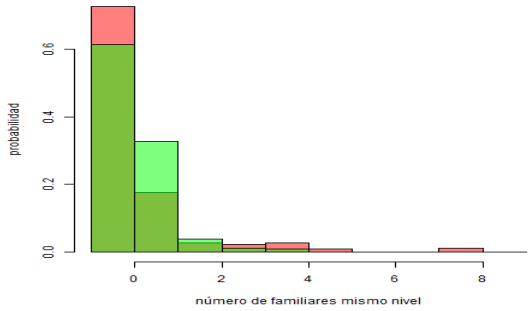
(a) Clase



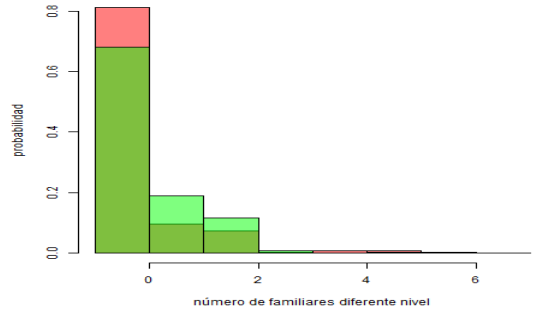
(b) Género



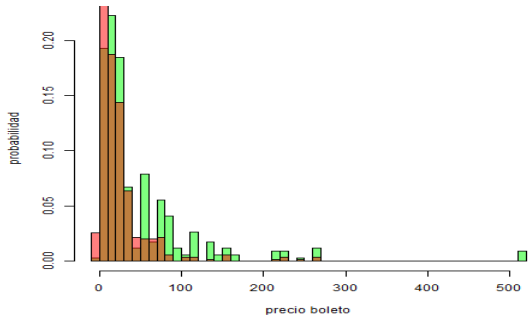
(c) Edad



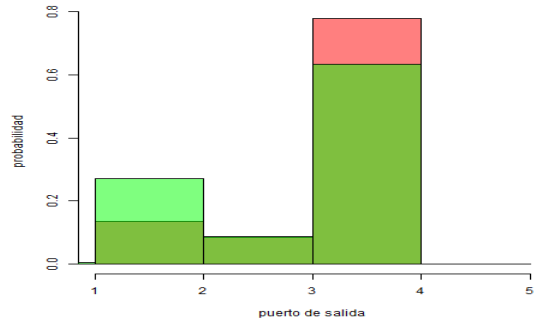
(d) Familiares del mismo nivel



(e) Familiares de diferente nivel



(f) Precio pagado por el boleto



(g) Puerto de salida

Figura 4.11: Función de distribución de probabilidad de diversas variables dada la condición de sobrevivencia (verde) o no-sobrevivencia (rojo).

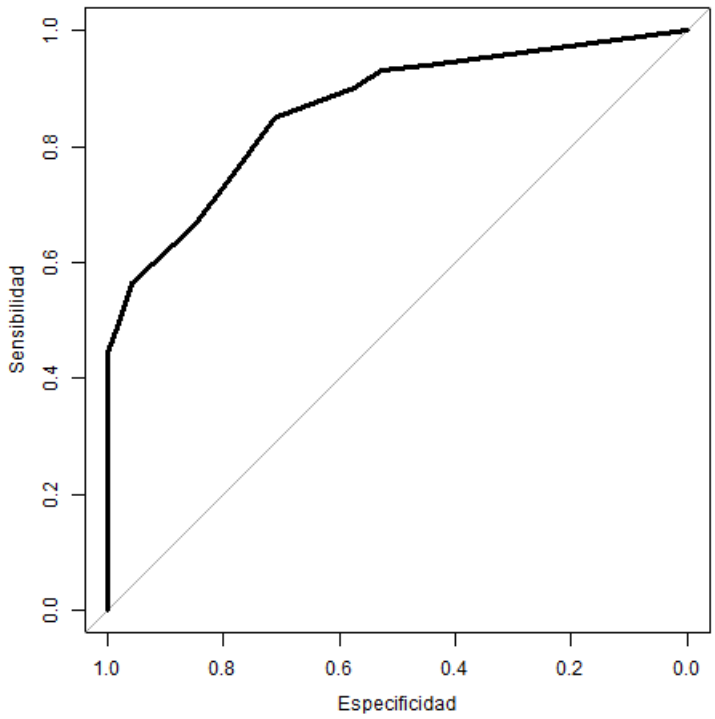


Figura 4.12: Curva ROC para el clasificador lineal. El área bajo de la curva para el clasificador lineal construido sobre los campos de la clase de viaje y el sexo del pasajero es de 0.86.

CAPÍTULO 5

Clasificación no Probabilística

En las siguientes secciones abundamos sobre enfoques cuya aproximación principal no es la obtención de una función de distribución de probabilidad.

5.1. Boosting

Una forma de realizar clasificación no lineal consiste en ir definiendo una estructura cada vez más sofisticada, incluyendo un parámetro a la vez. Es decir, primero se escoge la característica que mejor discrimina entre las clases. Enseguida, se entra a un ciclo en donde se considera este parámetro anterior fijo y se agrega uno nuevo siempre y cuando mejore la función objetivo. Por ejemplo, supóngase que se tiene la pdf

$$p(\mathbf{w}_i|\mathbf{x}_i) = \text{Bern}(\text{sig}(a_i)), \tag{5.1}$$

donde $a_i = \phi^T \mathbf{z}_i = \phi^T \mathbf{f}(\mathbf{x}_i)$ es el término de activación. Para la explicación que viene vamos a considerar la notación expandida de a_i como

$$a_i = \phi_0 + \sum_{k=1}^{K} \phi_k \mathbf{f}(\mathbf{x}_i, \xi_k), \tag{5.2}$$

para un conjunto de parámetros ξ_k. Algunas funciones que comúnmente se utilizan incluyen

- $\mathbf{f}(\mathbf{x}, \xi) = \arctan(\alpha^T \mathbf{x})$, para parámetros $\alpha = \xi$, ó
- $\mathbf{f}(\mathbf{x}, \xi) = \exp(-1/2(\mathbf{x} - \alpha)^T(\mathbf{x} - \alpha)/\lambda^2)$, para parámetros $\xi = (\alpha^T, \lambda)^T$).

El procedimiento de aprendizaje incremental procede como sigue. Primero se define un término de activación con una variable, tal como

$$a_i = \phi_0 + \phi_1 \mathbf{f}(\mathbf{x}_i, \xi_1), \tag{5.3}$$

con la variable ϕ_1 que mejor ayuda a distinguir las clases. Los parámetros ϕ_0, ϕ_1 y ξ_1 pueden ser aprendidos mediante ML. Enseguida se agrega un segundo conjunto de variables ϕ_2 y ξ_2, de tal forma que se crea el término de activación

$$a_i = \phi_0 + \phi_1 \mathbf{f}(\mathbf{x}_i, \xi_1) + \phi_2 \mathbf{f}(\mathbf{x}_i, \xi_2), \tag{5.4}$$

dejando fijo tanto ϕ_1 como ξ_1. Nuevamente los términos ϕ_0, ϕ_2 y ξ_2 se aprenden mediante ML. Esto se sigue mientras las variables que se incluyen agregen valor al clasificador. En particular para el término k-ésimo se tiene la función de activación

$$a_i = \phi_0 + \sum_{k=1}^{K} \phi_k \mathbf{f}(\mathbf{x}_i, \xi_k), \tag{5.5}$$

donde se aprenden los términos ϕ_0, ϕ_k y ξ_k y los demás se mantienen fijos.

Un caso particular de esta estrategia incremental se da al usar la función escalón, Escalón, toma el lugar de la función $\mathbf{f}$. La función Escalón(x) de define como uno para valores de x mayores o iguales a cero y cero para todos los demás. Una función tan sencilla da para un clasificador débil, uno que permite la distinción entre clases de tal forma que lo que más esperamos de él es que a largo plazo su desempeño sea mejor que una decisión tomada aleatoriamente. Se procura que los clasificadores débiles sean buenos al menos para decir lo que no es, dejando abierta la posibilidad a tener muchos falsos positivos. La cuestión es que al concatenar muchos clasificadores débiles, cada uno desechando lo que no es, se llega a la construcción de un clasificador fuerte. Esta estrategia es conocida como *boosting*.

Los parámetros se obtienen de forma iterativa mediante el método de Newton (4.48), lo cual requiere la evaluación del gradiente (4.47) y el Hessiano (4.49). Estas últimas dos expresiones dejan claro que la importancia de cada cambia dependiendo del resultado de la predicción. Los clasificadores posteriores se enfocan en las partes más difíciles del problema, aquellos que no fueron bien clasificados por otros antes.

5.2. Arboles de Decisión

Un grafo es un par ordenado $G = (V, E)$, donde V es un conjunto de vértices o nodos y E es un conjunto de aristas o arcos, los cuales en sí son pares de elementos de V[69]. Por ejemplo, un grafo $G = (\{1, 2, 3\}, \{(1, 2), (1, 3)\})$ es un grafo con tres nodos y dos aristas, una que va del primero al segundo nodo y otra que va del primero al tercero. Uno puede clasificar los grafos entre dirigidos y no dirigidos. Un grafo dirigido específica una dirección para el arco o arista. En el ejemplo anterior, las aristas estaban dirigidas de un origen a un destino. Cuando todas las aristas son bidireccionales se dice que el grafo es no dirigido. Esto establece una distinción importante que se reflejada en temas como tan dispersos como restricciones a los modelos o requerimientos de memoria para almacenamiento. Por ejemplo, si representaramos el grafo G como una matriz donde las hileras representan el nodo de origen y las columnas representan el nodo destino, un grafo no dirigido estaría representado por una matriz simétrica.

Un árbol es una grafo no dirigido en el cual los vértices están conectados por solo un sendero. Por ejemplo, el grafo G definido con anterioridad es un caso de un árbol. Un árbol de decisión modela las consecuencias de las decisiones como rutas que resultarían de ir en uno u otro camino (ver Figura 5.1). Los árboles de decisión crean superficies de clasificación no lineales al particionar el espacio de los datos en regiones distintas . Para la toma de una decisión, se puede definir una función de la forma[56]

$$a = (1 - g_j(\mathbf{x}, \gamma))\phi_0^T \mathbf{x} + g_j(\mathbf{x}, \gamma)\phi_1^T \mathbf{x}, \tag{5.6}$$

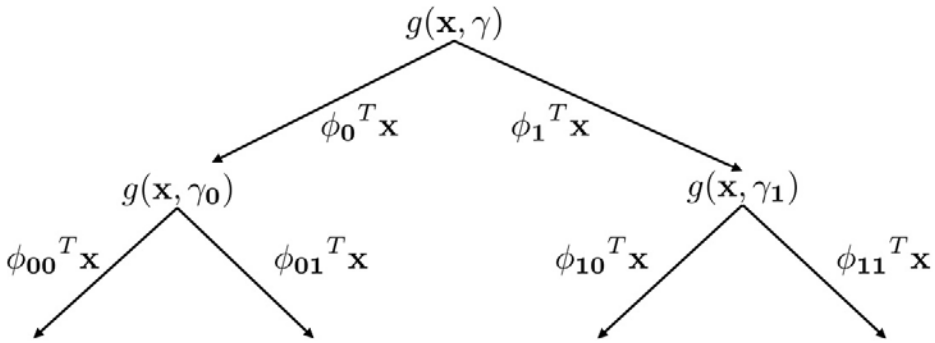


Figura 5.1: Arbol de decisión. El espacio es dividido por las superficies de decisión. La superficie que se escoge es una combinación lineal determinada por las funciones de activación.

donde $\mathbf{x}$ es una variable conteniendo las características, γ es un umbral para la función $g_j(\mathbf{x}, \gamma)$. Si el valor es uno, se escoge el modelo $\phi_1^T\mathbf{x}$; si el valor es cero, se escoge el modelo $\phi_0^T\mathbf{x}$; si el valor se encuentra entre cero y uno, el modelo toma la forma de una combinación entre ellos. Aun cuando formas más elaboradas pueden ser utilizadas, una expresión común corresponde al escalón unitario, definido como

$$g_j(\mathbf{x}, \gamma) = \begin{cases} 1 & \text{si } x_j > \gamma, \\ 0 & \text{en otro caso.} \end{cases} \tag{5.7}$$

y los valores de γ, ϕ_0, ϕ_1 pueden ser aprendidos por máxima verosimilitud (ML). Una forma de crear una estructura de árbol es anidando términos para la función de activación. Así, por ejemplo, la estructura

$$\begin{aligned} a_i &= (1 - g_j(\mathbf{x}, \gamma_i))(\phi_0^T\mathbf{x} + a_{i0}) + g_j(\mathbf{x}, \gamma_i)(\phi_1^T\mathbf{x} + a_{i1}), \text{ con} \\ a_{i0} &= (1 - g_k(\mathbf{x}, \gamma_{i0}))\phi_{00}^T\mathbf{x} + g_k(\mathbf{x}, \gamma_{i0})\phi_{01}^T\mathbf{x}, \text{ y} \\ a_{i1} &= (1 - g_l(\mathbf{x}, \gamma_{i1}))\phi_{10}^T\mathbf{x} + g_l(\mathbf{x}, \gamma_{i1})\phi_{11}^T\mathbf{x}, \end{aligned} \tag{5.8}$$

puede ser representada por la Figura 5.1.

En la práctica, se crean un gran número de árboles para separar los datos de forma lineal o no lineal. La idea es que durante la operación, la variabilidad se disminuye al promediar entre el número de árboles. Otra ventaja que ofrecen los árboles de decisión es que naturalmente se adaptan a datos faltantes, al no asignar una rama hacia superficies en donde no hay datos. Las múltiples divisiones terminan dividiendo el espacio de búsqueda en regiones. Para cada una de ellas se asigna un valor constante[20]. Para problemas de clasificación este valor corresponde a la clase más probable. En regresión, este valor corresponde al promedio de las observaciones en la región.

5.2.1. Clasificador basado en Arboles de Regresión

Ahora ilustramos el funcionamiento de los árboles de decisión para la base de datos de los pasajeros del Titanic[1], en particular se consideran la aproximación mediante *boosting*.

[1]Este problema y los datos asociados son descritos en `kaggle.com`

En la aproximación, se considera la característica que mejora más la función objetivo. Esta se toma y se considera fija en la elección de la siguiente característica a considerar. En adición, las características que se consideran están basadas en la función escalón. Después de cada elección el peso de las características cambia dependiendo si ocurrió o no una buena clasificadas. Con ello, el siguiente clasificador trabaja más cercanamente con los datos que han sido mal clasificados.

Para el ejemplo que elaboramos, primero los datos son leídos

```
data = read.csv("train.csv")
```

Luego, los datos son divididos en una porción de entrenamiento y una de validación, de la siguiente manera

```
smp_size <- floor(0.70 * nrow(data)) #porcion para entrenamiento
train_ind <- sample(seq_len(nrow(data)), size = smp_size)
train <- data[train_ind, ]
test <- data[-train_ind, ]
```

Ahora, entrena el modelo de clasificación

```
features = c(3,5,6,7,8,10,12)#predictores
label = 2 #respuesta
bag.fraction = 0.5 #porcion para seleccion de variables
learning.rate = 0.005 #peso de cada arbol
tree.complexity = 5
step.size = 2 #agregar este numero de arboles por ciclo
n.trees = 100 #numero inicial de arboles
#entrena modelo
fit <- gbm.step(data=train, gbm.x = features, gbm.y = label,
      family = "bernoulli", tree.complexity = tree.complexity,
      learning.rate = learning.rate, bag.fraction = bag.fraction,
      step.size = step.size,n.trees = n.trees, verbose = FALSE)
```

Para probar el desempeño del clasificador predecimos la respuesta para el conjunto de prueba

```
preds <- predict.gbm(fit, test[,features],
     n.trees=fit$gbm.call$best.trees, type="response")
```

Enseguida hacemos un análisis ROC sobre estos resultados

```
plt <- plot.roc(test[,label],preds, xlab="Especificidad",
    ylab="Sensibilidad", lwd=3)
```

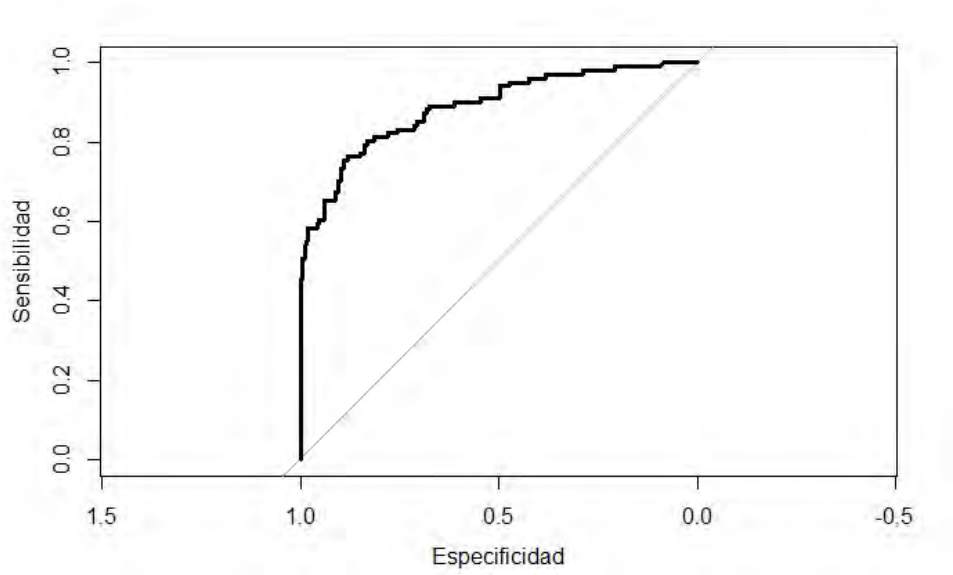


Figura 5.2: Desempeño de Arboles de Decision. Análisis ROC correspondiente a la base de datos de sobrevivencia del Titanic. El área bajo la curva es 0.89

La Figura 5.2 muestra el resultado de este análisis.

5.3. Bosques Aleatorios

La técnica de bosques aleatorios es un método que combina múltiples árboles de decisión, cada uno de los cuales incorpora muestreos aleatorios de los datos de la muestra y la selección aleatoria de las variables de discriminación en cada nodo de cada árbol, para producir decisiones sobre las clases o inferir el estado del proceso. La idea sobre la construcción de árboles donde una selección aleatoria de predictores se usaba para determinar la dirección de la inferencia fue propuesta por Ho[29]. Por su lado, Breiman[9] y Cutler[17] le dieron su forma actual introduciendo la idea del muestreo aleatorio de los datos de las observaciones.

Para los bosques aleatorios se consideran únicamente árboles binarios. En cada nodo, hay una prueba asociada $g(\mathbf{x}, \Gamma)$ que indica la decisión que se toma sobre si usar la rama izquierda o usar la rama derecha del árbol. Supóngase que se tiene $S = \{\mathbf{X}, \mathbf{w}\}$, las observaciones $\mathbf{X}$ y su etiqueta de clasificación asociada $\mathbf{w}$, donde $\mathbf{X}_{d\times n} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ es el conjunto de los datos de entrenamiento (d corresponde al número de variables y n al número de observaciones) y $\mathbf{w} = (w_1, \ldots, w_n)$ contiene la clase correspondiente a cada observación. La función $g(\mathbf{x}, \Gamma)$ divide a S en dos conjuntos, S^l y S^r, cuya unión forma el conjunto original y cuya intersección es nula.

Durante entrenamiento, se aprende la función $g(\mathbf{x}, \Gamma)$ que mejor separa S en los conjuntos S^l y S^r. En bosques aleatorios es común maximizar la ganancia de información[42], definida como

$$I = H(S) - \sum_{i\in\{l,r\}} \frac{|S^i|}{|S|} H(S^i), \tag{5.9}$$

donde $H(S)$ es la entropía de Shannon[59], la cual se define como

$$H(S) = -\sum_{i=1}^{n} p(w_i) \ln p(w_i), \tag{5.10}$$

y $p(w_i)$ es la probabilidad de la ocurrencia de la etiqueta w_i en el conjunto S. La etapa de entrenamiento termina cuando se alcanzan D niveles de profundidad en el árbol.

Cuando se presenta un nuevo dato $\mathbf{x}^*$, este viaja por todos los árboles del bosque, siendo evaluado en cada nodo de cada árbol por la respectiva función g para decidir si se toma la rama izquierda o derecha. Cuando el dato arriba a la hoja del árbol, se aplica un clasificador (o regresor) con los datos contenidos ahí. Los datos en cada hoja corresponden efectivamente a la probabilidad de un cierto estado del mundo dado el conjunto de datos, $p_t(w|\mathbf{x})$. Para tomar la decisión se toman en cuenta el resultado de todos los árboles del bosque. Es común usar el promedio, tal como

$$p(w|\mathbf{x}) = \frac{1}{T} \sum_{t} p_t(w|\mathbf{x}), \tag{5.11}$$

donde T corresponde al número de árboles del bosque. Es muy importante mencionar que los bosques aleatorios manejan de forma natural los casos en los que w toma valores múltiples, no solo binarios.

En el caso de los bosques aleatorios, la variable $\Gamma = \{\Omega, \phi, \tau)$ contiene un selector Ω de características, la primitiva geométrica para separar los datos ϕ y un conjunto de umbrales τ. Por ejemplo, la función[16]

$$g(\mathbf{x}, \Gamma) = \tau_1 < \Omega(\mathbf{x})^T \phi < \tau_2, \tag{5.12}$$

establece una separación lineal entre los datos. O por ejemplo,

$$g(\mathbf{x}, \Gamma) = \tau_1 < \Omega(\mathbf{x})^T \phi \Omega(\mathbf{x}) < \tau_2, \tag{5.13}$$

establece una separación cuadrática entre los datos.

La principal diferencia entre los árboles de regresión logística y los bosques aleatorios es que mientras que los primeros escogen aleatoriamente un subconjunto de los datos con el cual hacer el entrenamiento de cada árbol, los segundos, en adición, escogen de forma aleatoria el conjunto de parámetros con los cuales se construye el camino a seguir en los árboles. La ventaja de escoger un subconjunto de los datos es que la velocidad de entrenamiento se acelera. La ventaja de usar un subespacio del espacio de parámetros es que se trabaja con todo el conjunto de datos y se maximiza el margen entre clases cuando se toman todos los árboles del bosque.

En los bosques aleatorios hay varias propiedades a estudiar, las cuales incluyen la profundidad D de los árboles, la cantidad de parámetros que escogen aleatoriamente para la función de división, el tamaño T del bosque, el modelo de clasificador débil, la función de entrenamiento objetivo. Estos factores influyen en el resultado final del clasificador. Por ejemplo, al aumentar la profundidad de los árboles se promueve el overfitting en detrimento de la generalización. Por otro lado, la selección aleatoria de los datos o de las variables busca promover la decorrelación entre los árboles, logrando con ello también mejores propiedades de generalización.

En las versiones moderanas de bosques aleatorios[10], cada árbol es construido con un subconjunto de los datos obtenido de aleatoria, uniforme y con reemplazo (*bootstrapping* o *bagging*) del conjunto de datos originales[8]. Como consecuencia, un cierto número de datos, conocidos como *out-of-the-bag(oob)*, son dejados no son utilizados para la construcción del árbol y pueden ser usados para probar su desempeño. Para cada variable, su importancia se calcula observando la degradación contra fluctuaciones aleatorias del contenido de sus variables. Primero se mide la precisión para muestras oob. Enseguida se mide la precisión cuando se permuta el valor de las variables entre las muestras oob. La diferencia en la precisión dividido entre el número de árboles nos proporciona un *score* de importancia. La ejecución de este procedimiento durante un cierto número de ocasiones nos permiten obtener valores medios y desviaciones estándar para cada uno de los predictores. Con ellos, el *Z score* se define como el valor medio de la pérdida dividido entre la desviación estándar.

En **R** se puede utilizar la librería de nombre `randomForest` para entrenar y probar árboles aleatorios.

5.4. Máquina de Vectores de Soporte (SVM)

Dado un conjunto de observaciones y etiquetas $D = \{\mathbf{x}_i, w_i\}_{i=1}^{n}$, con $w_i = \{-1, 1\}$, la tarea de clasificación involucra la construcción de una superficie que separe ambas clases. Cuando las clases son perfectamente separables es posible que haya varias de estas superficies. La máquina de vectores de soporte (SVM) busca elegir aquella superficie cuya distancia a los puntos más cercanos a ella se maximice (ver Figura 5.3). Cuando la separación de las clases se da por una superficie no lineal, se utiliza el truco de los kerneles: las observaciones son proyectadas a espacios multidimensionales no lineales buscando que ahí la separación pueda ser mediante un hiperplano.

5.4.1. Clases Separables

En notación matemática, el problema consiste en encontrar los parámetros del hiperplano que distingue a la observación $\mathbf{x}_i$ entre las clases tal que

$$w_i = \begin{cases} 1 & \text{si } \tilde{\phi}^T\mathbf{x}_i - \tilde{\phi}_0 > 0, \text{ y} \\ -1 & \text{si } \tilde{\phi}^T\mathbf{x}_i - \tilde{\phi}_0 < 0, \forall(\mathbf{x}_i, w_i) \in D. \end{cases} \tag{5.14}$$

o de forma más compacta

$$w_i(\tilde{\phi}^T\mathbf{x}_i - \tilde{\phi}_0) > 0, \forall(\mathbf{x}_i, w_i) \in D. \tag{5.15}$$

Utilizando la transformación $\phi = \tilde{\phi}$ y $\phi_0 = \tilde{\phi}_0 - 1/w_i$, la expresión (5.15) puede convertirse en

$$w_i(\phi^T\mathbf{x}_i - \phi_0) \geq 1, \forall(\mathbf{x}_i, w_i) \in D. \tag{5.16}$$

Es decir, el problema consiste en determinar el hiperplano $F(\mathbf{x})$, definido como

$$F(\mathbf{x}) = \phi^T\mathbf{x} - \phi_0, \tag{5.17}$$

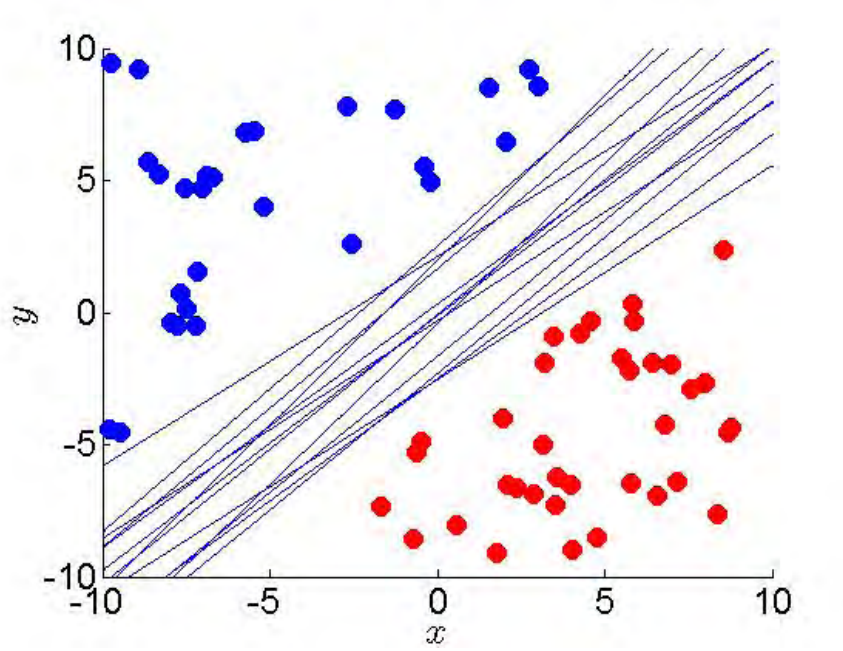


Figura 5.3: Cuando las clases son separables, hay varias superficies de discriminación que pueden ser definidas. En la máquina de vectores de soporte (SVM) se busca identificar aquella que máximice la distancia a los puntos más cercanos.

donde

$$|F(\mathbf{x})|/\|\phi\|, \tag{5.18}$$

es la distancia del punto al hiperplano. Consecuentemente, el margen, que es el factor que se busca maximizar, se encuentra a $1/\|\phi\|$ pues por construcción este se encuentra en $F(\mathbf{x}) = 1$. En ese sentido, entre menor sea $\|\phi\|$ mayor será $1/\|\phi\|$. Con ello, el problema puede refrasearse como minimizar la función $Q(\phi)$ tal que

$$\begin{gathered} \text{minimizar } Q(\phi) = \tfrac{1}{2}\|\phi\|^2, \\ \text{sujeto a } w_i(\phi^T\mathbf{x}_i - \phi_0) \geq 1, \forall(\mathbf{x}_i, w_i) \in D, \end{gathered} \tag{5.19}$$

donde los puntos para los cuales $1/\|\phi\| = 1$ son llamados vectores de soporte. Esta formulación es conocida como la formulación primal. La forma común de resolverla consiste en plantear conjuntamente los términos por optimizar y embeber la restricción usando multiplicadores de Lagrange. La formulación se puede expresar como

$$J(\phi, \phi_0, \alpha) = \frac{1}{2}\phi^T\phi - \sum_{i=1}^{n} \alpha_i(w_i(\phi^T\mathbf{x}_i - \phi_0) - 1), \tag{5.20}$$

donde $\alpha = \{\alpha_i\}_{i=1}^n$ son los operadores de Lagrange. Para obtener los valores óptimos de los parámetros, se obtiene la derivada de (5.20) con respecto ellos y se resuelve la expresión igualada a cero. En particular, las derivadas con respecto a ϕ y ϕ_0 resultan en las expresiones

$$\phi = \sum_{i=1}^{n} \alpha_i w_i \mathbf{x}_i, \text{ y } \sum_{i=1}^{n} \alpha_i w_i = 0, \tag{5.21}$$

respectivamente.

La expresión dual de (5.20) se puede obtener de la siguiente forma. Primero, la expandimos logrando la expresión

$$J(\phi, \phi_0, \alpha) = \frac{1}{2}\phi^T\phi - \sum_{i=1}^{n} \alpha_i w_i \phi^T \mathbf{x}_i - \sum_{i=1}^{n} \alpha_i w_i \phi_0 + \sum_{i=1}^{n} \alpha_i. \tag{5.22}$$

Esta ecuación puede ser transformada de la siguiente forma. Por un lado, el tercer término es igual a cero. Por su parte, el término $\phi^T\phi$ puede expresarse como

$$\phi^T\phi = \sum_{i=1}^{n} \alpha_i w_i \phi^T \mathbf{x}_i = \sum_{j=1}^{m}\sum_{i=1}^{n} \alpha_j \alpha_i w_j w_i \mathbf{x}_j^T \mathbf{x}_i, \tag{5.23}$$

lo que en el proceso nos ayuda a eliminar el término ϕ^T del segundo término. Con todos estos cambios, (5.20) puede ser expresada únicamente en términos de las observaciones tal como

$$J(\phi, \phi_0, \alpha) = Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{j=1}^{m}\sum_{i=1}^{n} \alpha_j \alpha_i w_j w_i \mathbf{x}_j^T \mathbf{x}_i, \tag{5.24}$$

y el problema de encontrar la superficie de clasificación que distinga entre dos clases puede plantearse como

$$\begin{aligned} \text{minimizar } Q(\alpha) &= \sum_{i=1}^{n} \alpha_i - \sum_{j=1}^{m}\sum_{i=1}^{n} \alpha_j \alpha_i w_j w_i \mathbf{x}^T \mathbf{x}_i, \\ \text{sujeto a} \sum_{i=1}^{m} \alpha_i w_i &= 0, \alpha_i \geq 0, \end{aligned} \tag{5.25}$$

la cual tiene la ventaja de estar expresada como productos punto. El valor de los parámetros ϕ puede ser obtenido evaluando (5.21) utilizando el valor obtenido para α.

Es interesante observar que para minimizar (5.20) se debe cumplir la condición

$$\alpha^*(w_i(\phi^{*T}\mathbf{x}_i - \phi_0) - 1) = 0. \tag{5.26}$$

En ese caso ó el operador de Lagrange α^* es cero o la restricción $(w_i(\phi^{*T}\mathbf{x}_i - \phi_0) - 1)$ es cero. Este segundo caso se da cuando $\mathbf{x}_i$ es un vector de soporte. En el resto de los casos, el operador α_i^* es igual a cero.

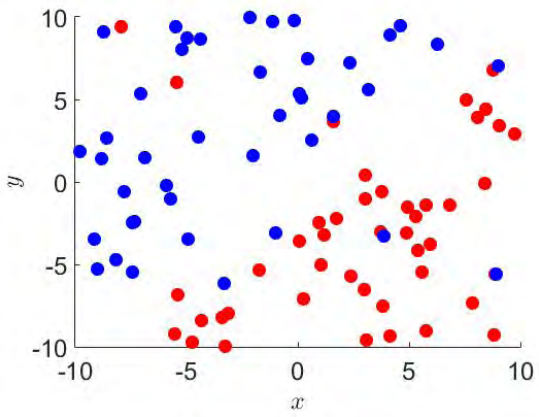
El término ϕ_0 puede ser calculado como

$$\phi_0^* = 1 - \phi^T \mathbf{x}_i, \tag{5.27}$$

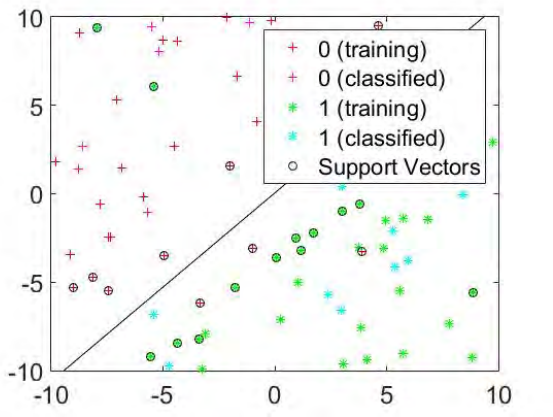
para un vector de soporte $\mathbf{x}_i$.

Finalmente, la expresión dual para la superficie de clasificación queda como

$$F(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i w_i \mathbf{x}_i^T \mathbf{x} - \phi_0. \tag{5.28}$$



(a) Conjunto de Observaciones



(b) Clasificador SVM

Figura 5.4: Se muestra la construcción de un clasificador SVM lineal cuando las clases no son separables. En (a) se ilustra el conjunto original de observaciones. En (b) se muestra el resultado de construir el clasificador. La ilustración muestra las observaciones de entrenamiento y de prueba, los vectores de soporte y la superficie de clasificación.

5.4.2. Clases no Separables

En la práctica no podemos esperar que las clases sean perfectamente separables. En el caso de SVM se toleran la clasificación erronea de ciertas observaciones mediante la introducción de variables de holgura ξ_i. Con su uso, el problema puede refrasearse como minimizar la función $Q(\phi, \phi_0, \xi)$ tal que

$$\begin{aligned} &\text{minimizar } Q(\phi, \phi, \xi) = \frac{1}{2}\|\phi\|^2 + C\sum_{i=1}^{n}\xi_i, \\ &\text{sujeto a } w_i(\phi^T\mathbf{x}_i - \phi_0) \geq 1 - \xi_i \text{ y } \xi_i \geq 0. \end{aligned} \tag{5.29}$$

De esta forma C establece un compromiso entre la minimización de la clasificación erronea y la maximización del margen.

La representación del dual tiene una pequeña, pero importante, modificación, quedando expresada como

$$\begin{aligned} \text{minimizar } Q(\alpha) &= \sum_{i=1}^{n}\alpha_i - \sum_{j=1}^{n}\sum_{i=1}^{n}\alpha_j\alpha_i w_j w_i \mathbf{x}^T\mathbf{x}_i, \\ \text{sujeto a } \sum_{i=1}^{m}\alpha_i w_i &= 0, C \geq \alpha_i \geq 0, \end{aligned} \tag{5.30}$$

y los términos para ϕ^* y $F(\mathbf{x})$ están dados por (5.21) y (5.28).

La Figura 5.4 muestra un ejemplo de clasificación mediante SVM en donde las clases no son totalmente separables por un superficie lineal.

5.4.3. Clases no Linealmente Separables

Para encontrar la superficie de discriminación entre clases separadas no linealmente, se proyectan las observaciones $\mathbf{x}$ a espacios multidimensionales no lineales $\mathbf{f}(\mathbf{x})$. La expectativa es que en el espacio de las características las clases sean linealmente separables por la superficie

$$\phi^T \mathbf{f}(\mathbf{x}) - \phi_0 = 0, \tag{5.31}$$

en donde ϕ está dado por

$$\phi = \sum_{i=1}^{n} \alpha_i w_i \mathbf{f}(\mathbf{x}_i), \tag{5.32}$$

y la función de clasificación en el espacio dual está descrita por la ecuación

$$F(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i w_i \mathbf{f}(\mathbf{x}_i)^T \mathbf{f}(\mathbf{x}) - \phi_0. \tag{5.33}$$

La forma dual del problema de clasificación puede entonces ser planteada como

$$\begin{aligned} \text{minimizar } Q(\alpha) &= \sum_{i=1}^{n} \alpha_i - \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_j \alpha_i w_j w_i \mathbf{f}(\mathbf{x}_i)^T \mathbf{f}(\mathbf{x}_i), \\ \text{sujeto a } \sum_{i=1}^{m} \alpha_i w_i &= 0, C \geq \alpha_i \geq 0, \end{aligned} \tag{5.34}$$

la cual está lista para la utiilización de kerneles de la forma

$$k(\mathbf{u}, \mathbf{v}) = \mathbf{f}(\mathbf{u})^T \mathbf{f}(\mathbf{v}), \tag{5.35}$$

provisto que k satisface la condición de Mercer, ser una función positiva semi-definida. El problema dual puede entonces ser representado utilizando kerneles como

$$\begin{aligned} \text{minimizar } Q(\alpha) &= \sum_{i=1}^{n} \alpha_i - \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_j \alpha_i w_j w_i k(\mathbf{x}_i, \mathbf{x}_j), \\ \text{sujeto a} \sum_{i=1}^{n} \alpha_i w_i &= 0, C \geq \alpha_i \geq 0, \end{aligned} \tag{5.36}$$

y la función de clasificación se define como

$$F(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i w_i k(\mathbf{x}_i, \mathbf{x}) - \phi_0. \tag{5.37}$$

Tal como hemos explorado, hay diversas funciones que pueden ocupar el lugar de $k(\mathbf{u}, \mathbf{v})$. Entre ellas, algunas de las más populares incluyen la lineal, la polinomial, la función de base radial o Gaussiana, y la sigmoide.

En **R**, el paquete **e1071** contiene las rutinas para aplicar SVM a problemas de clasificación y regresión.

CAPÍTULO 6

Regresión

En la Figura 6.1 se observa un conjunto de muestras caracterizadas por la humedad relativa y la temperatura. La humedad relativa es el cociente entre la presión parcial del vapor de agua y la presión de equilibrio del vapor de agua a una cierta temperatura[50]. Por su lado, en una mezcla de gases cada gas tiene su presión parcial, la cual es la presión hipotética de esa cantidad de gas si ocupara todo el volumen a esa temperatura. Por otra parte la presión de equilibrio de vapor es la presión que ejerce un vapor en equilibrio termodinámico con su fase condensada a una cierta temperatura en un sistema cerrado[50]. En otras palabras, la humedad relativa es la humedad actual del vapor de agua con respecto a la máxima que puede tenerse a una cierta temperatura. En este capítulo exploramos algoritmos de regresión para modelar cuantitativamente comportamientos como este. A manera de ejemplo exploramos dos soluciones: Una utilizando un modelo discriminativo en donde aprendemos los parámetros por máxima verosimilitud (ML); enseguida, exploramos un modelo generativo en donde la evidencia y conocimiento previo son conjugados para obtener la mejor solución.

Enseguida estudiamos la aproximación de modelos regresión no lineales. Para ilustrar esta situación, considere el caso presentado en la Figura 6.4. En ella, se tiene la toma de la temperatura al paso del tiempo. En principio, aunque la hora del día es un dato sobre el cual se tiene mucha confianza, la lectura de la temperatura se obtiene con un cierto margen de incertidumbre. Asimismo, en la Figura, la lectura se obtiene cada 10 minutos. En ocasiones se tiene la necesidad de estimar la temperatura en los valores intermedios. Para resolver el problema de la regresión, consideramos la representación de la incertidumbre. Asímismo, consideramos la proyección de las observaciones a espacios no lineales en donde el herramental que hemos desarrollado previamente nos puede ser de utilidad.

6.1. Modelo Discriminativo

En este modelo identificamos una función de distribución de probabilidad (pdf) que aproxime los datos y utilizamos ML para el aprendizaje de los parámetros. Para ello, definimos la variación de la temperatura del aire en función de la humedad de acuerdo al modelo

$$p(w|x,\theta) = \mathrm{Norm}_w(\phi_0 + \phi x, \sigma^2), \tag{6.1}$$

donde w es la temperatura del aire, x es la humedad relativa y σ es la desviación estándar. Por tanto, los parámetros θ corresponden a $\{\phi_0, \phi_1, \sigma^2\}$. Nuestro objetivo es obtener los parámetros θ del modelo tal que se maximice la capacidad predictiva del modelo. Esto se

puede expresar como

$$\hat{\theta} = \arg\max_{\theta} \left(p(w_{1\ldots n} | x_{1,\ldots,n}, \theta) \right), \tag{6.2}$$

donde aplicando un criterio de independencia entre observaciones, tenemos

$$\hat{\theta} = \arg\max_{\theta} \left(\prod_{i=1}^{n} p(w_i | x_i, \theta) \right). \tag{6.3}$$

El criterio de verosimilitud puede expresarse como

$$L = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ -\frac{1}{2} \left(\frac{w - \phi_0 - \phi_1 x}{\sigma} \right)^2 \right\}. \tag{6.4}$$

Para simplificar aplicamos logaritmos a ambos lados de la expresión, lo que resulta en

$$\ln L = \sum_{i=1}^{n} \left\{ -\ln\sqrt{2\pi} - \ln\sigma - \frac{1}{2} \left(\frac{w - \phi_0 - \phi_1 x}{\sigma} \right)^2 \right\}. \tag{6.5}$$

Para obtener los valores de ϕ_0, ϕ_1, σ que maximizan (6.5) derivamos con respecto a esos parámetros e igualamos a cero. Esto resulta en las expresiones

$$\frac{\partial \ln L}{\partial \phi_0} = \sum_{i=1}^{n} w_i - \phi_0 n - \phi_1 \sum_{i=1}^{n} x_i = 0, \tag{6.6}$$

$$\frac{\partial \ln L}{\partial \phi_1} = \sum_{i=1}^{n} w_i x_i - \phi_0 \sum_{i=1}^{n} x_i - \phi_1 \sum_{i=1}^{n} x_i^2 = 0, \tag{6.7}$$

y

$$\frac{\partial \ln L}{\partial \sigma} = -n + \frac{\sum_{i=1}^{n} (w_i - \phi_0 - \phi_1 x_i)^2}{\sigma^2} = 0, \tag{6.8}$$

Resolviendo (6.8) para σ^2 resulta en la expresión

$$\sigma^2 = \frac{\sum_{i=1}^{n} (w_i - \phi_0 - \phi_1 x_i)^2}{n}. \tag{6.9}$$

Mientras que de (6.6) y (6.7) tenemos el valor de ϕ_1 dado por

$$\phi_1 = \frac{1/n \sum_{i=1}^{n} w_i x_i - \overline{wx}}{1/n \sum_{i=1}^{n} x_i^2 - \overline{x}^2}, \tag{6.10}$$

y

$$\phi_0 = \overline{w} - \phi_1 \overline{x}, \tag{6.11}$$

en donde $\overline{x} = 1/n \sum_{i=1}^{n} x_i$ y $\overline{w} = 1/n \sum_{i=1}^{n} w_i$.

Para los datos que dan origen a la Figura 6.1, la obtención de los parámetros por ML resulta en los valores $\phi_0 = -0.18$, $\phi_1 = 28.79^o C$, y una desviación estándar σ de 5.98 $^o C$. El valor de ϕ_0 podría interpretar la relación en la cual al aumentar la humedad relativa baja la temperatura.

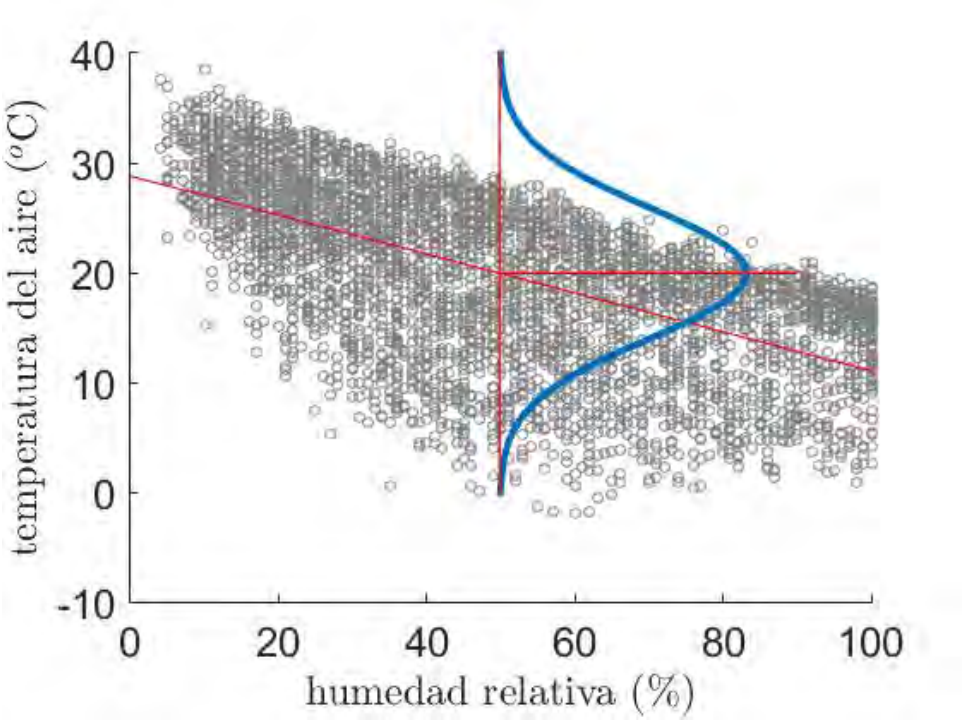


Figura 6.1: Relación entre la humedad relativa y la temperatura del aire. El modelo de regresión lineal resulta en los parámetros $\phi_0 = -0.18$, $\phi_1 = 28.79^oC$, y una desviación estándar σ de 5.98 ^oC.

6.2. Modelo Generativo

El proceso Bayesiano mediante el cual obtenemos el estado del mundo a partir de los datos se define como

$$p(w|x,\theta) = \frac{p(x|w,\theta)p(w,\theta)}{p(x,\theta)}. \tag{6.12}$$

En el modelo generativo se define una función que modele el comportamiento de los datos sujeto al estado que guarda el mundo. Como en el caso anterior, este modelo puede ser definido en términos de una Gaussiana como

$$p(x|w,\theta) = \text{Norm}_x(\phi_0 + \phi w, \sigma^2). \tag{6.13}$$

De una forma similar podemos asumir el prior del estado del mundo como distribuido en forma Gaussiana

$$p(w|\theta) = \text{Norm}_w(\mu_p, \sigma_p^2). \tag{6.14}$$

Prince[56] muestra que si una Gaussiana es conjugada de si misma el resultado del producto es una Gaussiana. En otras palabras

$$\text{Norm}_{\mathbf{x}}(\mathbf{a},\mathbf{A})\text{Norm}_{\mathbf{x}}(\mathbf{b},\mathbf{B}) = k\text{Norm}_{\mathbf{x}}(\mathbf{\Sigma}(\mathbf{A}^{-1}\mathbf{a} + \mathbf{B}^{-1}\mathbf{b}), \mathbf{\Sigma}), \tag{6.15}$$

donde la covarianza $\mathbf{\Sigma}$ corresponde a la siguiente expresión

$$\mathbf{\Sigma} = (\mathbf{A}^{-1} + \mathbf{B}^{-1})^{-1}. \tag{6.16}$$

Nuestro caso, notamos que $p(w|\theta)$ y $p(x|w,\theta)$ no están definidas sobre la misma variable. Por ello, realizamos un cambio de variable sobre $p(w|\theta)$ para expresarla en términos de x. Prince[56] plantea las siguientes ecuaciones para realizar un cambio de variable en una Gaussiana. La igualdad que se establece está dada por

$$\text{Norm}_{\mathbf{x}}(\mathbf{A}\mathbf{y}+\mathbf{b},\mathbf{\Sigma}) = k\text{Norm}_{\mathbf{y}}(\mathbf{A}'\mathbf{x}+\mathbf{b}',\mathbf{\Sigma}')), \tag{6.17}$$

donde

$$\begin{aligned} \mathbf{\Sigma}' &= \left(\mathbf{A}^T\mathbf{\Sigma}^{-1}\mathbf{A}\right)^{-1}, \\ \mathbf{A}' &= \mathbf{\Sigma}'\mathbf{A}^T\mathbf{\Sigma}^{-1}, \\ \mathbf{b}' &= -\mathbf{A}'\mathbf{b}. \end{aligned} \tag{6.18}$$

En nuestro caso, la relación $w = \phi_1 x + \phi_0$ permite que la media μ_p pueda expresarse como $\mu_p = \phi_1\mu_q + \phi_0$, donde μ_q es la media con respecto a la variable aleatoria x. Con ello, tenemos que $\mathbf{A} = \phi_1$, $\mathbf{b} = \phi_0$ y $\mathbf{\Sigma} = \sigma_p$. Por tanto, las variables $\mathbf{A}', \mathbf{b}'$ y $\mathbf{\Sigma}'$ quedan definidas como

$$\begin{aligned} \mathbf{\Sigma}' &= \left(\phi_1\left(\sigma_p^2\right)^{-1}\phi_1\right)^{-1} = \left(\frac{\phi_1^2}{\sigma_p^2}\right)^{-1} = \frac{\sigma_p^2}{\phi_1^2}. \\ \mathbf{A}' &= \frac{\sigma_p^2}{\phi_1^2}\phi_1\left(\sigma_p^2\right)^{-1} = \frac{\sigma_p^2}{\phi_1^2}\frac{\phi_1}{\sigma_p^2} = \frac{1}{\phi_1}, \\ \mathbf{b}' &= -\frac{1}{\phi_1}\phi_0 = -\frac{\phi_0}{\phi_1}. \end{aligned} \tag{6.19}$$

Por tanto tenemos la siguiente equivalencia

$$\text{Norm}_w(\mu_p,\sigma_p^2) = k\text{Norm}_x\left(\frac{1}{\phi_1}\mu_p - \frac{\phi_0}{\phi_1}, \frac{\sigma_p^2}{\phi_1^2}\right) = \text{Norm}_x\left(\frac{\mu_p-\phi_0}{\phi_1}, \frac{\sigma_p^2}{\phi_1^2}\right). \tag{6.20}$$

Una vez realizado el cambio de variable podemos proceder a realizar la multiplicación expresada en (6.15). Es decir,

$$\begin{aligned} p(x|w,\theta)p(w|\theta) &= \text{Norm}_x(\phi_0+\phi_1 w,\sigma^2)\text{Norm}_w(\mu_p,\sigma_p^2) \\ &= k\text{Norm}_x(\phi_0+\phi_1 w,\sigma^2)\text{Norm}_x\left(\frac{\mu_p-\phi_0}{\phi_1}, \frac{\sigma_p^2}{\phi_1^2}\right). \end{aligned} \tag{6.21}$$

Realizando el cambio de variable $\mathbf{a} = \phi_1 w + \phi_0$, $\mathbf{A} = \sigma^2$, $\mathbf{b} = (\mu_p - \phi_0)/\phi_1$ y $\mathbf{B} = \sigma_p^2/\phi_1^2$, tenemos que la media $\mathbf{\Sigma}$ puede expresarse como

$$\mathbf{\Sigma} = \left(\left(\sigma^2\right)^{-1} + \left(\frac{\sigma_p^2}{\phi_1^2}\right)^{-1}\right)^{-1} = \left(\frac{1}{\sigma^2} + \frac{\phi_1^2}{\sigma_p^2}\right)^{-1}. \tag{6.22}$$

Con ello, el producto en (6.21) puede expresarse como

$$p(x|w,\theta)p(w|\theta) = k'\text{Norm}_{\mathbf{x}}(\mathbf{\Sigma}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b}),\mathbf{\Sigma}), \tag{6.23}$$

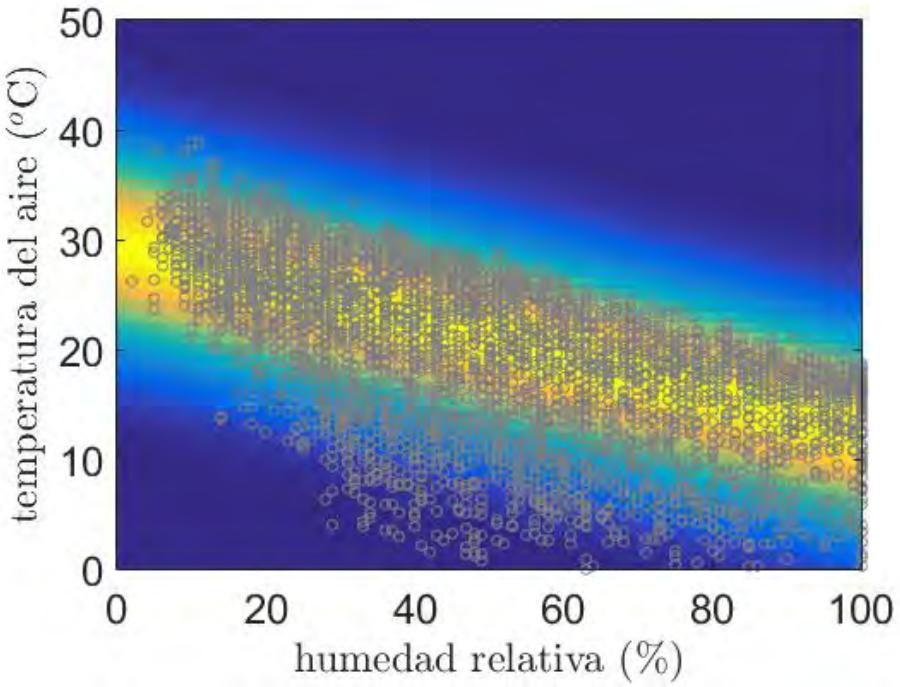


Figura 6.2: Regresión usando un modelo generativo. La gráfica muestra el modelo Gaussiano para la interpretación de la temperatura del aire en función de la humedad relativa. En la figura la pdf tiene sobrepuesta los puntos muestrales.

y k' es la constante de proporcionalidad requerida. Comparando (6.12) y (6.23) vemos que la solución al modelo generativo es

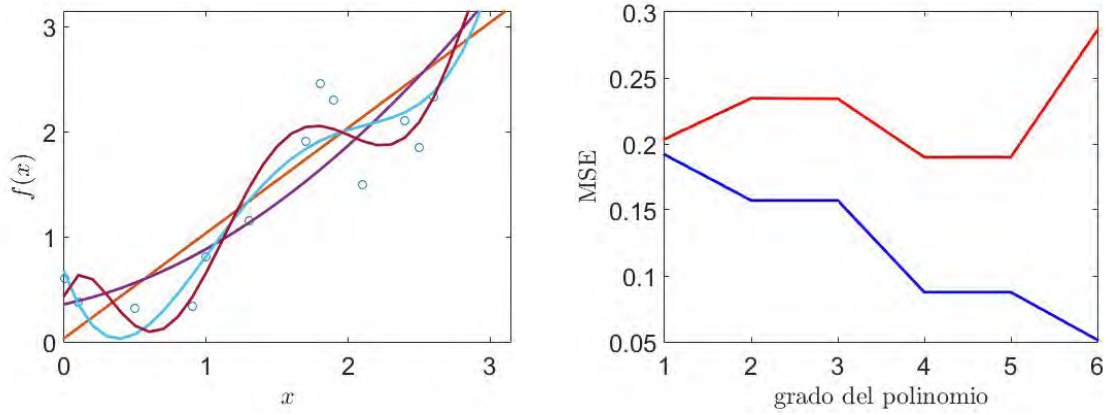
$$p(w|x,\theta) = \text{Norm}_{\mathbf{x}}(\mathbf{\Sigma}(\mathbf{A}^{-1}\mathbf{a} + \mathbf{B}^{-1}\mathbf{b}), \mathbf{\Sigma}). \tag{6.24}$$

La Figura 6.2 muestra el resultado de la implementación del modelo sobre los datos de humedad relativa contra temperatura del aire para la estación meteorológica de Calvillo. Los valores de $\mu_p = 19.91$ y $\sigma_p^2 = 35.0$ fueron obtenidos usando ML sobre los valores de $\{w_i\}_{i=1}^n$. Por su lado, los valores de ϕ_0, ϕ_1 y σ^2 fueron obtenidos mediante ML sobre los valores de $\{x_i, w_i\}_{i=1}^n$, tal como se describe en la subsección anterior.

6.3. Calidad del Ajuste

Una de las tareas más importantes en los métodos que estudiamos consiste en valorar que tan buena es la solución alcanzada en términos de su poder de generalización. Es decir, los parámetros son obtenidos al procesar un conjunto de datos $\{\mathbf{x}_i, w_i\}_{i=1}^n$. Sin embargo, en una gran cantidad de problemas nuestro interés se centra en saber el desempeño que nuestro modelo f tendrá ante datos que no ha visto. Para regresión, la medida de desempeño estándar es el error cuadrático medio (MSE). Este se define como

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(w_i - f(\mathbf{x}_i))^2. \tag{6.25}$$



(a) (b)

Figura 6.3: Selección de modelos. Un modelo más complejo tiene más flexibilidad y puede ajustar mejor a un conjunto de observaciones. Sin embargo, cuando hay *overfiting* el error de entrenamiento se hace más pequeño pero el error de prueba se hace más grande.

Para predecir la capacidad para generalizar, normalmente dividimos nuestro conjunto de datos en tres subconjuntos: entrenamiento, validación y prueba. El conjunto de datos de entrenamiento nos sirve para aprender los parámetros del modelo. Luego, el conjunto de validación nos sirve para comprobar los parámetros de desempeño. Si el modelo requiere algún ajuste podemos volver al conjunto de entrenamiento y nuevamente probar con el conjunto de validación. El conjunto de prueba solo se utiliza una vez. Sus resultados nos dan una idea del poder de generalización del modelo ante datos no vistos. Para la selección de la partición solo tenemos reglas generales que dependen del número de datos disponibles. También en general, grandes cantidades de datos ayudan a tener mejores evaluaciones. Sin embargo, no siempre es posible tener muchos datos. Ya sea porque tenerlos es caro o difícil. Cuando el número de datos es reducido utilizamos técnicas tales como *leave-one-out*, que será estudiado más adelante en el curso.

El tipo de modelo que se selecciona es muy importante. En general, tendemos a preferir modelos sencillos para interpretar la realidad. Una regla de dedo para la selección de la complejidad de un modelo la tenemos ilustrada en la Figura 6.3. En ella tenemos un proceso lineal que genera observaciones ruidosas. Supongamos que tenemos la posibilidad de elegir modelos basados en curvas polinomiales de diversos grados. La figura muestra que a mayor número de grados el ajuste es mejor. Esto se ve reflejado en el MSE. Sin embargo, para el caso en donde evaluamos con datos no utilizados en la etapa de entrenamiento observamos que el MSE se incrementa. En general, cuando el error de entrenamiento disminuye pero el error con los datos de prueba aumenta estamos ante la presencia del fenómeno de sobreajuste u *overfitting*.

Una observación importante sobre los resultados expresados en la Figura 6.3 se refiere al compromiso entre la varianza y el *bias*. La varianza aquí se refiere a la variación que observamos en el resultado del modelo al utilizar el conjunto de prueba cuando empleamos un conjunto diferente de datos para entrenamiento. *Bias* por su lado se refiere al error generado al seleccionar un modelo simple cuando uno más complicado sería necesario. En

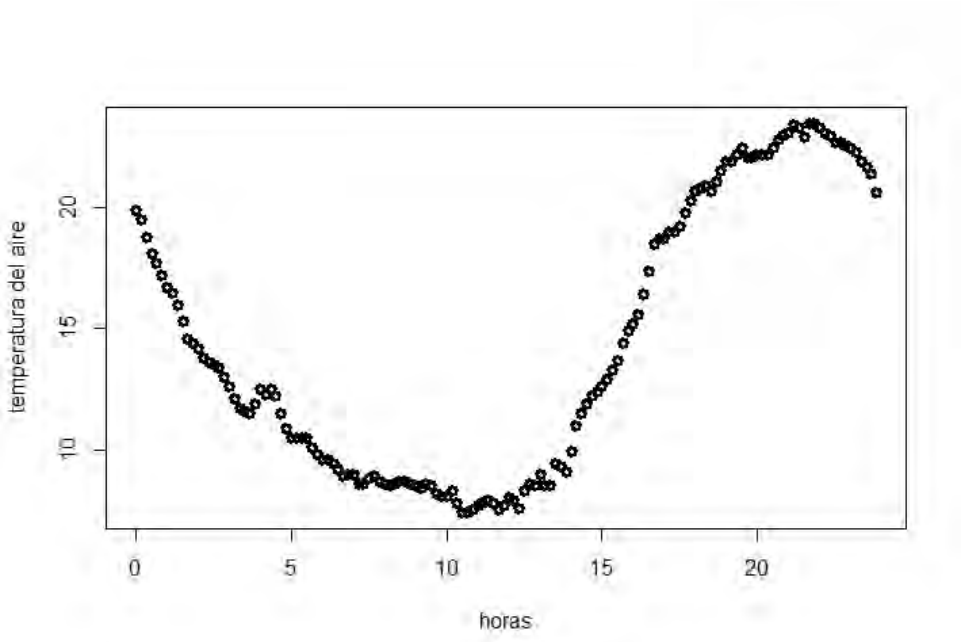


Figura 6.4: La temperatura del aire en función del tiempo tiene un comportamiento no lineal.

general cuando incrementamos la complejidad del modelo disminuimos el error por bias. Sin embargo, al aumentar la complejidad del modelo el error por varianza comienza a aumentar. El problema consiste básicamente en encontrar el punto en la complejidad del modelo cuya suma de errores por bias y varianza den el mínimo.

6.4. Regresión Lineal

El problema de la regresión lineal consiste en identificar los mejores parámetros $\phi = [\phi_0, \phi_1, \ldots, \phi_n]$ de una línea recta

$$w_i = \phi_0 + \phi_1 x_i^1 + \cdots + \phi_n x_i^i = \phi^T \mathbf{x}_n, \tag{6.26}$$

que describe un conjunto de observaciones $\{\mathbf{x}_i\}_{i=1}^m$ de la forma $\mathbf{x}_i = [1, x_i^1, \ldots, x_i^n]^T$.

Si suponemos que las observaciones están sujetas a ruido Gaussiano y que la desviación estándar σ es constante, la probabilidad de observar un valor particular w_i dada una observación $\mathbf{x}_i$ está dada por

$$p(w_i|\mathbf{x}_i, \theta) = \mathrm{Norm}_{w_i}(\mathbf{x}_i^T \phi, \sigma^2). \tag{6.27}$$

En consecuencia, si agrupamos todas las observaciones en la matriz $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ y el vector $\mathbf{w} = [w_1, \ldots, w_m]$, tendremos la expresión

$$p(\mathbf{w}|\mathbf{X}, \theta) = \mathrm{Norm}_{\mathbf{w}}(\mathbf{X}^T \phi, \sigma^2) = \frac{1}{(2\phi)^{D/2} \cdot \sigma^D} \exp\left(-1/2(\mathbf{X}^T \phi - \mathbf{w})^T (\mathbf{X}^T \phi - \mathbf{w})\right) / \sigma^2. \tag{6.28}$$

Bajo esa formulación, el mejor valor para los parámetros puede encontrarse en la posición donde la derivada con respecto a ellos es igual a cero. Esa posición no cambia si uno calcula el logaritmo antes. Sin embargo, la expresión resultante puede ser más fácil de evaluar. El logaritmo de (6.28) es

$$\ln p(\mathbf{w}|\mathbf{X},\theta) = -\frac{D}{2}\ln(2\pi) - D\ln\sigma - \frac{1}{2}(\mathbf{X}^T\phi - \mathbf{w})^T(\mathbf{X}^T\phi - \mathbf{w})/\sigma^2 = L. \tag{6.29}$$

Enseguida, la derivada de L con respecto a σ es igualada con cero, lo que resulta en

$$\partial L/\partial\sigma = -\frac{D}{\sigma} - 2\frac{1}{2}(\mathbf{X}^T\phi - \mathbf{w})^T(\mathbf{X}^T\phi - \mathbf{w})/\sigma^3 = 0. \tag{6.30}$$

De donde resulta que el valor de la covarianza es

$$\sigma^2 = \frac{(\mathbf{X}^T\phi - \mathbf{w})^T(\mathbf{X}^T\phi - \mathbf{w})}{D}. \tag{6.31}$$

De forma similar, la derivada de L con respecto a ϕ es igualada con cero, lo que resulta en

$$\partial L/\partial\phi = -2\frac{1}{2}(\phi^T\mathbf{X} - \mathbf{w}^T)\mathbf{X}^T/\sigma^2 = 0, \tag{6.32}$$

que puede ser simplificada, en virtud de que está igualada con cero a

$$(\phi^T\mathbf{X} - \mathbf{w}^T)\mathbf{X}^T = 0. \tag{6.33}$$

Para resolver con respecto a ϕ despejamos $\mathbf{w}$. Luego, multiplicamos ambos lados por $\mathbf{X}$. Dado que ahora $\mathbf{X}\mathbf{X}^T$ es cuadrada y suponiendo que su inversa existe, el valor de ϕ puede obtenerse como

$$\phi = \left(\mathbf{X}\mathbf{X}^T\right)^{-1}\mathbf{X}\mathbf{w}. \tag{6.34}$$

Consecuentemente, los valores de (6.31) y (6.34) para σ y ϕ, respectivamente, permiten ajustar un modelo lineal al conjunto de observaciones en $\mathbf{X}$ y $\mathbf{w}$.

6.5. Regresión Bayesiana

La derivación en la sección anterior supone que es posible llegar a un conocimiento perfecto de los parámetros. Un enfoque alternativo basado en la regla de Bayes permite tomar en cuenta la incertidumbre que tenemos en el valor que le asignamos. Esto es, el valor de certeza que tenemos en el valor de ϕ dadas las observaciones $\mathbf{X}$ y $\mathbf{w}$ está dado por

$$p(\phi|\mathbf{X},\mathbf{w}) = \frac{p(\mathbf{w}|\mathbf{X},\phi)p(\phi)}{p(\mathbf{w}|\mathbf{X})}. \tag{6.35}$$

La verosimilitud $p(\mathbf{w}|\mathbf{X},\phi)$ puede expresarse como en (6.28), mientras que el prior pueden expresarse como

$$p(\phi) = \mathrm{Norm}_\phi(0,\sigma^2). \tag{6.36}$$

Es decir, (6.35) puede ser expresada por el siguiente producto

$$p(\phi|\mathbf{X},\mathbf{w}) = \text{Norm}_{\mathbf{w}}(\mathbf{X}^T\phi,\sigma^2)\text{Norm}_{\phi}(0,\sigma^2). \tag{6.37}$$

La conveniencia de expresar ambas pdf como Gaussianas estriba en que puede calcularse de forma cerrada el producto. Sin embargo, primero hay que realizar un cambio de variable sobre una de las pdfs. Para realizar el cambio de variable, las siguientes expresiones pueden ser usadas[56]. Un cambio de variable entre las pdfs

$$\text{Norm}_{\mathbf{x}}(\mathbf{A}\mathbf{y}+\mathbf{b},\mathbf{\Sigma}) = \kappa\text{Norm}_{\mathbf{y}}(\mathbf{A}'\mathbf{x}+\mathbf{b}',\mathbf{\Sigma}), \tag{6.38}$$

donde $\mathbf{\Sigma}' = (\mathbf{A}^T\mathbf{\Sigma}^{-1}\mathbf{A})^{-1}$, $\mathbf{A}' = \mathbf{\Sigma}'\mathbf{A}^T\mathbf{\Sigma}^{-1}$ y $\mathbf{b}' = -\mathbf{A}'\mathbf{b}$.

Para el caso que nos ocupa, queremos transformar la pdf $\text{Norm}_{\mathbf{w}}(\mathbf{X}^T\phi,\sigma^2)$. Para la transformación, las variables están definidas como $\mathbf{\Sigma}' = \sigma^2(\mathbf{X}\mathbf{X}^T)^{-1}$, $\mathbf{A}' = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}$ y $\mathbf{b}' = 0$. Con ello, el cambio de variable puede planterse como

$$\text{Norm}_{\mathbf{w}}(\mathbf{X}^T\phi,\sigma^2) = \kappa_1\text{Norm}_{\phi}(\mathbf{A}'\mathbf{w},\mathbf{\Sigma}'). \tag{6.39}$$

Enseguida, planteamos el producto identificado en (6.40), resultando en

$$p(\phi|\mathbf{X},\mathbf{w}) = \kappa_1\text{Norm}_{\phi}(\mathbf{A}'\mathbf{w},\mathbf{\Sigma}')\text{Norm}_{\phi}(0,\sigma^2). \tag{6.40}$$

Ahora podemos proceder a realizar el producto entre dos Gaussianas. En forma general, el producto de dos normales se define como

$$\begin{aligned}\text{Norm}_{\mathbf{x}}(\mathbf{a},\mathbf{A})\cdot\text{Norm}_{\mathbf{x}}(\mathbf{b},\mathbf{B}) = \\ \kappa\text{Norm}_{\mathbf{x}}((\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b}),(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}),\end{aligned} \tag{6.41}$$

donde κ es en sí una Gaussiana definida como

$$\kappa = \text{Norm}_{\mathbf{a}}(\mathbf{b},\mathbf{A}+\mathbf{B}) = \text{Norm}_{\mathbf{b}}(\mathbf{a},\mathbf{A}+\mathbf{B}). \tag{6.42}$$

Para nuestro caso en particular, $\mathbf{a} = \mathbf{A}'\mathbf{w}$, $\mathbf{A} = \mathbf{\Sigma}'$, $\mathbf{b} = 0$ y $\mathbf{B} = \sigma_p^2\mathbf{I}$. De donde resulta que

$$\begin{aligned}p(\phi|\mathbf{X},\mathbf{w}) &= \text{Norm}_{\phi}(\mathbf{A}'\mathbf{w},\mathbf{\Sigma}')\text{Norm}_{\phi}(0,\sigma^2) = \\ &= \kappa_2\text{Norm}_{\phi}((\mathbf{\Sigma}'^{-1}+(\sigma_p^2\mathbf{I})^{-1})^{-1}\mathbf{\Sigma}'^{-1}\mathbf{A}'\mathbf{w},(\mathbf{\Sigma}'^{-1}+(\sigma_p^2\mathbf{I})^{-1})^{-1}), \\ &= \kappa_2\text{Norm}_{\phi}((1/\sigma^2\mathbf{X}\mathbf{X}^T+1/\sigma_p^2\mathbf{I})^{-1}1/\sigma^2\mathbf{X}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{w},(1/\sigma^2\mathbf{X}\mathbf{X}^T+1/\sigma_p^2\mathbf{I})^{-1}), \\ &= \kappa_2\text{Norm}_{\phi}((1/\sigma^2\mathbf{X}\mathbf{X}^T+1/\sigma_p^2\mathbf{I})^{-1}1/\sigma^2\mathbf{X}\mathbf{w},(1/\sigma^2\mathbf{X}\mathbf{X}^T+1/\sigma_p^2\mathbf{I})^{-1}), \\ &= \kappa_2\text{Norm}_{\phi}(1/\sigma^2\mathbf{A}^{-1}\mathbf{X}\mathbf{w},\mathbf{A}^{-1}),\end{aligned} \tag{6.43}$$

con $\mathbf{A} = (1/\sigma^2\mathbf{X}\mathbf{X}^T + 1/\sigma_p^2\mathbf{I})$. Este valor es muy importante pues permite realizar inferencia. Así, dado una nueva observación $\mathbf{x}^*$, uno podría estimar la probabilidad de su resultado $p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w})$ usando la expresión

$$\begin{aligned} p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) &= \int p(w^*|x^*, \phi)p(\phi|\mathbf{X}, \mathbf{w})\phi, \\ &= \int \kappa_2 \text{Norm}_{w^*}(\mathbf{x}^{*T}\phi, \sigma^2)\text{Norm}_{\phi}(1/\sigma^2\mathbf{A}\mathbf{X}\mathbf{w}, \mathbf{A})d\phi, \\ &= \text{Norm}_{w^*}\left(\frac{1}{\sigma^2}\mathbf{x}^{*T}\mathbf{A}^{-1}\mathbf{X}\mathbf{w}, \mathbf{x}^{*T}\mathbf{A}^{-1}\mathbf{x}^* + \sigma^2\right), \end{aligned} \tag{6.44}$$

donde para resolver nuevamente necesitamos realizar un cambio de variable para el primer factor en términos de ϕ. Antes de resolver la multiplicación notamos que el resultado estará en términos de ϕ, resultando en que la integral dará uno. Por ello, únicamente necesitamos el valor del factor de escala para expresar el resultado.

Prince[56] hace la importante nota que se puede expresar la identidad de Woodbury[71] para expresar la identidad

$$\begin{aligned} \mathbf{A}^{-1} &= (1/\sigma^2\mathbf{X}\mathbf{X}^T + 1/\sigma_p^2\mathbf{I})^{-1} \\ &= \sigma_p^2\mathbf{I}_D - \sigma_p^2\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}\mathbf{X}^T, \end{aligned} \tag{6.45}$$

donde n es el número de observaciones y D el número de variables en la observación. Utilizando esta igualdad en (6.87) tenemos

$$\begin{aligned} p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) &= \text{Norm}_{w^*}\left(\frac{1}{\sigma^2}\mathbf{x}^{*T}\mathbf{A}^{-1}\mathbf{X}\mathbf{w}, \mathbf{x}^{*T}\mathbf{A}^{-1}\mathbf{x}^* + \sigma^2\right), \\ &= \text{Norm}_{w^*}\left(\frac{1}{\sigma^2}\mathbf{x}^{*T}\left(\sigma_p^2\mathbf{I}_D - \sigma_p^2\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}\mathbf{X}^T\right)\mathbf{X}\mathbf{w},\right. \\ &\quad \left.\mathbf{x}^{*T}\left(\sigma_p^2\mathbf{I}_D - \sigma_p^2\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}\mathbf{X}^T\right)\mathbf{x}^* + \sigma^2\right), \\ &= \text{Norm}_{w^*}\left(\frac{\sigma_p^2}{\sigma^2}\mathbf{x}^{*T}\mathbf{X}\mathbf{w} - \frac{\sigma_p^2}{\sigma^2}\mathbf{x}^{*T}\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}\mathbf{X}^T\mathbf{X}\mathbf{w},\right. \\ &\quad \left.\sigma_p^2\mathbf{x}^{*T}\mathbf{x}^* - \sigma_p^2\mathbf{x}^{*T}\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}\mathbf{X}^T\mathbf{x}^* + \sigma^2\right), \end{aligned} \tag{6.46}$$

6.6. Regresión no Lineal

La aproximación para regresión no lineal que estudiamos en esta sección es una en la cual los datos son proyectados a un espacio no lineal, regularmente de mayor dimensión, en donde

la anterior maquinaria de regresión lineal puede es utilizada. Es decir, para una observación particular $\mathbf{x}_i$ buscamos un mapeo no lineal $\mathbf{f}$ de la forma

$$\mathbf{z}_i = \mathbf{f}(\mathbf{x}_i). \tag{6.47}$$

La expectativa es que en ese espacio de mayor dimensión se pueda identificar un hiperplano que sobre el cual el valor w_i pueda ser evaluado. De forma similar al caso lineal, buscamos evaluar la probabilidad de observar un valor w_i dada una observación $\mathbf{x}_i$ de la forma

$$p(w_i|\mathbf{x}_i, \theta) = \text{Norm}_{w_i}(\mathbf{z}_i^T\phi, \sigma^2). \tag{6.48}$$

Tal como en el caso de la regresión lineal, si agrupamos todas las observaciones en la matriz $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_m]$ y el vector $\mathbf{w} = [w_1, \ldots, w_m]$, tendremos la expresión

$$p(\mathbf{w}|\mathbf{X}, \theta) = \text{Norm}_{\mathbf{w}}(\mathbf{Z}^T\phi, \sigma^2) = \frac{1}{(2\phi)^{D/2} \cdot \sigma^D} \exp\left(-1/2(\mathbf{Z}^T\phi - \mathbf{w})^T(\mathbf{Z}^T\phi - \mathbf{w})\right)/\sigma^2. \tag{6.49}$$

Similar a (6.31) y (6.34), los estimadores de los parámetros están dados por las siguientes expresiones. Para la covarianza tenemos

$$\sigma^2 = \frac{(\mathbf{Z}^T\phi - \mathbf{w})^T(\mathbf{Z}^T\phi - \mathbf{w})}{D}, \tag{6.50}$$

y para los parámetros del hiperplano que describe el modelo tenemos

$$\phi = \left(\mathbf{Z}\mathbf{Z}^T\right)^{-1}\mathbf{Z}\mathbf{w}. \tag{6.51}$$

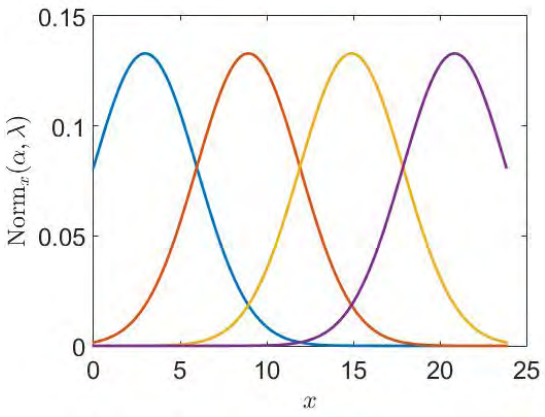
Como ilustración, considérese los datos de la Figura 6.4. Supóngase que se define la función de proyección

$$\mathbf{z}_i = \mathbf{f}(\mathbf{x}_i) = \begin{pmatrix} 1 \\ \text{Norm}_x(\mu_1, \sigma) \\ \cdots \\ \text{Norm}_x(\mu_m, \sigma) \end{pmatrix}, \tag{6.52}$$

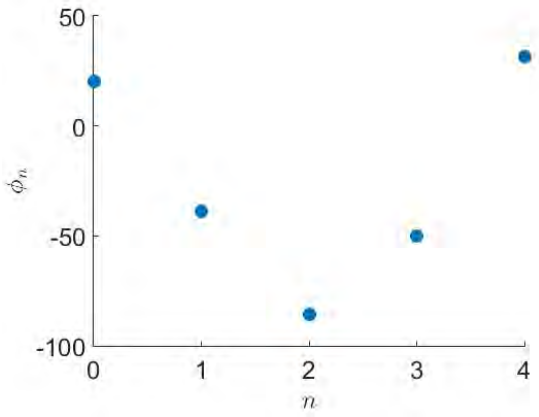
para m curvas base Gaussianas. Usando el conjunto de observaciones $\{x_i, w_i\}$, correspondientes respectivamente a la hora del día y la temperatura, se realizó la estimación de ϕ. La Figura 6.5(a) muestra la solución cuando el intervalo de observación temporal se dividió uniformemente para colocar 4 curvas base Gaussianas. En Figura 6.5(b) se muestran los valores resultantes para ϕ y en la Figura 6.5(c) se muestra la suma pesada individual y colectiva de las curvas base Gaussianas. La posición de las curvas base pudiera ser optimizada pero aquí solo se quiere ilustrar como su combinación lineal puede aproximar razonablemente a la función original.

Utilizando (6.46) como base, la formulacion para regresión Bayesiana pudiera plantearse como

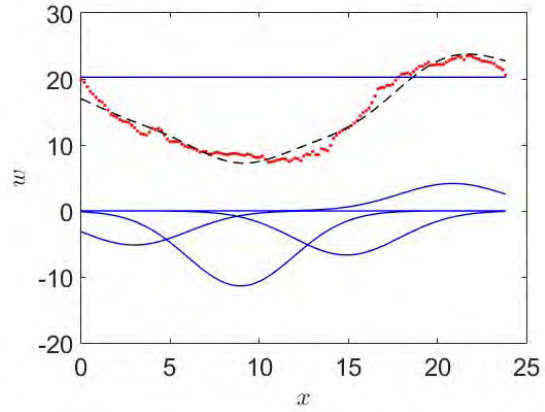
$$\begin{aligned} p(w^*|\mathbf{z}^*, \mathbf{Z}, \mathbf{w}) = \text{Norm}_{w^*}\Bigg(&\frac{\sigma_p^2}{\sigma^2}\mathbf{z}^{*T}\mathbf{Z}\mathbf{w} - \frac{\sigma_p^2}{\sigma^2}\mathbf{z}^{*T}\mathbf{Z}\left(\mathbf{Z}^T\mathbf{Z} + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}\mathbf{Z}^T\mathbf{Z}\mathbf{w}, \\ &\sigma_p^2\mathbf{z}^{*T}\mathbf{z}^* - \sigma_p^2\mathbf{z}^{*T}\mathbf{Z}\left(\mathbf{Z}^T\mathbf{Z} + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}\mathbf{Z}^T\mathbf{z}^* + \sigma^2\Bigg), \end{aligned} \tag{6.53}$$



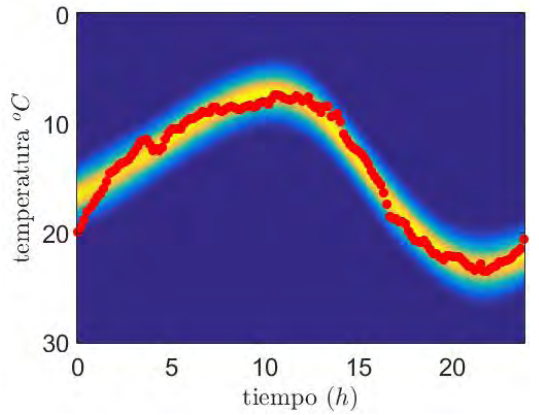
(a) Curvas base Gaussianas



(b) Parámetros ϕ



(c) Ajuste de curvas base Gaussianas



(d) Regresión Bayesiana

Figura 6.5: Aproximación de una curva mediante la combinación lineal de curvas base Gaussianas.

6.7. Kernels

La noción de la representación no lineal y proyección a espacios multidimensionales puede generalizarse mediante el uso de los llamados kernels. Un kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ representa $\mathbf{x}_i$ y $\mathbf{x}_j$ en un espacio definido por la transformación $\mathbf{f}$, resultando en los vectores $\mathbf{z}_i$ y $\mathbf{z}_j$, y ejecuta el producto punto entre los vectores $\mathbf{z}_i$ y $\mathbf{z}_j$, por medio de la siguiente operación

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{f}(\mathbf{x}_i)^T \mathbf{f}(\mathbf{x}_j) = \mathbf{z}_i^T \mathbf{z}_j. \tag{6.54}$$

Como un ejemplo, considere el kernel definido por

$$k(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T \mathbf{x}')^2, \tag{6.55}$$

en donde los vectores están definidos como $\mathbf{x} = (x, y)^T$ y $\mathbf{x}' = (x', y')^T$. En ese caso, la expresión anterior puede representarse como

$$\begin{aligned}
(1 + \mathbf{x}^T \mathbf{x}')^2 &= \left(1 + \begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix}\right)^2, \\
&= (1 + xx' + yy')^2, \\
&= 1 + x^2 x'^2 + y^2 y'^2 + 2xx' + 2yy' + 2xyx'y', \\
&= \begin{pmatrix} 1 & x^2 & y^2 & \sqrt{2}x & \sqrt{2}y & \sqrt{2}xy \end{pmatrix} \begin{pmatrix} 1 \\ x'^2 \\ y'^2 \\ \sqrt{2}x' \\ \sqrt{2}y' \\ \sqrt{2}x'y' \end{pmatrix}.
\end{aligned} \tag{6.56}$$

Como es posible observar, el kernel puede representarse como un producto punto formado por la expansión del vector original a un espacio no lineal.

La teoría de los kernels es muy generosa. Muchas funciones pueden ser un kernel, bastando con ello que satisfagan el teorema de Mercer[1]. El teorema de Mercer establece que un kernel es válido cuando es positivo semidefinido, *i.e.*, satisface la condición

$$\sum_{i,j} k(\mathbf{x}_i, \mathbf{x}'_j) a_i a_j \geq 0, \tag{6.57}$$

para cualquier número real a_i y a_j. Algunos kernels muy utilizados en la práctica incluyen los siguientes

- Lineal
$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'. \tag{6.58}$$
- Polinomial de orden p
$$k(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T \mathbf{x}')^p. \tag{6.59}$$

- Gaussianos o de función base radial

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}\frac{(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)}{\sigma^2}\right). \tag{6.60}$$

El kernel Gaussiano es interesante pues ni siquiera parece tener un producto punto entre dos vectores. Para verificar lo anterior, considere el caso del kernel

$$k(x, x') = \exp\left(-(x - x')^2\right). \tag{6.61}$$

La exponencial puede representarse de la siguiente forma

$$\exp\left(-(x - x')^2\right) = \exp\left(-x^2 + 2xx' - x'^2\right) = \exp\left(-x^2\right)\exp\left(2xx'\right)\exp\left(-x'^2\right). \tag{6.62}$$

El término central puede ser expandido en términos de su serie de Taylor como

$$\exp\left(2xx'\right) = \sum_{k=0}^{\infty} \frac{2^k x^k x'^k}{k!}. \tag{6.63}$$

Realizando esta el reemplazo de esta definción en (6.63) en (6.62) tenemos

$$\exp\left(-x^2\right)\sum_{k=0}^{\infty} \frac{2^k x^k x'^k}{k!}\exp\left(-x'^2\right), \tag{6.64}$$

el cual puede ser representado en forma vectorial como

$$\exp\left(-(x - x')^2\right) = \exp\left(-x^2\right)\begin{pmatrix} 1 & \sqrt{2}x & \dots & \frac{\sqrt{2^n}}{\sqrt{n!}}x^n & \dots \end{pmatrix}\begin{pmatrix} 1 \\ \sqrt{2}x' \\ \vdots \\ \frac{\sqrt{2^n}}{\sqrt{n!}}x'^n \\ \vdots \end{pmatrix}\exp\left(-x'^2\right). \tag{6.65}$$

Mostrando que efectivamente, implícitamente estamos expandiendo x a un espacio de un número infinito de dimensiones antes de realizar el producto punto.

De acuerdo a la expresión anterior, la regresión Bayesiana puede generalizarse al uso de kernels con la siguiente expresión

$$p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) = \text{Norm}_{w^*}\left(\frac{\sigma_p^2}{\sigma^2}k(\mathbf{x}^*, \mathbf{X})\mathbf{w} - \frac{\sigma_p^2}{\sigma^2}k(\mathbf{x}^*, \mathbf{X})\left(k(\mathbf{X}, \mathbf{X}) + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}k(\mathbf{X}, \mathbf{X})\mathbf{w},\right.$$
$$\left.\sigma_p^2 k(\mathbf{x}^*, \mathbf{x}^*) - \sigma_p^2 k(\mathbf{x}^*, \mathbf{X})\left(k(\mathbf{X}, \mathbf{X}) + \frac{\sigma^2}{\sigma_p^2}\mathbf{I}_n\right)^{-1}k(\mathbf{X}, \mathbf{x}^*) + \sigma^2\right). \tag{6.66}$$

Cuadro 6.1: Ilustración de la generación de variables auxiliares. Una variable categórica que puede tomar k valores da origen a $k-1$ variables auxiliares, las cuales toman valores cero o uno, pero no uno más de una.

Variable Original	Variable Auxiliar 1	Variable Auxiliar 2
x	$x1$	$x2$
1	0	0
2	0	1
3	1	0

6.8. Regresión Lineal Dispersa

Dados un conjunto de vectores $\{\mathbf{x}_n\}_{n=1}^N$ y etiquetas correspondientes $\{w_n\}$, el problema del aprendizaje consiste en aprender los parámetros de una función $y(\mathbf{x})$. En la pasada sección, vimos como una forma para esta función consiste en parametrizarla por un conjunto de funciones kernel $k(\mathbf{x}, \mathbf{x}')$, tal como

$$y(\mathbf{x}; \phi) = \sum_{i=1}^{N} \phi_i k(\mathbf{x}, \mathbf{x}_i) + \phi_0, \tag{6.67}$$

donde $\phi = (\phi_1, \ldots, \phi_M\}$ son pesos asociados.

En general, nuestra observación proviene de una alteración aleatoria del valor real, tal como

$$w_n = y(\mathbf{x}; \phi) + \epsilon_n, \tag{6.68}$$

donde w_n sigue una distribución Gaussiana con media cero y covarianza σ^2, $p(w_n|\mathbf{x}) = \text{Norm}_{w_n}(y(\mathbf{x}_n), \sigma^2)$.

Comenzaremos formulando la solución para los datos $\mathbf{X}$, donde el objetivo es determinar los pesos ϕ de la combinación lineal. Para estimarlos, se puede utilizar la siguiente formulación Bayesiana

$$\begin{aligned} p(\phi|\mathbf{X}, \mathbf{w}, \phi) &= \frac{p(\mathbf{w}, \phi|\mathbf{X}, \sigma^2)}{p(\mathbf{w}|\mathbf{X}, \sigma^2)}, \\ &= \frac{p(\mathbf{w}|\mathbf{X}, \phi, \sigma^2)p(\phi)}{p(\mathbf{w}|\mathbf{X}, \sigma^2)}. \end{aligned} \tag{6.69}$$

Hemos visto que asumiendo independencia, la verosimilitud de las observaciones y etiquetas toma la forma

$$p(\mathbf{w}|\mathbf{X}, \phi, \sigma^2) = \text{Norm}_{\phi}(\mathbf{X}^T\phi, \sigma^2\mathbf{I}) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{w} - \mathbf{X}^T\phi\|^2\right\}, \tag{6.70}$$

donde $\mathbf{w} = (w_1, \ldots, w_N)^T$, $\phi = (\phi_0, \ldots, \phi_N)^T$ y y $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$, es una matriz de $N \times (N+1)$, con $\mathbf{x}_n = (1, \mathbf{x}_1, \ldots, \mathbf{x}_N)^T$.

Debido a la gran cantidad de parámetros involucrados uno esperaría *overfitting*. Para evitarlo se definen priors sobre los pesos de los kernels que promueven soluciones dispersas,

tal como

$$p(\phi) = \prod_{d=1}^{M} \text{Stud}_{\phi_d}(0, 1, \nu), \tag{6.71}$$

donde $\text{Stud}(0, 1, \nu)$ es una función de distribución Student de media cero, covarianza uno y grados de libertad ν. Para los propósitos siguientes será útil la interpretación de la pdf de Student como la suma del producto entre una distribución Gamma y una Normal. Lo anterior queda expresado como

$$p(\phi) = \prod_{d=1}^{M} \int \text{Norm}_{\phi}(0, 1/h_d)\text{Gam}_{h_d}(\nu/2, \nu/2)dh_d, \tag{6.72}$$

donde h_d es un hiperparámetro que se define para cada una de las observaciones. Dado que la Gaussiana es separable, podemos conjuntar los efectos de las multiplaciones en un solo factor como

$$p(\phi) = \int \text{Norm}_{\phi}(0, \mathbf{H}^{-1}) \prod_{d=1}^{M} \text{Gam}_{h_d}(\nu/2, \nu/2)dh_d, \tag{6.73}$$

La característica esencial del ajuste que buscamos es la asignación de un hiperparámetro a cada peso o función base de tal forma de promover una solución dispersa.

El denominador de (6.69) puede emplearse para el cálculo de la pdf de observar un valor particular de $\mathbf{w}$ como la siguiente marginal

$$\begin{aligned}
p(\mathbf{w}|\mathbf{X}, \phi) &= \int p(\mathbf{w}, \phi|\mathbf{X}, \sigma^2)d\phi, \\
&= \int p(\mathbf{w}|\mathbf{X}, \phi, \sigma^2)p(\phi)d\phi, \\
&= \int \text{Norm}_{\phi}(\mathbf{X}^T\phi, \sigma^2\mathbf{I}) \int \text{Norm}_{\phi}(0, \mathbf{H}^{-1}) \prod_{d=1}^{M} \text{Gam}_{h_d}(\nu/2, \nu/2)d\mathbf{H}d\phi, \\
&= \int \int \text{Norm}_{\phi}(\mathbf{X}^T\phi, \sigma^2\mathbf{I})\text{Norm}_{\phi}(0, \mathbf{H}^{-1})d\phi \prod_{d=1}^{M} \text{Gam}_{h_d}(\nu/2, \nu/2)d\mathbf{H}, \\
&= \int \text{Norm}_{\phi}(\mathbf{0}, \mathbf{X}^T\mathbf{H}^{-1}\mathbf{X} + \sigma^2\mathbf{I}) \prod_{d=1}^{M} \text{Gam}_{h_d}(\nu/2, \nu/2)d\mathbf{H}d\phi.
\end{aligned} \tag{6.74}$$

Debido a que no hay conjugados, la anterior expresión no se puede externar de forma cerrada. Por tanto se aproxima mediante la relación

$$p(\mathbf{w}|\mathbf{X}, \phi) \approx \max_{\mathbf{H}} \left(\int \text{Norm}_{\phi}(\mathbf{0}, \mathbf{X}^T\mathbf{H}^{-1}\mathbf{X} + \sigma^2\mathbf{I}) \prod_{d=1}^{M} \text{Gam}_{h_d}(\nu/2, \nu/2)d\mathbf{H} \right). \tag{6.75}$$

Los parámetros $\mathbf{H}$ y σ^2 se obtienen aplicando logaritmos a esta expresión, derivandola con respecto a los parámetros, igualando la expresión a cero y resolviendo con respecto a los parámetros. Aún así, hay que resolver el valor de los parámetros usando el algoritmo 3.

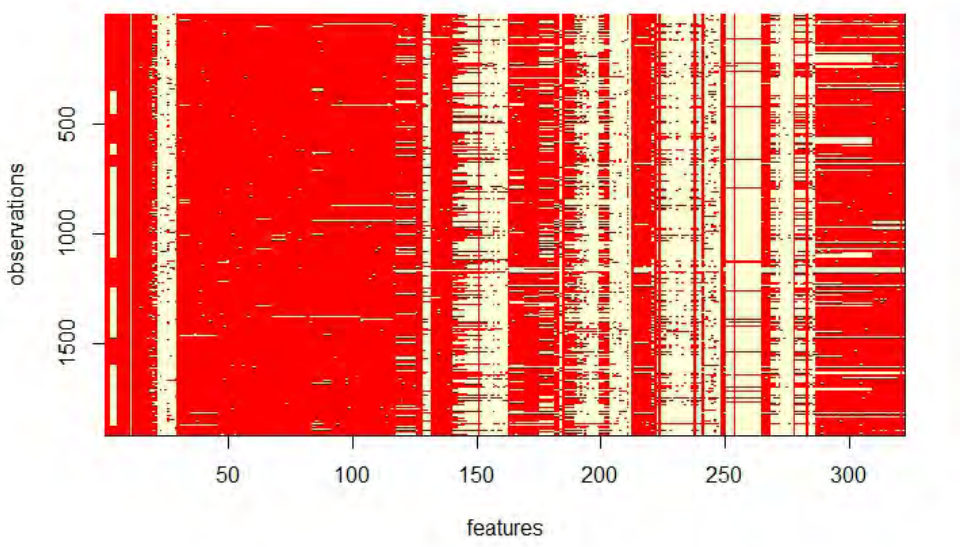
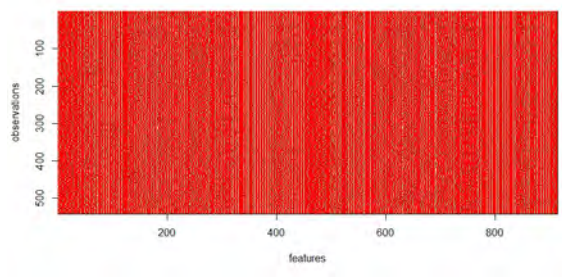


Figura 6.6: Matriz de llenado de la encuesta 2015 sobre consumo de adicciones llevada a cabo por el Observatorio Ciudadano de Seguridad Pública del Municipio de Querétaro

Para ilustrar la importancia de la selección de características utilizamos el siguiente ejemplo. Supóngase que se tiene una base de datos con el resultado de la Encuesta Anual de Adicciones producido por el Observatorio Ciudadano de Seguridad Pública del Municipio de Querétaro sobre el consumo de sustancias. Estamos interesados en construir una función de regresión sobre la edad a la que se consume la primera copa de alcohol completa. Normalmente, estas encuestas son exhaustivas. Por ejemplo, en la encuesta del 2015 se tomaron 322 variables sobre un universo de 1917 encuestados. La matriz de llenado tiene la forma que se ilustra en la Figura 6.6. En nuestro problema, estamos interesados en determinar si algunas de las variables son más importantes que otras en la determinación de la variable de interés. Para realizar este ejercicio, vamos a descartar todas las variables no númericas y de ahí tomar las entradas correspondientes al bloque de predictores más densamente poblado, aproximadamente entre las columnas 20 y 120.

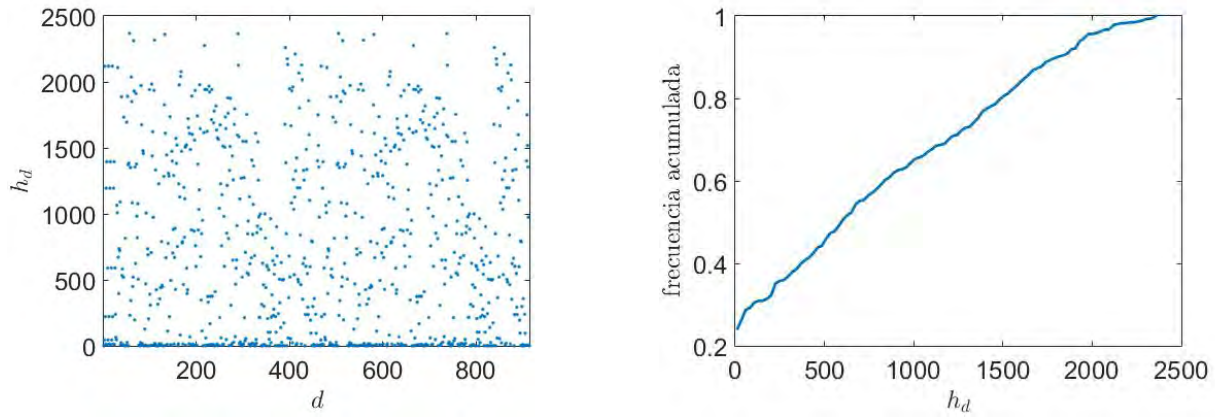
Una nota importante es que las variables en este intervalo son categóricas y no pueden ser introducidas directamente en los mecanismos de regresión. Por ejemplo, supongamos que la variable x tiene un rango de valores entre 1 y 3. Para estos casos, en donde se tienen variables categóricas que pueden tomar k valores, se definen $k-1$ variables auxiliares con valores cero o uno. Para el caso anterior se definirían variables *x1* y *x2* y los valores de x se representarían como se ilustra en la Tabla 6.1. Una vez que las variables auxiliares han sido definidas, se tienen 541 observaciones cada una con 914 variables. Esto se ilustra en la Figura 6.7.

La Figura 6.8 muestra el resultado de 1,000 iteraciones del Algoritmo 3 para el caso de la base de datos comentada al inicio. En (a) se muestra el valor individual de las variables ocultas. Para visualizar el establecimiento del umbral, en (b) se muestra la frecuencia acu-



(a) Matriz con variables auxiliares (b) Edad en la que se tomó una copa completa

Figura 6.7: Matriz conteniendo los predictores (a) y la predicción (b) para el problema de determinar la edad para tomar la primera copa.



(a) Valor de las variables auxiliares (b) Frecuencia acumulada de variables auxiliares

Figura 6.8: Resultado de correr el Algoritmo 3 con los datos en la base de datos sobre el consumo de sustancias.

mulada de los valores de las variables ocultas. La figura ilustra el porcentaje de variables ocultas que se encuentran abajo de un cierto valor de umbral.

6.9. Regresión Lineal Dual

Sea una matriz $\mathbf{A}_{n\times m}$, un vector $\mathbf{x}_{m\times 1}$ y un vector $\mathbf{c}_{n\times 1}$ con los cuales se forma el sistema lineal

$$\mathbf{A}\mathbf{x} = \mathbf{c}. \tag{6.76}$$

Uno pudiera escoger al menos las siguientes interpretacíones sobre como calcular $\mathbf{c}$ a partir de $\mathbf{A}$ y $\mathbf{x}$.

La matriz $\mathbf{A}$ puede intepretarse por colum-

La matriz $\mathbf{A}$ puede intepretarse por hileras

nas como

$$\begin{pmatrix} \mathbf{b}_1^T \\ \vdots \\ \mathbf{b}_n^T \end{pmatrix} \mathbf{x} = \mathbf{c}.$$

Es decir, $\mathbf{c}$ es resultado de la proyección de las hileras de $\mathbf{A}$ sobre el vector $\mathbf{x}$, tal como

$$\begin{pmatrix} \mathbf{b}_1^T\mathbf{x} \\ \vdots \\ \mathbf{b}_n^T\mathbf{x} \end{pmatrix} = \mathbf{c}.$$

como

$$\begin{pmatrix} \mathbf{a}_1 & \dots & \mathbf{a}_m \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} = \mathbf{c}.$$

Es decir, $\mathbf{c}$ es resultado de la combinación lineal de las columnas de $\mathbf{A}$, tal como

$$x_1\mathbf{a}_1 + \cdots + x_m\mathbf{a}_m = \mathbf{c}.$$

De una forma similar, dado $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_m)$, donde las hileras son interpretadas como características y las columnas como observaciones, en la operación $\mathbf{X}^T\phi$ hemos estado trabajando sobre el espacio de las características. En la sección anterior esto dio como resultado la selección de las características más importantes. Una forma alternativa es trabajar sobre el espacio de las observaciones. De forma similar esto resulta en la selección de las observaciones más sobresalientes para la descripción del modelo de regresión.

Hasta ahora, hemos considerado un modelo lineal de la forma

$$p(\mathbf{w}|\mathbf{X}) = \mathrm{Norm}_{\mathbf{w}}(\mathbf{X}^T\phi, \sigma^2\mathbf{I}). \tag{6.77}$$

Ahora consideramos el cambio de variable

$$\mathbf{\Phi} = \mathbf{X}\psi, \tag{6.78}$$

que representa que $\mathbf{\Phi}$ puede ser obtenida a partir de los datos en $\mathbf{X}$. Con este cambio de variable la predicción puede ser hecha en base al modelo

$$p(\mathbf{w}|\mathbf{X}, \theta) = \mathrm{Norm}_{\mathbf{w}}(\mathbf{X}^T\mathbf{X}\psi, \sigma^2). \tag{6.79}$$

Consecuentemente, los parámetros ψ y σ^2 pueden ser obtenidos mediante ML usando la expresión

$$\ln p(\mathbf{w}|\mathbf{X}, \theta) = -\frac{D}{2}\ln(2\pi) - D\ln\sigma - \frac{1}{2}(\mathbf{X}^T\mathbf{X}\psi - \mathbf{w})^T(\mathbf{X}^T\mathbf{X}\psi - \mathbf{w})/\sigma^2. \tag{6.80}$$

De donde se derivan expresiones para ψ, tal como

$$\psi = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{w}, \tag{6.81}$$

y para la covarianza σ^2, tal como

$$\sigma^2 = \frac{(\mathbf{X}^T\mathbf{X}\psi - \mathbf{w})^T(\mathbf{X}^T\mathbf{X}\psi - \mathbf{w})}{D}. \tag{6.82}$$

La expresión de regresión lineal Bayesiana equivalente, en términos de ψ está ahora dada por

$$p(\psi|\mathbf{X}, \mathbf{w}) = \frac{p(\mathbf{w}|\mathbf{X}, \psi)p(\psi)}{p(\mathbf{w}|\mathbf{X})}. \tag{6.83}$$

donde el prior pueden expresarse como

$$p(\psi) = \text{Norm}_{\psi}(0, \sigma^2\mathbf{I}). \tag{6.84}$$

Siguiendo un procedimiento similar al desarrollado en la §6.5, la expresión equivalente para $p(\psi|\mathbf{X}, \mathbf{w})$ está dada por

Para nuestro caso en particular, $\mathbf{a} = \mathbf{A}'\mathbf{w}$, $\mathbf{A} = \mathbf{\Sigma}'$, $\mathbf{b} = 0$ y $\mathbf{B} = \sigma_p^2\mathbf{I}$. De donde resulta que

$$p(\phi|\mathbf{X}, \mathbf{w}) = \text{Norm}_{\psi}(1/\sigma^2\mathbf{A}^{-1}\mathbf{X}\mathbf{w}, \mathbf{A}^{-1}), \tag{6.85}$$

con

$$\mathbf{A} = (1/\sigma^2\mathbf{X}\mathbf{X}^T\mathbf{X}\mathbf{X}^T + 1/\sigma_p^2\mathbf{I})^{-1}. \tag{6.86}$$

Así, para realizar inferencia, dado una nueva observación $\mathbf{x}^*$, uno podría estimar la probabilidad de su resultado $p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w})$, en términos de la expresión dual, usando la expresión

$$\begin{aligned} p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) &= \int p(w^*|x^*, \psi)p(\psi|\mathbf{X}, \mathbf{w})\phi, \\ &= \text{Norm}_{w^*}\left(\frac{1}{\sigma^2}\mathbf{x}^{*T}\mathbf{X}\mathbf{A}^{-1}\mathbf{X}^T\mathbf{X}\mathbf{w}, \mathbf{x}^{*T}\mathbf{X}\mathbf{A}^{-1}\mathbf{X}^T\mathbf{x}^* + \sigma^2\right). \end{aligned} \tag{6.87}$$

La representación anterior es importante pues el uso de productos punto la hace útil para la aplicación su generalización a modelos no lineales usando kernels.

Un ejemplo para un conjunto artificial de datos se muestra en la Figura 6.9. La figura fue obtenida utilizando el pseudo-código ilustrado en el Algoritmo 4. Note como hacia los extremos, donde no se tiene evidencia, la incertidumbre crece. En el algoritmo note la línea donde se calcula el valor de ψ, la cual proporciona estabilidad numérica al cálculo de la variable.

6.10. Regresión por Vectores Relevantes (RVM)

Las secciones anteriores han servido para desarrollar la base conceptual sobre la cual podemos construir el modelo de regresión RVM. Para ello, utilizamos una formulación dual sobre la regresión. Enseguida, se formulan variables auxiliares para cada una de las observaciones. Esto resulta en la selección de las observaciones más relevantes para la función base seleccionada.

Es decir, ahora se define el prior sobre los parámetros duales ψ tal como

$$p(\psi) = \prod_{i=1}^{M} \text{Stud}_{\psi_i}(0, 1, \nu). \tag{6.88}$$

Siguiendo un proceso similar al desarrollado en la §6.8 se obtiene la probabilidad de la verosimilitud como

$$p(\mathbf{w}|\mathbf{X}, \phi) \approx \max_{\mathbf{H}}\left(\int \text{Norm}_{\phi}(\mathbf{0}, \mathbf{X}^T\mathbf{X}\mathbf{H}^{-1}\mathbf{X}^T\mathbf{X} + \sigma^2\mathbf{I})\prod_{i=1}^{M}\text{Gam}_{h_i}(\nu/2, \nu/2)d\mathbf{H}\right), \tag{6.89}$$

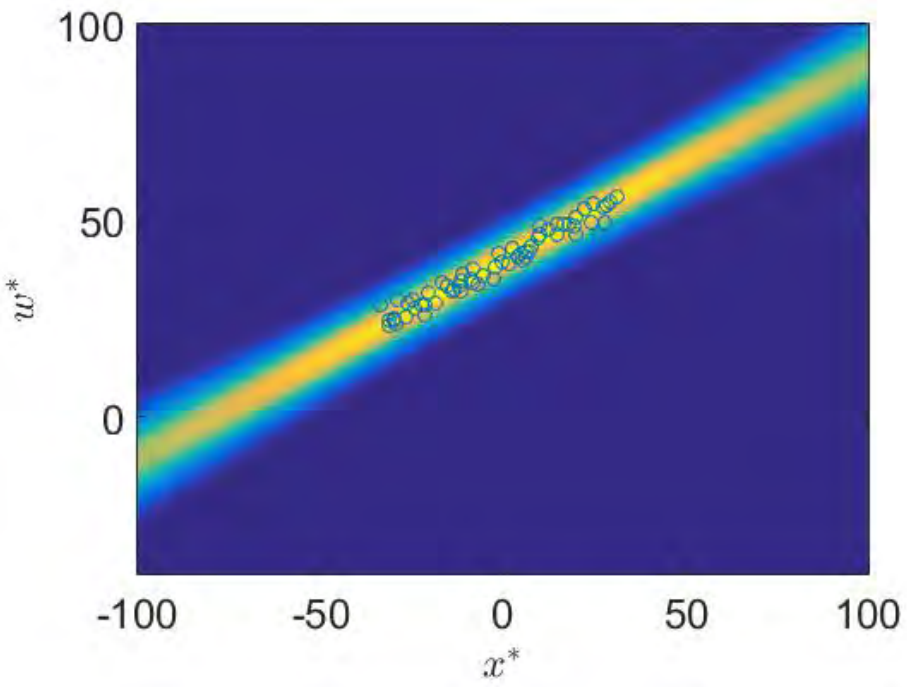


Figura 6.9: Ejemplo de regresión lineal Bayesiana usando la formulacón dual. Note como hacia los extremos, donde no se tiene evidencia, la incertidumbre crece.

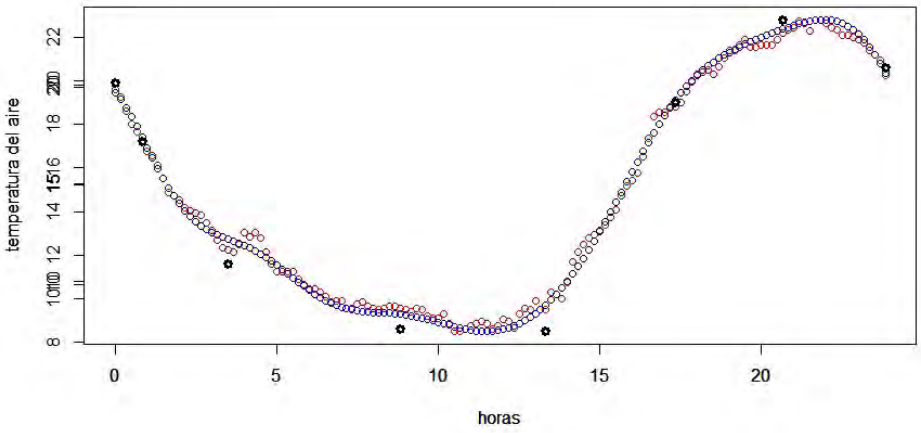


Figura 6.10: Resultado de la regresión no lineal usando RVM. Los datos se muestran como círculos rojos, la aproximación como circulos azules y los vectores de soporte en negro. Solo se requirieron ocho observaciones para aproximar la curva.

donde ahora las variables auxiliares en $\mathbf{H}$ están asociadas a las observaciones. En forma similar al Algoritmo 3, las variables auxiliares y la varianza se obtienen de forma iterativa usando las expresiones

$$h_i = \frac{(1 - h_i \Sigma_{i,i} + \nu)}{\mu_i^2 + \nu}, \text{ y } \sigma^2 = \frac{1}{\left(I - \sum_i (1 - h_i \Sigma_{i,i}) \right)}, \tag{6.90}$$

respectivamente. Entre iteraciones, se calcula la media μ y matriz de covarianza $\mathbf{\Sigma}$ tal como

$$\mu \leftarrow 1/\sigma^2 \mathbf{A}^{-1} \mathbf{X} \mathbf{w}, \text{ y } \Sigma \leftarrow \mathbf{A}^{-1}, \tag{6.91}$$

respectivamente. En donde $\mathbf{A}$ está dada por

$$\mathbf{A} \leftarrow 1/\sigma^2 \mathbf{X} \mathbf{X}^T + \mathbf{H}. \tag{6.92}$$

Para resolver las ecuaciones anteriores podemos utilizar la rutina **rvm** del paquete **kernlab** en **R**. La ilustración del resultado se muestra en la Figura 6.10.

Ejercicios

1. Para la base de datos correspondiente a la estación meterológica asignada obtén el modelo de regresión lineal para la relación entre la humedad relativa y la temperatura del aire usando

   - un modelo discriminativo. ¿Cuál es la relación entre el cambio de una y otra variable (ϕ_0)? ¿Cuál es el valor del cruce por cero (ϕ_1)? ¿Cuál es el valor de la desviación estándar (σ)?
   - un modelo generativo. ¿Cuáles son los valores de los priors de la pdf de $p(w|\theta)$ (μ_p y σ_p)?

2. Desarrolle un modelo de regresión lineal usando un modelo discriminativo y MAP para la obtención de los parámetros.

Llamada : $< \mathbf{H}, \sigma >\leftarrow$ `Hiperparámetros-Covarianza` $(\mathbf{X}, \mathbf{w})$
Entradas: Los datos $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$ y las etiquetas $\mathbf{w} = (w_1, \ldots, w_N)$.
Salidas : Los valores de los hiperparámetros $\mathbf{H}$ y la covarianza σ^2.

`// Calcula el valor de los parámetros`
$\phi \leftarrow (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{w}$;
`// Estima el valor inicial de la covarianza`
$\sigma^2 \leftarrow (\mathbf{w} - \mathbf{X}^T\phi)^T(\mathbf{w} - \mathbf{X}^T\phi)/M$;
`// Inicializa los hiperparámetros`
$\mathbf{H} \leftarrow \alpha\mathbf{I}_{N\times N}$;
`// Evalua la norma de` $\mathbf{H}$
$h^- \leftarrow |\mathbf{H}\|$;
`// Inicializa condición de paro`
$c \leftarrow$**false;**
while $(c ==$ **false**$)$ **do**
  `// Calcula el valor de` $\mathbf{A}$
  $\mathbf{A} \leftarrow 1/\sigma^2\mathbf{X}\mathbf{X}^T + \mathbf{H}$**;**
  `// Calcula el valor de` μ
  $\mu \leftarrow 1/\sigma^2\mathbf{A}^{-1}\mathbf{X}\mathbf{w}$**;**
  `// Calcula el valor de la covarianza`
  $\Sigma \leftarrow \mathbf{A}^{-1}$**;**
  `// Actualiza el valor de` $\mathbf{H}$
  $\mathbf{H}_{d,d} = \frac{(1-h_d\Sigma_{d,d}+\nu)}{\mu_d^2+\nu}$**;**
  `// Actualiza el valor de la covarianza` σ^2
  $\sigma^2 = \dfrac{1}{\left(D - \sum_d (1 - h_d\Sigma_{d,d})\right)}(\mathbf{w} - \mathbf{X}^T\mu)^T(\mathbf{w} - \mathbf{X}^T\mu)$**;**
  `// Evalua la norma de` $\mathbf{H}$
  $h^+ \leftarrow |\mathbf{H}\|$**;**
  `// Evalua condición de convergencia`
  $c \leftarrow |h^- - h^+|$**;**
end

Algorithm 3: Algoritmo para el cálculo de los hiperparámetros de los pesos para la combinación lineal y la covarianza del estimado de $\mathbf{w}$.

Llamada : $< \mathbf{H}, \sigma > \leftarrow$ `RegresiónLinealBayesianaDual` $(\mathbf{X}, \mathbf{w})$
Entradas: Los datos $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$ y las etiquetas $\mathbf{w} = (w_1, \ldots, w_N)$.
Salidas : La función $p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w})$.

```
// Calcula el valor de los parámetros. La siguiente formulación
   proporciona estabilidad numérica
```
$\psi \leftarrow \mathbf{X} \setminus (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{w}$;
```
// Estima el valor inicial de la covarianza
```
$\sigma^2 \leftarrow (\mathbf{w} - \mathbf{X}^T\mathbf{X}\psi)^T(\mathbf{w} - \mathbf{X}^T\mathbf{X}\psi)/N$;
```
// Evalua el valor de A
```
$\mathbf{A} \leftarrow 1/\sigma^2\mathbf{X}\mathbf{X}^T\mathbf{X}\mathbf{X}^T + 1/\sigma_p^2\mathbf{I}_{N\times N}$;
for $x \in \mathcal{R}$ **do**
  for $w^* \in \mathcal{R}$ **do**
    ```
    // Prepara predictor para evaluación
    ```
    $x^* \leftarrow (1, x)^T$;
    ```
    // Calcula la media
    ```
    $\overline{x} \leftarrow \frac{1}{\sigma^2}\mathbf{x}^{*T}\mathbf{X}\mathbf{A}^{-1}\mathbf{X}^T\mathbf{X}\mathbf{w}$;
    ```
    // Calcula la varianza
    ```
    $\overline{s^2} \leftarrow \mathbf{x}^{*T}\mathbf{X}\mathbf{A}^{-1}\mathbf{X}^T\mathbf{x}^* + \sigma^2$;
    ```
    // Evalua la probabilidad
    ```
    $p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) = \mathrm{Norm}_w(\overline{x}, \overline{s^2})$;
  end
end

Algorithm 4: Algoritmo para evaluar la probabilidad $p(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w})$ de una muestra $\mathbf{x}^*$ usando la formulación dual de la regresión Bayesiana.

Parte III

Aprendizaje Profundo

CAPÍTULO 7

Aprendizaje e Inferencia

Las NN son representaciones de una función no lineal mediante la cual ciertas entradas son transformadas en valores de salida. Regularmente son modeladas mediante un grafo acíclico (ver Figura 7.1), aunque hay un importante grupo de arquitecturas llamado redes neuronales recurrentes (RNN) donde se admiten ciclos[39]. Las redes están organizadas en una capa de entrada, una de salida y entre ellas varias llamadas ocultas.

7.1. Modelo de *Feed Forward*

Una neurona puede verse como recibiendo de la capa anterior un conjunto de señales, $\mathbf{x} = (x_1, \ldots, x_n)$, pesadas por un factor $\mathbf{w} = (w_1, \ldots, w_n)$. Dentro de la neurona estas entradas son sumadas, $w_1x_1 + \cdots + w_nx_n$, y un valor de sesgo b es adicionado. Como resultado, la neurona genera una señal $f(z)$ que denota su respuesta a las entradas. Es decir, la salida de una neurona está dada por $f(\mathbf{w}^T\mathbf{x} + b)$. Estas operaciones son repetidas en subsiguientes capas. Por ejemplo, en la Figura 7.1(a), las operaciones de *Feed Forward* tendrían la siguiente secuencia

$$z_j = \sum_{i \in \text{entradas}} w_{ij}x_i \rightarrow y_j = f(z_j) \rightarrow$$
$$z_k = \sum_{j \in H_1} w_{jk}y_j \rightarrow y_k = f(z_k) \rightarrow$$
$$z_l = \sum_{k \in H_2} w_{kl}y_k \rightarrow y_l = f(z_l), \quad (7.1)$$

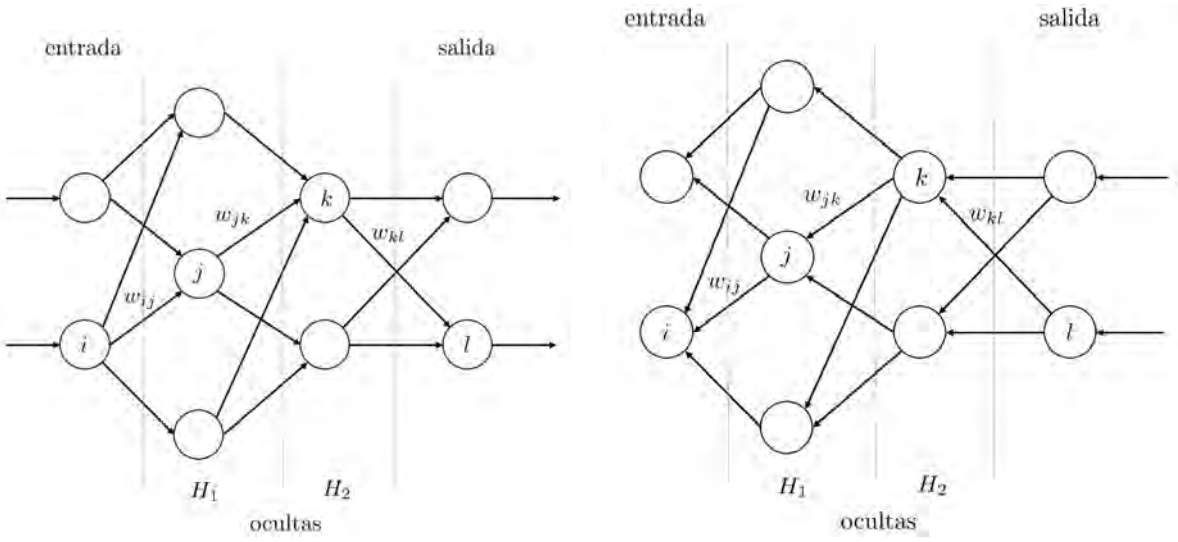
donde x_i corresponde a los valores de entrada y y_l corresponde a las salidas.

Las funciones de activación tienen el papel de realizar el mapeo a espacios no lineales. En su operación, emiten la salida de las neuronas en función de las entradas que se reciben. Son esenciales para el proceso de entrenamiento. Algunas de ellas incluyen la sigmoide, la RELU, la *leaky* RELU y la *MaxOut*[45]. Hace algunos años, la sigmoide, definida como

$$\sigma(x) = \frac{1}{1 + \exp(-x)}, \quad (7.2)$$

era muy común. Sin embargo, la función no está centrada en cero. Para aliviar este problema se sugirió la *tangente hiperbólica*, definida como

$$\tanh(x) = 2\sigma(2x) - 1, \quad (7.3)$$



(a) *Feed Forward* (b) *Back Propagation*

Figura 7.1: Representación gráfica de las etapas relacionadas a la evaluación de la función expresada en la red neuronal para una cierta entrada (a) y el cálculo de su derivada (b). En la ilustración no se muestra el nodo de sesgo en cada capa.

la cual está centrada en cero. Sin embargo, en ambos casos, durante el etapa de ajuste de los pesos por *back propagation*, cuando las activaciones se saturan en las colas, los gradientes se desvanecen. Actualmente, la función más utilizada es la unidad lineal rectificada (ReLU)[33], la cual se define como

$$\text{ReLU}(x) = \text{máx}(0, x). \tag{7.4}$$

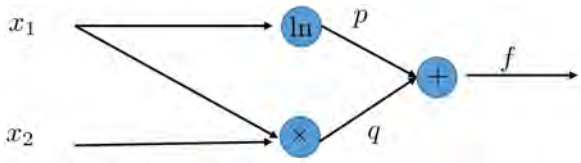
La función ha mostrado acelerar la convergencia y es sencilla de implementar.

Se ha mostrado que las NN son aproximadores universales, *i.e.*, dada una función $f(x)$ y una constante $\epsilon > 0$, existe una NN con una capa oculta $g(x)$ tal que $\forall x|f(x) - g(x)| < \epsilon$[28]. En ese sentido, uno pensaría que NN de pocas capas evitarían *overfitting* de las funciones. Sin embargo, se ha visto experimentalmente que NN de tres capas trabajan mejor que las de 2 (aunque no se ve un efecto tan dramático para 4, 5 o 6 capas). Actualmente se ha visto que la profundidad es un componente importante, donde 10 capas son comunes[60]. Para evitar *overfitting*, la función objetivo se regulariza. Esto es preferido a disminuir el número de neuronas.

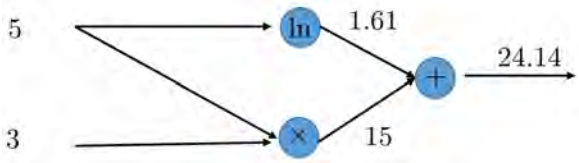
7.2. Entrenamiento por *Back Propagation*

Es generalmente aceptado que hay cuatro formas de obtener las derivadas de una ecuación: Hacer el cálculo manual, realizar el cálculo numérico, obtener el resultado simbólico, o hacer diferenciación automática[2]. Aquí revisamos este último método y nos familiarizamos con el algoritmo de aprendizaje denominado propagación hacia atrás o *back propagation*.

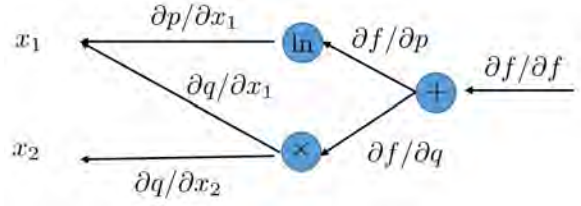
Dada una función $f(\mathbf{x})$, estamos interesados en obtener el gradiente $\nabla f(\mathbf{x})$. Por ejemplo, si $f(x, y)$, entonces $\nabla f = (\partial f/\partial x, \partial f/\partial y)^T$. Aquí estamos interesados en aplicar la regla de la cadena para resolver el problema. Por ejemplo, considere la siguiente expresión (tomada



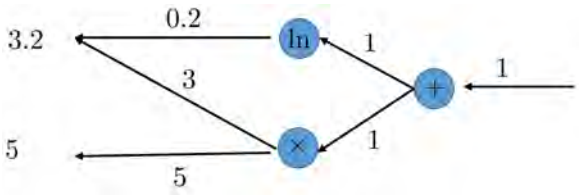
(a) Propagación hacia adelante símbolica



(b) Propagación hacia adelante numérica



(c) Propagación hacia atrás símbolica



(d) Propagación hacia atrás numérica

Figura 7.2: Ejemplo de obtención de la derivada de la función $f(x_1, x_2) = \ln x_1 + x_1 x_2$ mediante propagación hacia atrás. La función es descompuesta en términos de funciones sencillas, aplicando la regla de la cadena. Para la evaluación del gradiente se parte del final a contraflujo.

parcialmente de [2])

$$f(x_1, x_2) = \ln x_1 + x_1 x_2. \tag{7.5}$$

Esta expresión se puede descomponer en operaciones sencillas tales como sumas, multiplicaciones o derivadas de funciones de una variable, tal como

Cambio de variable	Derivada parcial
$p = \ln x_1$	$\partial p/\partial x_1 = 1/x_1$
$q = x_1 x_2$	$\partial q/\partial x_1 = x_2$ $\partial q/\partial x_2 = x_1$
$f = p + q$	$\partial f/\partial p = 1$ $\partial f/\partial q = 1$

(7.6)

La obtención del gradiente se puede realizar mediante la regla de la cadena. Utilizando las expresiones anteriores podemos llegar al siguiente resultado:

$$\nabla f(x_1, x_2) = \begin{pmatrix} \partial f/\partial x_1 \\ \\ \partial f/\partial x_2 \end{pmatrix} = \begin{pmatrix} \frac{\partial q}{\partial x_1}\frac{\partial f}{\partial q} + \frac{\partial p}{\partial x_1}\frac{\partial f}{\partial p} \\ \\ \frac{\partial q}{\partial x_2}\frac{\partial f}{\partial q} \end{pmatrix} \tag{7.7}$$

En una red neuronal, *back propagation* sirve para ajustar el valor de los pesos de la red. Para ello, se supone que se cuenta con un conjunto $\{\mathbf{X}, \mathbf{t}\}$ de observaciones $\mathbf{X}_{n\times m} = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$, con $\mathbf{x}_i = (x_1, \ldots, x_n)^T$ y sus correspondientes m valores de salida $\mathbf{t} = (t_1, \ldots, t_m)^T$. Si asumimos que nuestra función de costo está dada por la diferencia entre el valor de salida

de la red y el valor deseado, el error en cierto momento puede representarse como la suma de las diferencias al cuadrado, tal como

$$E = \sum_{l=1}^{m} (y_l - t_l)^2. \tag{7.8}$$

Utilizando como ilustración la arquitectura de red presentada en la Figura 7.1, la propagación del error puede describirse como

$$\begin{aligned} \frac{\partial E}{\partial y_l} = 2(y_l - t_l) \rightarrow \frac{\partial E}{\partial z_l} = \frac{\partial E}{\partial y_l} \cdot \frac{\partial y_l}{\partial z_l} \rightarrow \\ \frac{\partial E}{\partial y_k} = \sum_{l \in \text{salida}} \frac{\partial z_l}{\partial y_k} \frac{\partial E}{\partial z_l} \rightarrow \frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial z_k} \rightarrow \\ \frac{\partial E}{\partial y_j} = \sum_{k \in H_2} \frac{\partial z_k}{\partial y_j} \frac{\partial E}{\partial z_k} \rightarrow \frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial z_j}, \end{aligned} \tag{7.9}$$

donde recordamos que $y = f(z)$ y la derivada $\frac{\partial y}{\partial z}$ puede obtenerse evaluando la derivada de $f(z)$. Igualmente, notamos que $\frac{\partial z_l}{\partial y_k} = w_{kl}$ indica que los pesos son iguales a la derivada entre las entradas a los nodos y las salidas de nodos en capaz anteriores. Los pesos se modifican considerando como varía el error junto con ellos. Es decir, evaluando la expresión

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial z_k} \cdot \frac{\partial z_k}{\partial w_{jk}}, \tag{7.10}$$

donde el término $\dfrac{\partial z_k}{\partial w_{jk}}$ puede expresarse como

$$\frac{\partial z_k}{\partial w_{jk}} = \frac{\partial}{\partial w_{jk}} \left(\sum_{r=1}^{n} w_{rk} y_r \right) = y_j, \tag{7.11}$$

puesto que en la sumatoria solo un término depende de w_{jk}. De esta forma, los pesos son modificados usando la ecuación

$$w_{jk}^{(+)} = w_{jk}^{(-)} + \alpha \frac{\partial E}{\partial w_{jk}} = w_{jk}^{(-)} + \alpha y_j \frac{\partial E}{\partial z_k}, \tag{7.12}$$

para una constante de aprendizaje α.

CAPÍTULO 8

Redes Neuronales Convolucionales

Desde hace algunos años, el aprendizaje profundo han tomado gran relevancia por su nivel de desempeño en una variedad de tareas de reconocimiento de patrones. Especialmente notable han sido los resultados obtenidos en el *Imagenet Large Scale Visual Recognition Challenge*(ILSVRC), donde según algunas medidas han logrado resultados superiores a los humanos en lectura de signos de tráfico[14], reconocimiento de rostros[6] o interpretación de imágenes retinales[43]. La ventaja esencial de las CNN sobre otros modelos de detección es que la selección de características se realiza de forma automática. Asimismo, se ha mostrado que, con respecto a las redes neuronales (NN) tradicionales, el número de capas es un parámetro muy importante. Esto se traduce en que es normal que el número de parámetros por ajustar ronde en los millones, el tiempo de entrenamiento ronde en las semanas y el número de muestras para el entrenamiento este en el orden de las decenas de millones. Ciertamente, esto comienza a colocar la disciplina más allá de la computadora de escritorio normal. Sin embargo, los resultados obtenidos parecen apuntar a que continuará habiendo un fuerte impulso en esta dirección.

Algunas arquitecturas conocidas incluyen: LeNet, la primera CNN, desarrollada para identificar dígitos por Yan LeCun en los 1990's[34]; AlexNet, la CNN que las popularizó con su desempeño en *Imagenet Large Scale Visual Recognition Challenge*(ILSVRC) en 2012[32]; ZFNet, la ganadora en el ILSVRC 2013, donde esencialmente se afinaron los parámetros de AlexNet[72]; GoogleNet, la ganadora del ILSVRC 2014 reduciendo el número de parámetros[63]; VGGNet, la cual con sus 16 capas CONV/FC y solo ejecutaba convoluciones de 3×3 y sumarizaciones 2×2, mostró que la profundidad juega un rol primordial en las CNN[60]; ResNet, con 138M de parámetros, resultó la ganadora del ILSVRC 2015[26]; de forma interesante, en el 2016 los ganadores ensamblaron seis modelos, por lo que se puede decir que no hubo avances significativos en las arquitecturas propuestas[52].

8.1. Definición de la Arquitectura

Las CNN son una arquitectura de red neuronal especializada en el trabajo con imágenes. Por su origen, comparten con las NN algunos aspectos fundamentales. Por ejemplo, en ambos casos, el aprendizaje de los pesos es mediante *back propagation*. En ambos casos una neurona recibe entradas que son pesadas mediante un producto punto y con el resultado se realiza una transformación no lineal. Asimismo, en ambos casos hay una función de pérdida diferenciable. En una CNN las neuronas son agrupadas en capas tridimensionales (que tienen una anchura,

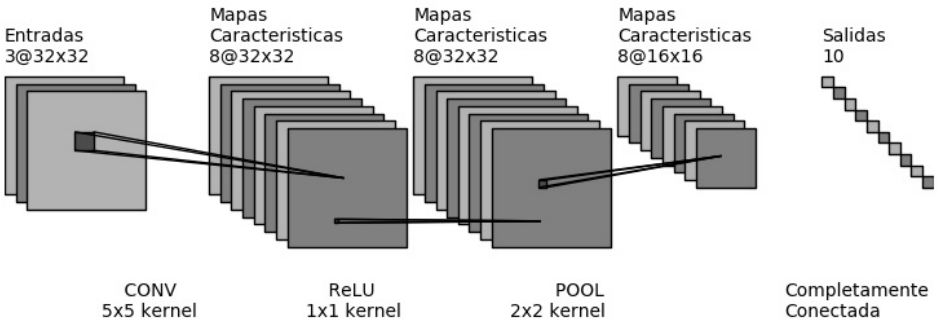


Figura 8.1: Ejemplo de arquitectura de una CNN con CONV→ ReLU → POOL. Una imagen de color con 32×32 pixeles de resolución es convolucionada con 8 filtros. Enseguida, este banco de imágenes pasa por una función no lineal ReLU. Luego, una función de POOL extrae las respuestas más grandes reduciendo la resolución de los datos espaciales a la mitad. Figura creada con `draw_convnet` de Gavin Weiguang Ding.

altura y profundidad) que se conectan una tras otra mediante una función diferenciable que puede requerir o no parámetros. Por ejemplo, una imagen de $m \times n \times 3$ requiere una capa de entrada de $3mn$ neuronas. Las neuronas en las capas siguientes serán conectadas a una pequeña región de la capa anterior, no a todas las neuronas. Al final, la capa de salida tendrá dimensiones $1 \times 1 \times K$, correspondiendo a las K clases de interés.

Hay tres tipos de capas en una CNN: Convoluciones (CONV), Sumarizado (POOL) y Completamente Conectadas (FC). Por ejemplo, supóngase que se tiene una CNN que toma imágenes de $32 \times 32 \times 3$, pretende distinguir entre 10 clases y tiene la siguiente arquitectura: INPUT-CONV-RELU-POOL-FC. Las capas tienen la siguiente interpretación (ver Figura 8.1):

- INPUT. Es una capa de $32 \times 32 \times 3$, la cual tendrá la imagen de color.
- CONV. Esta capa calculará la salida de las neuronas en una región de la imagen de entrada. En cada región se calcula un producto punto entre los pesos y una pequeña región conectada a la entrada. Si se tienen 8 filtros, se tendrá un volumen de $32 \times 32 \times 8$ unidades.
- RELU. En esta capa se aplica la función de activación $\max(0, x)$, elemento por elemento. Esto deja el volumen en $32 \times 32 \times 8$ unidades.
- POOL. En esta capa se hace una sumarización sobre la parte espacial de la imagen, seleccionado solo algunos elementos de ella. Por ejemplo, puede ser que el volumen resultante sea de $16 \times 16 \times 8$.
- FC. En esta capa se calculan los scores resultando en un volumen de tamaño $1 \times 1 \times 10$.

En este ejemplo, las capas CONV/FC harán transformaciones que son función del volumen de entrada, los pesos y los sesgos. Las capas son entrenadas con gradiente descendente. Las capas RELU/POOL implementan una función fija.

Las CNN pueden construirse con capas CONV, POOL, RELU y FC. Por ejemplo

$$\text{INPUT} \to [[\text{CONV} \to \text{RELU}] \times N \to \text{POOL?}] \times M \to [\text{FC} \to \text{RELU}]K \to \text{FC}, \quad (8.1)$$

donde $\times$ indica repetición, POOL? una capa de sumarización opcional, $N \geq 0$ (usualmente $N \leq 3$), $M \geq 0, K \geq 0$ (usualmente $K < 3$). Por regla de dedo, se prefiere muchos CONV con filtros pequeños que un CONV con un campo receptivo grande. Esto debido a que entre cada CONV hay normalmente una función no lineal.

Capa CONV. Durante la etapa de procesamiento de la red, las entradas con convolucionadas con filtros sobre las dimensiones de anchura y altura. Esto genera un mapa de activación en 2D. La convolución se realiza entre las entradas y los pesos. La CNN aprenderá los filtros más apropiados que se activarán cuando alguna característica, tal como un contorno en alguna orientación o una región de un cierto color esté presente en la primera capa, o algunas características más complejas en capas posteriores de la red. Cada filtro aplicado produce mapas de activación bidimensionales separados. La extensión de la conectividad es llamado campo receptivo de la neurona y se regula con un hiperparámetro. Las conexiones son locales en espacio pero siempre completas sobre la profundidad del volumen de entrada.

Capa POOL. La función de la capa POOL es reducir gradualmente el tamaño espacial de la representación, reducir el número de parámetros y evitar *overfitting*. En común insertar una capa de sumarización entre capas de convolución. La capa de sumarización usa la operación máx. La forma más común es aplicarla sobre una vecindad de 2×2 con un paso de 2, logrando con ello descartar el 75 % de las activaciones.

8.2. Clasificación Lineal

Una red neuronal funciona como una transformación no lineal de las entradas. La expectativa es que en ese nuevo espacio, uno pueda distinguir las clases mediante un hiperplano. En las arquitecturas actuales, las opciones son utilizar ya sea una SVM multiclase o un clasificador Softmax. Aquí exploramos ambos.

Primero, partimos del supuesto de que si $\mathbf{x}_i \in \mathcal{R}^D$, para $i = 1, \ldots, N$, representa una imagen que tiene una etiqueta $y_i \in \{1, \ldots, K\}$, entonces $\mathbf{f}$ podría ser definida como una función que mapea de pixeles al *score* de pertenencia a la clase,

$$\mathbf{f} : \mathcal{R}^D \to \mathcal{R}^K. \quad (8.2)$$

Un ejemplo de esta función podría ser un clasificador lineal, definido como

$$\mathbf{f}(\mathbf{x}_i, \mathbf{W}) = \mathbf{W}\mathbf{x}_i + \mathbf{b}, \quad (8.3)$$

donde $\mathbf{W}_{K \times D}$ es una matriz de pesos y $\mathbf{b}_{K \times 1}$ un vector de sesgo. En este sentido, las hileras de $\mathbf{W}$ pueden ser interpretadas como la descripción de las líneas que separan las clases. Abusando de la notación, antes de proceder a realizar las operaciones de clasificación podríamos

Figura 8.2: Algunos ejemplos contenidos en la base de datos de MNIST.

incluir una columna de unos a $\mathbf{W}$ y un uno a cada vector $\mathbf{x}_i$ de tal forma de utilizar la representación

$$\mathbf{f}(\mathbf{x}_i, \mathbf{W}, \mathbf{b}) = \mathbf{W}\mathbf{x}_i. \tag{8.4}$$

8.2.1. Clasificador SVM

Con respecto al problema de clasificación multiclase usando SVM, el criterio de asignación es que el *score* de la clase correcta debe ser mayor que el resto por al menos un valor de margen Δ[19]. Si definimos la función de *score* como

$$s_j = f(\mathbf{x}_i, \mathbf{w}_j) = \mathbf{w}_j^T \mathbf{x}_i, \tag{8.5}$$

donde $\mathbf{w}_j$ corresponde a la j-ésima línea de discriminación, o j-ésima columna de $\mathbf{W}^T$, entonces la función de pérdida SVM para la i-ésima clase podría definirse como

$$\begin{aligned} L_i &= \sum_{j \neq y_i} \text{máx}(0, s_j - s_{y_i} + \Delta), \\ &= \sum_{j \neq y_i} \text{máx}(0, \mathbf{w}_j^T \mathbf{x}_i - \mathbf{w}_{y_i}^T \mathbf{x}_i + \Delta), \end{aligned} \tag{8.6}$$

conocida como *hinge loss function*. Por ejemplo, si se tuvieran tres clases con scores $\mathbf{s} = (13, -7, 11)$, un valor mínimo de margen $\Delta = 10$, y la primera fuera la clase verdadera, la función de pérdida sería

$$\begin{aligned} L_1 &= \text{máx}(0, s_2 - s_1 + \Delta) + \text{máx}(0, s_3 - s_1 + \Delta), \\ &= \text{máx}(0, -7 - 13 + 10) + \text{máx}(0, 11 - 13 + 10) = 21. \end{aligned} \tag{8.7}$$

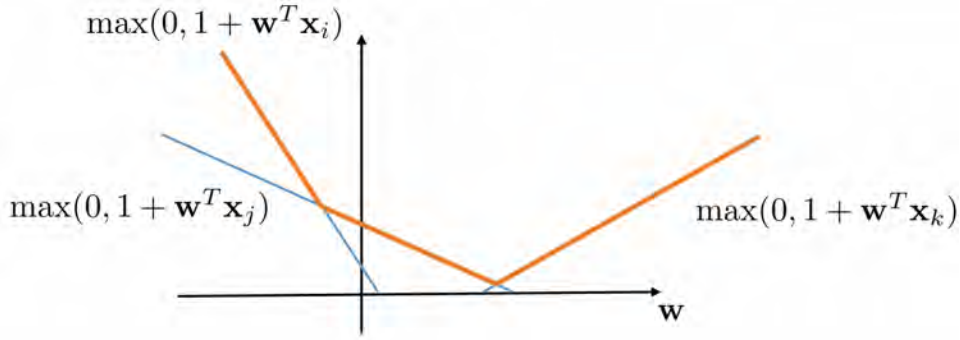


Figura 8.3: Clasificación mediante SVM multiclase. El clasificador SVM optimiza la llamada *hinge loss function*. Para el conjunto de los parámetros bajo consideración esto resulta en una función cóncava.

Dada la interpretación geométrica de la ecuación de la línea recta, hay un factor de escala que hay que ajustar. En SVM multiclase, se quiere minimizar la magnitud de $\mathbf{W}$, $R(\mathbf{W})$, definida como

$$R(\mathbf{W}) = \sum_{k=1}^{K}\sum_{l=1}^{D} w_{kl}^2. \tag{8.8}$$

Así pues, la función de pérdida queda definida como (ver Figura 8.3)

$$L = \frac{1}{N}\sum_{i=1}^{N} L_i + \lambda R(\mathbf{W}), \tag{8.9}$$

donde λ es el multiplicador de Lagrange y N es el número de muestras. En la práctica, el valor de Δ se deja igual a uno y se deja el peso de la regularización al factor λ.

8.2.2. Clasificador Softmax

El clasificador Softmax es una generalización del clasificador binario de regresión logística para el caso multidimensional. La función de pérdida cambia a

$$L_i = -\log\left(\frac{\exp\left(f_{y_i}\right)}{\sum_{j=1}^{K}\exp\left(f_j\right)}\right) = -f_{y_i} + \log\sum_{j=1}^{K}\exp\left(f_j\right), \tag{8.10}$$

donde f_j es el j-ésimo elemento del vector $\mathbf{f} = (f_1, \ldots, f_K)^T$ de *scores* para las clases. La función de pérdida completa corresponde a (8.9).

8.3. Aspectos Estáticos de una CNN

Antes de proceder a entrenar la CNN es recomendable preprocesar los datos con la finalidad de centralizarlos y que su valor medio sea igual a cero y normalizar las entradas

para que su desviación estándar sea uno[35].

8.3.1. Preproceso de los Datos

Sea $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, el conjunto de datos, cuya dimensionalidad es D. El centrado requiere sustraer el valor medio, $\overline{\mathbf{x}}$, lo cual puede llevarse a cabo con la operación

$$\mathbf{X}_m = \mathbf{X} - \overline{\mathbf{x}}\mathbf{1}^T, \tag{8.11}$$

donde $\mathbf{1}$ es un vector de N dimensiones.

Para la normalización se requiere que las características tengan desviación estándar unitaria en cada una de las dimensiones. Para ello, sea $\sigma = (\sigma_1, \ldots, \sigma_D)$ el vector de $D \times 1$ con las desviaciones estándar sobre las características. En adición sea la representación de $\mathbf{X} = (\mathbf{u}_1, \ldots, \mathbf{u}_D)^T$ la representación de los renglones de $\mathbf{X}$. La operación de centralización y normalización sobre las características sería

$$\hat{\mathbf{u}} = \frac{\mathbf{u}_i - \overline{\mathbf{u}}_i}{\sigma_i}. \tag{8.12}$$

8.3.2. Inicialización de los Pesos

Si los pesos $\mathbf{W}$ se inicializan a ceros, las neuronas calcularían la misma salida, con lo cual tendrían los mismos gradientes y las mismas actualizaciones de parámetros. Hay varios esquemas de inicialización. Uno de ellos consiste en proporcionar un pequeño valor aleatorio con función de distribución de probabilidad Gaussiana que rompa la simetría. La distribución de las salidas de una neurona inicializada aleatoriamente tiene varianza que crece con el número de entradas. Lo conveniente es normalizar la varianza de cada neurona a uno escalando su vector de pesos por la raíz cuadrada del número de entradas. Así los pesos se inicializan con

$$\mathbf{W} = \frac{\text{matrix de } n \times n \text{ con ruido Gaussiano}}{\sqrt{n}}, \tag{8.13}$$

donde n es el número de entradas.

8.3.3. Regularización

La regularización controla la capacidad de la NN para evitar `overfitting`. Las opciones de regularización se aplican sobre (8.9) e incluyen[35, 30]:

Regularización Euclideana. Sigue la formulación $\frac{1}{2}\lambda w^2$, donde λ es el grado de la regularización. Su efecto es usar todas las entradas un poco, contrario a usar algunas pocas entradas mucho.

Regularización de Manhattan. Sigue la formulación $\lambda|w|$, en la práctica algunos de los pesos se hacen cero, haciendo una suerte de selección de características.

MaxNorm. Estas restricciones forzan a que la magnitud de $\mathbf{w}$ no sobrepase de un cierto valor absoluto. En la práctica, la actualización de los parámetros es hecha como de costumbre y entonces el vector de pesos es limitado para $\|\mathbf{w}\|_2 < c$, para valores típicos de c de 2 o 4.

Dropout. Durante entrenamiento, `dropout` mantiene una neurona activa con una probabilidad P (un hiperparámetro) o de otra forma la pone a cero.

El `dropout` solo de aplica en la etapa de entrenamiento. En esa etapa, el valor esperado es $xp + (1-p)0$. Sin embargo, ya en la etapa de pruebas se tiene que hacer $x \to px$ para mantener el mismo valor esperado.

En la práctica, es común usar una regularización L_2 cuya constante λ se calcula vía *cross-validation*.

8.4. Aspectos Dinámicos de una CNN

El entrenamiento de una CNN es muy largo y puede estar en el orden de días. Algunos aspectos que pueden ser monitoreados durante ese tiempo incluyen[35, 30]:

- Hay que verificar que se obtiene la pérdida esperada al inicializar con valores pequeños para los parámetros. Es mejor verificar el término de pérdida de los datos por separado. El incrementar la contribución del término de regularización se debería incrementar el valor de la pérdida. Conviene, por ejemplo, tomar un pequeño subconjunto de datos y verificar que se puede llegar a un costo cero, al hacer *overfitting* sobre esos datos.

- Durante el proceso conviene estar verificando el comportamiento, por ejemplo con gráficas que se actualicen en cada época.

- La función de pérdida es evaluada en subconjuntos de datos de entrenamiento, por lo que puede mostrar oscilaciones. En todo caso, el cociente entre el error entre el conjunto de entrenamiento y el conjunto de validación puede ser un indicador. Por ejemplo, si hay una variación pequeña sobre el desempeño cuando analizamos el conjunto de validación puede significar *overfitting* en los datos de entrenamiento.

- El cociente entre el valor de las actualizaciones de los pesos y la magnitud de los gradientes debe ser aproximadamente 10^{-3}. Si es mayor, la tasa de aprendizaje puede ser muy alta. Si es menor, la tasa de aprendizaje puede ser muy baja.

- Una mala inicialización puede reducir o parar el proceso de aprendizaje. Una forma de hacer el diagnóstico de esta situación es graficar los histogramas de activación o gradiente para todas las capas de la NN.

- Las características de la primera capa pueden visualizarse y muestran las clases que se desea aprender.

dígitos	0	1	2	3	4	5	6	7	8	9
muestras	1,272	1,389	1,273	1,316	1,285	1,104	1,279	1,258	1,186	1,239

Cuadro 8.1: Distribución de dígitos en la muestra utilizada para probar la CNN LeNet con la base de datos de MNIST

8.5. Ejemplo de MNIST

MNIST es el ejemplo *Hola Mundo* para CNN. MNIST se compone de decenas de miles de imágenes correspondientes a dígitos numéricos escritos a mano (ver Figura 8.2). La base de datos puede dividirse cómodamente en decenas de miles de imágenes para entrenamiento, prueba y validación. Cada imagen tiene una resolución de 28×28 pixeles, cuyo valor para pixel es un número real entre cero y uno.

Para este ejercicio utilizamos la CNN LeNet, la cual se define como[34]

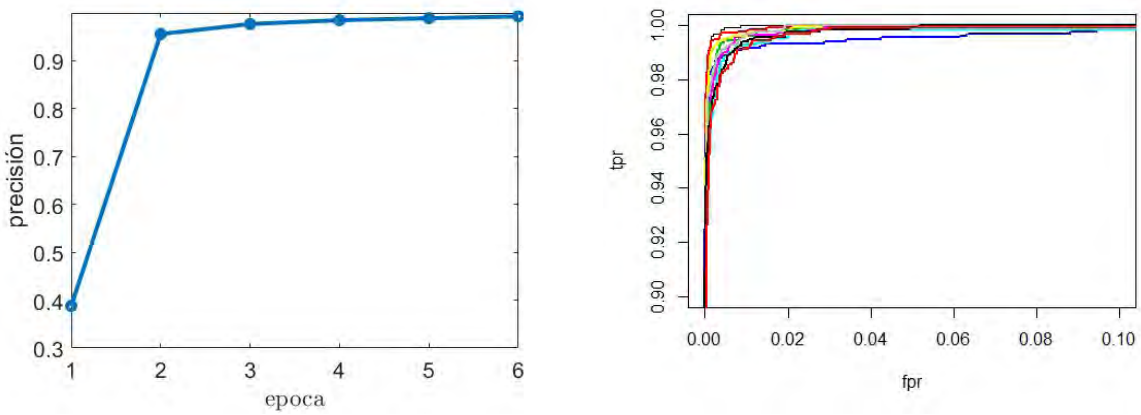
$$\text{INPUT} \to [[\text{CONV} \to \text{TANH}] \to \text{POOL}] \times 2 \to [\text{FC} \to \text{TANH}]10 \to \text{FC}, \tag{8.14}$$

donde TANH es la tangente hiperbólica, como en (7.3). Para este ejercicio en particular, utilizamos 42,000 dígitos de la base de datos MNIST. De ellos, escogimos el 70 % (29,399) de las imágenes para entrenar y el 30 % (12,601) imágenes para probar. El tiempo para entrenar una red de este tipo es de 196 segundos (3 minutos y 16 segundos) en el CPU. El uso de una GPU puede reducir en órdenes de magnitud el tiempo de procesamiento. Para el entrenamiento se utilizaron grupos de 100 imágenes escogidos aleatoriamente para calcular el gradiente, en una técnica conocida como *stochastic gradient descend*[7]. La tasa de aprendizaje α fue de 0.05. Para este ejemplo, se realizaron 6 iteraciones sobre los datos de entrenamiento. El incremento de la precisión con cada iteración se muestra en la Figura 8.4(a). Para entrenamiento, la distribución de dígitos sigue la distribución que se muestra en la Tabla 8.1, la cual exhibe homegeneidad. La evaluación del proceso de entrenamiento se muestra en la Figura 8.4(b)

En un problema de reconocimiento de patrones, mucho se puede entender de los datos por clasificar utilizando visualización de datos. Sin embargo, la visualización de datos multidimensionales es un reto. Para poder tener una intuición de esto, considere a cada pixel como representando una característica dentro de un espacio multidimensional y al valor del pixel como una posición dentro de esta dimensión. Esto resultará al conjunto de datos como un subespacio dentro del espacio de las todas las imágenes posibles. El *t-Distributed Stochastic Neighbor Embeeding* (t-SNE)[41] es una técnica de visualiación de datos que preserva su topología, *i.e.*, trata de tener los mismos vecinos, independientemente de la dimensionalidad del espacio donde se visualice. De su representación visual (ver Figura 8.5), uno podría concluir que MNIST es una base de datos estructurada donde la distinción entre clases permite un alto nivel de desempeño de los clasificadores.

Ejercicios

1. Instalar Caffe, MxNet, TensorFlow o alguna otra plataforma de CNN para implementar el ejemplo de MNIST para la clasificación de imágenes de dígitos escritos a mano.



(a) (b)

Figura 8.4: MNIST usando la CNN LeNet. En (a) se muestra el incremento de la precisión por epoca de entrenamiento para la base de datos MNIST usando la CNN LeNet. Luego, en (b), se muestra la curva ROC para cada dígito clasificado. El promedio del área bajo la curva es 0.9997

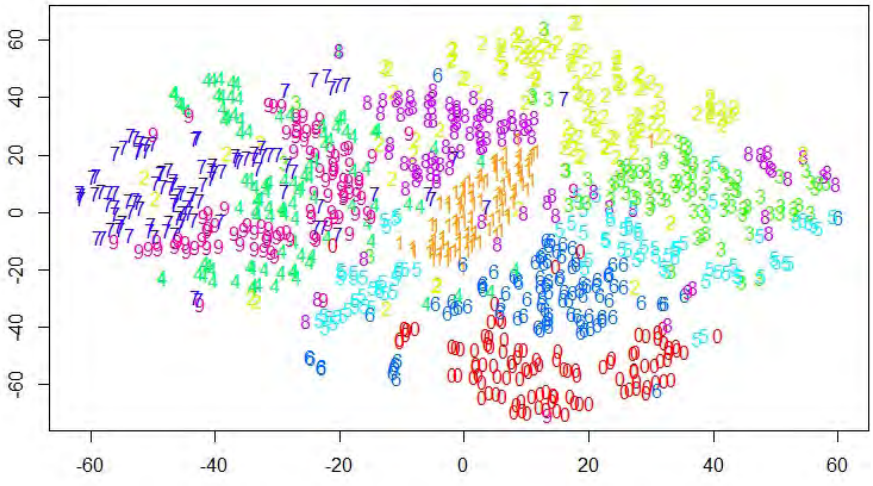


Figura 8.5: Visualización t-SNE de la base de datos del MNIST. Aquí se representa una muestra aleatoria de 1,000 dígitos de la base de datos del MNIST visualizados mediante t-SNE.

2. Explorar algunas arquitecturas como la de AlexNet o VGG para la detección de objetos.

3. Comparar el uso de CPU contra GPU en el tiempo requerido para entrenar un modelo de CNN.

CAPÍTULO 9

Entrenamiento de Redes Neuronales Convolucionales

La tendencia en el aprendizaje profundo es hacia arquitecturas más complejas, tal vez con un mayor número de capaz pero al mismo tiempo con un menor número de parámetros. Esto trae como consecuencia la necesidad de tratar de forma explícita el *overfitting* mediante esquemas de regularización, paro temprano y esquemas de *dropout*. El incremento la dimensionalidad del espacio de parámetros trae como consecuencia la necesidad de ganar intuición sobre los mecanismos de optimización de parámetros. En este capítulo repasamos algunos temas relacionados con esta problemática. Revisamos la arquitectura de *Inception* como una forma de reducir el número de parámetros, la regularización L_1 y L_2 y algunos mecanismos de optimización y la dificultad que enfrentan.

9.1. Modelo de *Inception*

Inception obtuvo el primer lugar en el *ImageNet Large-Scale Visual Recognition Challenge 2014*[63]. Mejoraron el uso de recursos dentro de la red al incrementar la profundidad y anchura de la red manteniendo los recursos computacionales constantes. La propuesta de Inception explora el asunto de tener una arquitectura que use parámetros extra dispersos pero explote el hardware actual y utilice cálculo en matrices densas. Szegedy *et al.*[63] no están seguros de la razón por la cual la red tiene éxito e invitan a un mayor análisis y verificación. Su s diseños de arquitectura se basaron en un principio Hebbiano, donde un incremento en la plasticidad sináptica surge por la repetida y persistente estimulación de la célula pos-sináptica por la célula pre-sináptica o en pocas palabras, las *células que disparan juntas se alambran juntas*[40].

Una forma particular de Inception con 22 capas de profundidad fue llamada GoogLeNet. Con 1.5 mil millones de multiplicaciones y sumas, Inception usa 7 millones de parámetros, 9× menos parámetros que AlexNet[32]. Inception surge de la motivación de incrementar el número de capas o la anchura de cada capa con la finalidad de mejorar el desempeño. Sin embargo, esto tiene el potencial de generar sobre-ajuste y la necesidad de incrementar los recursos computacionales necesarios. En operación la solución de GoogLeNet usa R-CNN (Regions with Convolutional Neural Networks) [22] en donde se descompone el problema de la detección en una parte de propuesta de regiones seguida de un clasificador CNN para identificar categorias en ellas.

Inception de basa en el concepto de red dentro de red (Lin *et al.*[38]), la cual puede ser vista como una convolución 1×1 seguida de activación lineal rectificada [32]. Las convoluciones 1×1 funcionan como sigue. Sea la salida de una capa de convolución (N, F, H, W), donde N es el tamaño del *batch*, F es el número de filtros de convolución, H, W es la altura y anchura. Las convoluciones 1×1 sirven para incrementar o reducir (en el caso de Inception) la dimensionalidad en la dirección de las características. Por ejemplo, sea una entrada de 256 niveles de profundidad seguida de una convolución de 4×4 con 128 niveles de profundidad. Si antes agregamos una convolución de 1×1 con 64 niveles de profundidad entonces los 256 niveles de profundidad de la entrada se colapsarán, volviendo computacionalmente más eficiente la convolución con los filtros de 4×4. En Inception las convoluciones 1×1 son seguidas de activaciones de rectificación lineal.

El modelo original de inception se presenta en la Figura 9.1 (a). En la conocida como Inception v3, capas más profundas se han especializado[64]. Su diseño se presenta en la Figura 9.1(b)-(d). Esta nueva versión de inception usa 5 mil millones de multiplicaciones y sumas y usa menos de 25 millones de parámetros. Algunos principios que llevaron a la nueva arquitectura incluyen los siguientes: a) evitar cuellos de botella en la representación, especialmente en las primeras capas de la red; b) las representaciones de altas dimensiones son más fáciles de procesar localmente en la red; c) la agregación espacial puede ser hecha en embebidos en dimensiones bajas sin pérdida de capacidad de representación; y d) es conveniente balancear la anchura y profundidad de la red.

9.2. Regularización

La regularización es el proceso mediante el cual se introduce información adicional a un problema que no está bien definido o restricciones que evitan que ocurra *overfitting* en la solución. Por ejemplo, en el caso de la visión por computadora, se tiene el problema de que se quiere recuperar la información de estructura de un escenario tridimensional. Sin embargo, lo que se tiene son proyecciones bidimensionales, por lo que en sí, el problema no tiene solución. En este caso la introducción de restricciones de suavidad o continuidad puede hacer que el problema tenga solución. En el caso de aprendizaje maquinal es común buscar un mapeo entre entradas y salidas. Puede llevar a darse el caso, por ejemplo en el caso de las redes neuronales, donde el número de parámetros sea muy grande y la solución tienda a hacer sobre ajuste de los datos de entrenamiento. En este caso igualmente, se puede plantear alguna restricción de costo sobre la magnitud de los parámetros que permita obtener soluciones con mayor capacidad de generalización.

Así, dado un mapeo $J(\theta; \mathbf{X}, \mathbf{y})$, la regularización se ve como la operación

$$\hat{J}(\theta; \mathbf{X}, \mathbf{y}) = J(\theta; \mathbf{X}, \mathbf{y}) + \alpha\Omega(\theta), \tag{9.1}$$

donde θ es el conjunto de parámetros, α pasa la contribución de la información adicional en el problema, y Ω es la función que define las restricciones adicionales. En lo que sigue del documento estudiamos algunos esquemas de regularización comúnmente utilizados en redes neuronales profundas.

Considere la regularización basado en la norma L_1,

$$\hat{J}(\theta; \mathbf{X}, \mathbf{y}) = J(\theta; \mathbf{X}, \mathbf{y}) + \alpha \parallel \theta \parallel, \tag{9.2}$$

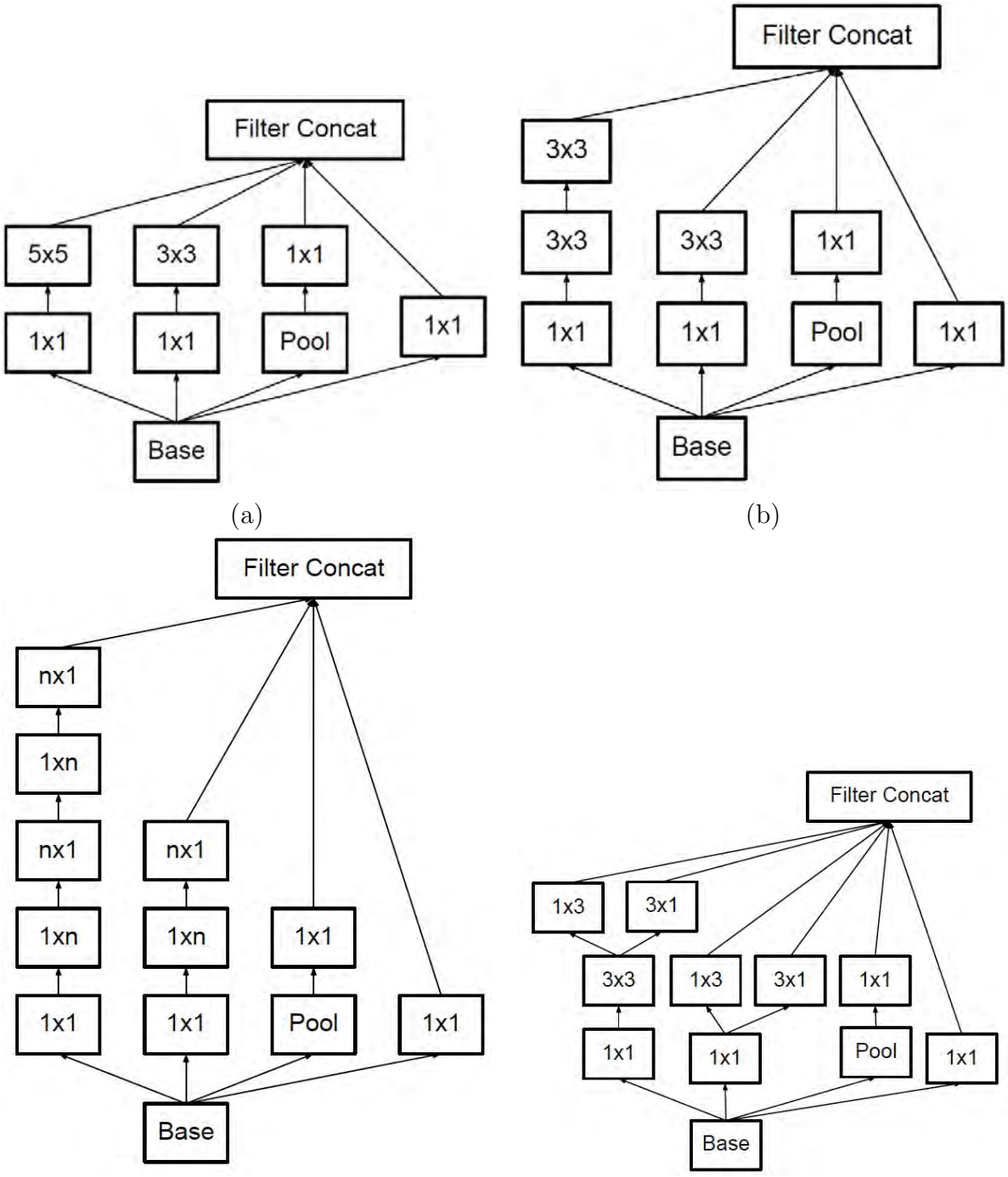
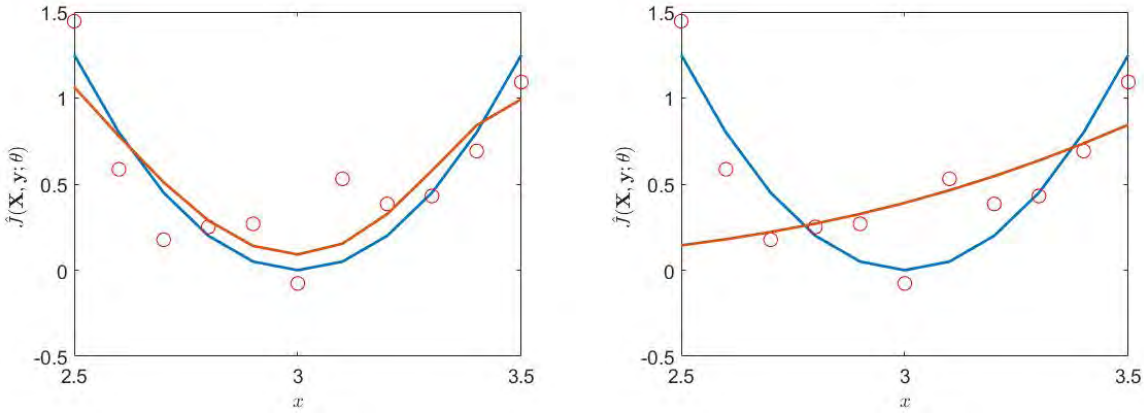


Figura 9.1: Representación gráfica del modelo de inception. Imágenes obtenidas de Szegedy *et al.*[64].



(a) (b)

Figura 9.2: Solución por mínimos cuadrados con regularización L_1. Sobre un modelo cuadrático subyacente se definen 11 puntos con ruido. Aplicando el mecanismo de minimización de pérdida con regularización L_1 se tiene la solución (a) para $\alpha = 0.001$, donde el modelo ajusta razonablemente bien, y $\alpha = 100$, donde el modelo requiere mayor complejidad.

donde el gradiente está dado por

$$\nabla_\theta \hat{J}(\theta; \mathbf{X}, \mathbf{y}) = \nabla_\theta J(\theta; \mathbf{X}, \mathbf{y}) + \alpha \text{ signo}(\theta), \text{ para } \| \theta \| > 0. \tag{9.3}$$

Mientras que para la normal L_2 se tiene la función

$$\hat{J}(\theta; \mathbf{X}, \mathbf{y}) = J(\theta; \mathbf{X}, \mathbf{y}) + \frac{1}{2}\alpha\theta^T\theta, \tag{9.4}$$

donde el gradiente está dado por

$$\nabla_\theta \hat{J}(\theta; \mathbf{X}, \mathbf{y}) = \nabla_\theta J(\theta; \mathbf{X}, \mathbf{y}) + \alpha\theta. \tag{9.5}$$

Esto puede apreciarse en la Figura 9.3.

9.2.1. Ilustración del Proceso de Regularización

Supóngase que se tiene un conjunto de puntos (x_i, y_i) para $i = 1, \ldots, m$, y un modelo polinomial que se quiere ajustar a ellos, tal como

$$\begin{aligned} y_i &= \theta_0 + \theta_1 x_i + \cdots + \theta_n x_i^n = \sum_{k=0}^{n} \theta_k x_i^k = \mathbf{x}_i^T \theta, \\ \mathbf{y} &= \mathbf{X}\theta. \end{aligned} \tag{9.6}$$

donde $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_m)^T$ y $\mathbf{y} = (y_1, \ldots, y_m)^T$ corresponden al conjunto de los puntos embebidos en el modelo. La función de pérdida con regularización L_1 podría plantearse como

$$\hat{J}(\theta; \mathbf{X}, \mathbf{y}) = \frac{1}{2} (\mathbf{X}\theta - \mathbf{y})^T (\mathbf{X}\theta - \mathbf{y}) + \alpha \| \theta \|, \tag{9.7}$$

con un gradiente dado por la función

$$\nabla_\theta \hat{J}(\theta; \mathbf{X}, \mathbf{y}) = \mathbf{X}^T (\mathbf{X}\theta - \mathbf{y}) + \alpha \texttt{ signo } (\theta), \tag{9.8}$$

donde la función `signo` se aplica a cada componente. Por su lado, en el caso de regularización L_2, la función de pérdida con regularización podría plantearse como

$$\hat{J}(\theta; \mathbf{X}, \mathbf{y}) = \frac{1}{2} (\mathbf{X}\theta - \mathbf{y})^T (\mathbf{X}\theta - \mathbf{y}) + \frac{1}{2}\alpha\theta^T\theta, \tag{9.9}$$

con un gradiente dado por la función

$$\nabla_\theta \hat{J}(\theta; \mathbf{X}, \mathbf{y}) = \mathbf{X}^T (\mathbf{X}\theta - \mathbf{y}) + \alpha\theta. \tag{9.10}$$

En ambos casos, el valor óptimo de los parámetros podría encontrarse por descenso por gradiente mediante el esquema de iteración

$$\theta^+ = \theta^- + \rho\nabla_\theta \hat{J}(\theta; \mathbf{X}, \mathbf{y}). \tag{9.11}$$

En ambos casos, cuando el valor de $\alpha = 0$ se tiene el esquema normal sin restricciones, mientras que cuando el valor de α se incrementa se tienen soluciones más suaves. Por ejemplo, considere la Figura9.5, donde se presenta un ejemplo de regularización para un modelo polinomial de grado 11, donde el valor de α se escoge como 0.1 o 100. En el primer caso, la solución ajuste razonablemente bien al modelo subyacente mientras que en el segundo caso el modelo parece requerir mayor complejidad. El modelo L_1 tiene popularidad pues permite que algunos valores sean cero, de ser posible. La razón para ello es explicada en la siguiente sección.

9.2.2. Restricciones L_1 y L_2

Las restricciones L_1 y L_2 tienen la forma $\Omega(\theta) = \alpha \parallel \theta \parallel$ y $\Omega(\theta) = 1/2\alpha\theta^T\theta$, respectivamente. La diferencia entre ellas podría quedar ilustrada en el siguiente ejemplo, tomado de un grupo de discusión. Supóngase que se tiene un vector $\mathbf{x} = (1, \epsilon)$, para un número pequeño $\epsilon > 0$. Las normas L_1 y L_2 para este caso son

$$\begin{aligned} \parallel \mathbf{x} \parallel_1 &= \parallel 1 \parallel + \parallel \epsilon \parallel = 1 + \epsilon, \\ \parallel \mathbf{x} \parallel_2 &= 1^2 + \epsilon^2 = 1 + \epsilon^2, \end{aligned} \tag{9.12}$$

Supóngase ahora que se busca la reducción de $\mathbf{x}$ por una cantidad $\delta < \epsilon$. Si la reducción ocurre en el primer componente de $\mathbf{x}$, entonces el valor de 1 pasa a ser $1 - \delta$, por ejemplo con el vector $\mathbf{x}' = (1 - \delta, \epsilon)^T$. Como consecuencia el valor de las normas pasa a ser

$$\begin{aligned} \parallel \mathbf{x}' \parallel_1 &= \parallel 1 - \delta \parallel + \parallel \epsilon \parallel = 1 - \delta + \epsilon, \\ \parallel \mathbf{x}' \parallel_2 &= (1 - \delta)^2 + \epsilon^2 = 1 - 2\delta + \delta^2 + \epsilon^2. \end{aligned} \tag{9.13}$$

Por otro lado, si la reducción es en el segundo componente, entonces el valor de ϵ para a ser $\epsilon - \delta$, por ejemplo con el vector $\mathbf{x}'' = (1, \epsilon - \delta)^T$. Como consecuencia el valor de las normas pasa a ser

$$\begin{aligned} \parallel \mathbf{x}'' \parallel_1 &= \parallel 1 \parallel + \parallel \epsilon - \delta \parallel = 1 + \epsilon - \delta = 1 - \delta + \epsilon, \\ \parallel \mathbf{x}'' \parallel_2 &= 1^2 + (\epsilon - \delta)^2 = 1 + \epsilon^2 - 2\delta\epsilon + \delta^2 = 1 - 2\delta\epsilon + \delta^2 + \epsilon^2. \end{aligned} \tag{9.14}$$

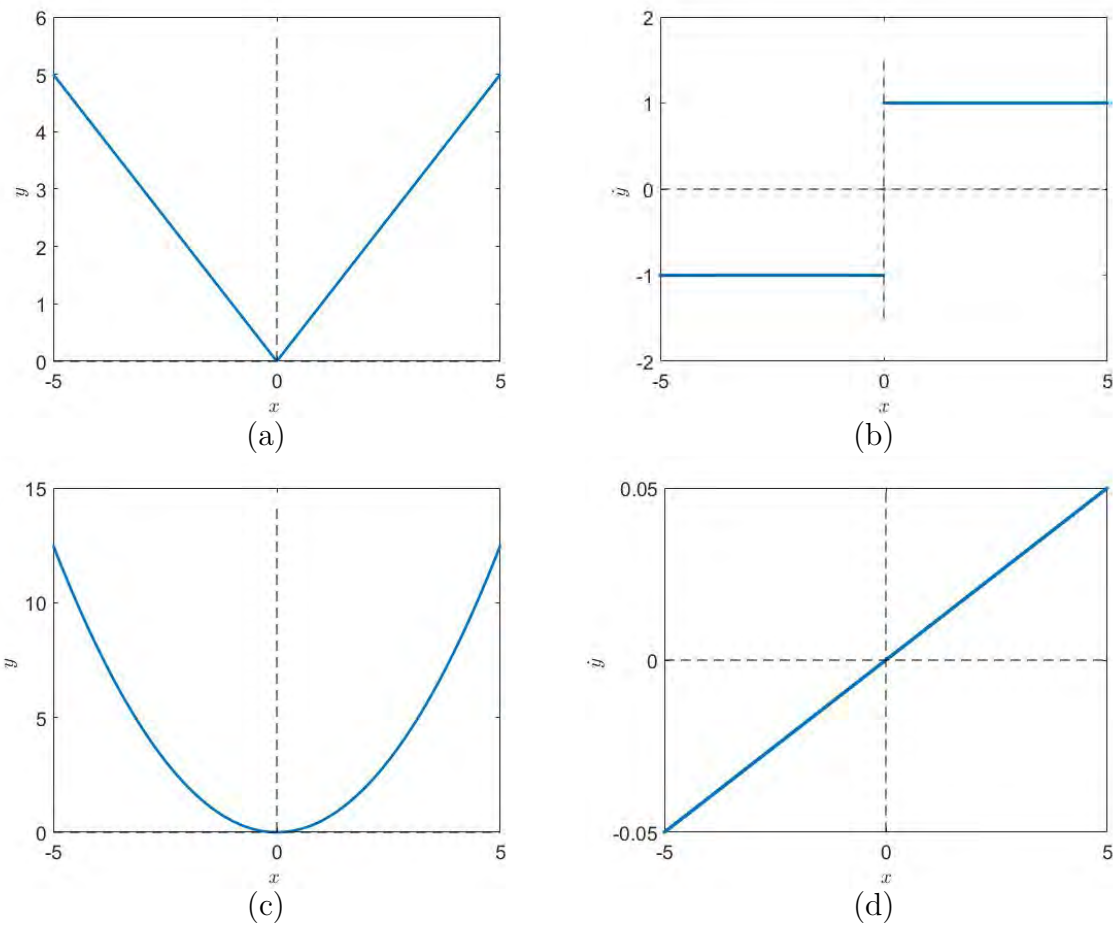


Figura 9.3: Funciones $\parallel x \parallel$ y $1/2x^2$. Para $\parallel x \parallel$ el gradiente (subgradiente) se mantiene constante. Para $1/2x^2$ entre más pequeño es el valor de la función, su gradiente es más pequeño. Esto hace sea más difícil llegar cero en la función cuadrática al utilizar descenso por gradiente.

Como se puede observar, en la norma L_1 una reducción en cualquiera de los dos componentes resulta en la misma magnitud del vector. Por su lado, en la norma L_2 la magnitud del resultado depende de la magnitud del componente en donde ocurre la reducción. De hecho, entre más pequeño es el valor original del componente sobre el que ocurre la reducción, menor será el impacto sobre el valor de la norma.

9.2.3. Terminación Temprana como Regularización

Las siguientes notas se basan en el libro de Goodfellow.

La función de costo puede expandirse en términos de su serie de Taylor como

$$\hat{J}(\omega) = J(\omega^*) + (\omega - \omega^*)\nabla J(\omega) + \frac{1}{2}(\omega - \omega^*)^T \mathbf{H}(\omega - \omega^*) + \varphi^3, \tag{9.15}$$

donde $\mathbf{H}$ es el Hessiano y φ corresponde a los términos cúbico y superior. En el caso en que $\hat{J}$ tenga forma cuadrática, la aproximación de Taylor es exacta y el gradiente y los términos

arriba del cuadrático desaparecen. Esto resulta en la función

$$\nabla_\omega \hat{J}(\omega) = \mathbf{H}(\omega - \omega^*). \tag{9.16}$$

Si incluimos regularización, y resolvemos ahora para la variable regularizada $\tilde{\omega}$, etonces podemos tener la expresión

$$\begin{aligned} \mathbf{H}(\tilde{\omega} - \omega^*) + \alpha\tilde{\omega} &= 0, \\ \mathbf{H}\tilde{\omega} - \mathbf{H}\omega^* + \alpha\tilde{\omega} &= 0, \\ (\mathbf{H} - \alpha\mathbf{I})\tilde{\omega} &= \mathbf{H}\omega^* \\ \tilde{\omega} &= (\mathbf{H} - \alpha\mathbf{I})^{-1}\mathbf{H}\omega^* \end{aligned} \tag{9.17}$$

La matriz $\mathbf{H}$ es real y simétrica. Por ello, acepta una descomposición en eigenvalores y eigenvectores de la forma $\mathbf{H} = \mathbf{Q}\Lambda\mathbf{Q}^T$. Resolviendo la descomposición para la ecuación anterior, recordando que $\mathbf{Q}^T = \mathbf{Q}^{-1}$, tenemos

$$\begin{aligned} \tilde{\omega} &= (\mathbf{H} - \alpha\mathbf{I})^{-1}\mathbf{H}\omega^* \\ &= (\mathbf{Q}\Lambda\mathbf{Q}^T - \alpha\mathbf{I})^{-1}\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* \\ &= (\mathbf{Q}(\Lambda\mathbf{Q}^T - \alpha\mathbf{Q}^T\mathbf{I}))^{-1}\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* \\ &= (\mathbf{Q}(\Lambda - \alpha\mathbf{Q}^T\mathbf{I}\mathbf{Q})\mathbf{Q}^T)^{-1}\mathbf{Q}\Lambda\mathbf{Q}^T \\ &= (\mathbf{Q}(\Lambda - \alpha\mathbf{I})\mathbf{Q}^T)^{-1}\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* \\ &= \left(\left(\mathbf{Q}^T\right)^{-1}(\Lambda - \alpha\mathbf{I})^{-1}\mathbf{Q}^{-1}\right)\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* \\ &= (\mathbf{Q}(\Lambda - \alpha\mathbf{I})^{-1}\mathbf{Q}^{-1})\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* \\ &= \mathbf{Q}(\Lambda - \alpha\mathbf{I})^{-1}\Lambda\mathbf{Q}^T\omega^* \\ \mathbf{Q}^T\tilde{\omega} &= (\Lambda - \alpha\mathbf{I})^{-1}\Lambda\mathbf{Q}^T\omega^* \end{aligned} \tag{9.18}$$

La siguiente igualdad puede ser demostrada

$$\begin{aligned} (\Lambda - \alpha\mathbf{I})^{-1}\Lambda &= \mathbf{I} - (\Lambda + \alpha\mathbf{I})^{-1}\alpha, \\ (\Lambda - \alpha\mathbf{I})^{-1}\Lambda + (\Lambda + \alpha\mathbf{I})^{-1}\alpha &= \mathbf{I}, \\ (\Lambda - \alpha\mathbf{I})^{-1}(\Lambda + \alpha\mathbf{I}) &= \mathbf{I}, \\ \mathbf{I} &= \mathbf{I}. \end{aligned} \tag{9.19}$$

Terminación temprana es el procedimiento mediante el cual tomamos el valor de los pesos de la red cuando mientras el error de entrenamiento comienza a bajar, el error de prueba comienza a subir. En el proceso de descenso por gradiente, los nuevos pesos son actualizados iterativamente, siguiendo el esquema

$$\omega^\tau = \omega^{\tau-1} + \epsilon\nabla_\omega \hat{J}(\omega^{\tau-1}), \tag{9.20}$$

donde, para una ecuación cuadrática, el mínimo ocurre cuando (9.16) se cumple. Esto resulta

en

$$\begin{aligned}
\omega^{\tau} =&\omega^{\tau-1} - \epsilon\mathbf{H}(\omega^{\tau-1} - \omega^*),\\
=&\omega^{\tau-1} - \epsilon\mathbf{H}\omega^{\tau-1} + \epsilon\mathbf{H}\omega^*,\\
=&(\mathbf{I} - \epsilon\mathbf{H})\omega^{\tau-1} + \epsilon\mathbf{H}\omega^*,\\
\omega^{\tau} - \omega^* =&(\mathbf{I} - \epsilon\mathbf{H})\omega^{\tau-1} - \omega^* + \epsilon\mathbf{H}\omega^*,\\
\omega^{\tau} - \omega^* =&(\mathbf{I} - \epsilon\mathbf{H})\omega^{\tau-1} - (\mathbf{I} - \epsilon\mathbf{H})\omega^*,\\
\omega^{\tau} - \omega^* =&(\mathbf{I} - \epsilon\mathbf{H})(\omega^{\tau-1} - \omega^*).
\end{aligned} \tag{9.21}$$

Reescribiendo $\mathbf{H}$ nuevamente en términos de su descomposición en valores singulares tenemos

$$\begin{aligned}
\omega^{\tau} - \omega^* =&(\mathbf{I} - \epsilon\mathbf{Q}\Lambda\mathbf{Q}^T)(\omega^{\tau-1} - \omega^*),\\
\omega^{\tau} - \omega^* =&\mathbf{Q}(\mathbf{Q}^T\mathbf{I} - \epsilon\Lambda\mathbf{Q}^T)(\omega^{\tau-1} - \omega^*),\\
\omega^{\tau} - \omega^* =&\mathbf{Q}(\mathbf{Q}^T\mathbf{I}\mathbf{Q} - \epsilon\Lambda)\mathbf{Q}^T(\omega^{\tau-1} - \omega^*),\\
\omega^{\tau} - \omega^* =&\mathbf{Q}(\mathbf{I} - \epsilon\Lambda)\mathbf{Q}^T(\omega^{\tau-1} - \omega^*).
\end{aligned} \tag{9.22}$$

Si suponemos que en $\tau = 0$, $\omega^0 = 0$, entonces el valor de ω^1 estará dado por

$$\begin{aligned}
\omega^1 - \omega^* =&\mathbf{Q}(\mathbf{I} - \epsilon\Lambda)\mathbf{Q}^T(-\omega^*),\\
\omega^1 - \omega^* =& -\mathbf{Q}(\mathbf{I} - \epsilon\Lambda)\mathbf{Q}^T\omega^*,\\
\omega^1 - \omega^* =& -\mathbf{Q}(\mathbf{I}\mathbf{Q}^T\omega^* - \epsilon\Lambda\mathbf{Q}^T\omega^*),\\
\omega^1 - \omega^* =& -\mathbf{Q}\mathbf{I}\mathbf{Q}^T\omega^* + \epsilon\mathbf{Q}\Lambda\mathbf{Q}^T\omega^*,\\
\omega^1 - \omega^* =&(-\mathbf{Q}\mathbf{I} + \epsilon\mathbf{Q}\Lambda)\mathbf{Q}^T\omega^*,\\
\omega^1 =&\mathbf{Q}(-\mathbf{I} + \epsilon\Lambda)\mathbf{Q}^T\omega^* + \omega^* = (-\mathbf{Q}\mathbf{I}\mathbf{Q}^T + \epsilon\mathbf{Q}\Lambda\mathbf{Q}^T)\omega^* + \omega^* = \epsilon\mathbf{Q}\Lambda\mathbf{Q}^T,\\
\mathbf{Q}^T\omega^1 =&(-\mathbf{I} + \epsilon\Lambda)\mathbf{Q}^T\omega^* + \mathbf{Q}^T\omega^*,\\
\mathbf{Q}^T\omega^1 =&((-\mathbf{I} + \epsilon\Lambda) + \mathbf{I})\mathbf{Q}^T\omega^*,\\
\mathbf{Q}^T\omega^1 =&(\mathbf{I} - (\mathbf{I} - \epsilon\Lambda))\mathbf{Q}^T\omega^*,
\end{aligned} \tag{9.23}$$

En $\tau = 1$, el valor de ω^2 estará dado por

$$\begin{aligned}
\omega^2 - \omega^* =&\mathbf{Q}(\mathbf{I} - \epsilon\Lambda)\mathbf{Q}^T(\omega^1 - \omega^*),\\
\omega^2 - \omega^* =&\mathbf{Q}(\mathbf{I} - \epsilon\Lambda)\mathbf{Q}^T(\epsilon\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* - \omega^*),\\
\omega^2 - \omega^* =&\mathbf{Q}(\mathbf{I} - \epsilon\Lambda)(\epsilon\mathbf{Q}^T\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* - \mathbf{Q}^T\omega^*),\\
\omega^2 - \omega^* =&(\mathbf{Q}\mathbf{I} - \epsilon\mathbf{Q}\Lambda)(\epsilon\Lambda\mathbf{Q}^T\omega^* - \mathbf{Q}^T\omega^*),\\
\omega^2 - \omega^* =&\mathbf{Q}\mathbf{I}(\epsilon\Lambda\mathbf{Q}^T\omega^* - \mathbf{Q}^T\omega^*) - \epsilon\mathbf{Q}\Lambda(\epsilon\Lambda\mathbf{Q}^T\omega^* - \mathbf{Q}^T\omega^*),\\
\omega^2 - \omega^* =&\epsilon\mathbf{Q}\mathbf{I}\Lambda\mathbf{Q}^T\omega^* - \mathbf{Q}\mathbf{I}\mathbf{Q}^T\omega^* - \epsilon\epsilon\mathbf{Q}\Lambda\Lambda\mathbf{Q}^T\omega^* + \epsilon\mathbf{Q}\Lambda\mathbf{Q}^T\omega^*,\\
\omega^2 - \omega^* =&\epsilon\mathbf{Q}\Lambda\mathbf{Q}^T\omega^* - \mathbf{I}\omega^* - \epsilon^2\mathbf{Q}\Lambda^2\mathbf{Q}^T\omega^* + \epsilon\mathbf{Q}\Lambda\mathbf{Q}^T\omega^*,\\
\omega^2 - \omega^* =& -\mathbf{Q}(\mathbf{I} - 2\epsilon\Lambda + \epsilon^2\Lambda^2)\mathbf{Q}^T\omega^*,\\
\omega^2 =& -\mathbf{Q}(\mathbf{I} - 2\epsilon\Lambda + \epsilon^2\Lambda^2)\mathbf{Q}^T\omega^* + \omega^*,\\
\mathbf{Q}^T\omega^2 =& -\mathbf{Q}^T\mathbf{Q}(\mathbf{I} - 2\epsilon\Lambda + \epsilon^2\Lambda^2)\mathbf{Q}^T\omega^* + \mathbf{Q}^T\omega^*,\\
\mathbf{Q}^T\omega^2 =&(-(\mathbf{I} - 2\epsilon\Lambda + \epsilon^2\Lambda^2) + \mathbf{I})\mathbf{Q}^T\omega^*,\\
\mathbf{Q}^T\omega^2 =&(\mathbf{I} - (\mathbf{I} - \epsilon\Lambda)^2)\mathbf{Q}^T\omega^*.
\end{aligned} \tag{9.24}$$

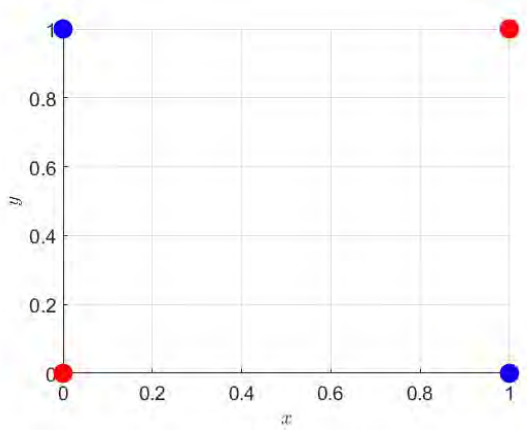
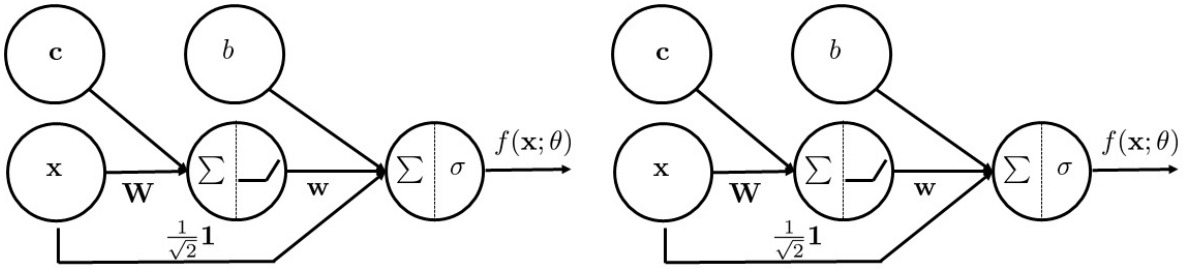


Figura 9.4: Ilustración de la función XOR. Se ilustra como no hay una función lineal que separe las muestras rojas de las azules (mejor vista en color).



(a) XOR (b) XOR ResNet

Figura 9.5: Redes neuronales para resolver el problema del XOR.

De donde se puede deducir que en general

$$\mathbf{Q}^T\omega^\tau = (\mathbf{I} - (\mathbf{I} - \epsilon\Lambda)^\tau)\mathbf{Q}^T\omega^*. \tag{9.25}$$

Con ello puede apreciarse que terminación temprana y regularización son equivalentes cuando

$$(\mathbf{I} - \epsilon\Lambda)^\tau = (\Lambda + \alpha\mathbf{I})^{-1}\alpha \tag{9.26}$$

9.3. Proceso de Optimización

Los esquemas de optimización para aprendizaje profundo siguen alguna variante de descenso por gradiente. En ellos, un valor inicial busca su camino hacia un valor mínimo. Siendo un espacio multidimensional, con funciones de pérdidas no lineales, los mínimos locales abundan.

9.3.1. Optimización de Redes Neuronales

Para ilustrar la dificultad en la búsqueda del valor mínimo para los parámetros, considere el caso de la función XOR. La función XOR es un mapeo lógico con la siguiente tabla de

verdad

$$f^* : \left\{ \begin{array}{ll} 0, & 0 \\ 0, & 1 \\ 1, & 0 \\ 1, & 1 \end{array} \right\} \rightarrow \left\{ \begin{array}{l} 0 \\ 1 \\ 1 \\ 0 \end{array} \right\}.$$

Tal como se ilustra en la Figura 9.4, no hay una función en 2D que separe las clases $\{0, 1\}$[44]. Una red que pudiera resolver el XOR pudiera ser construida mediante la función

$$f(\mathbf{x}^T; \mathbf{W}, \mathbf{c}^T, \mathbf{w}^T, b) = \sigma\left(\text{máx}\left\{0_{1\times 2}, \mathbf{x}^T\mathbf{W} + \mathbf{c}^T\right\}\mathbf{w} + b\right),$$

donde σ corresponde a la función sigmoide

$$\sigma(\gamma) = \frac{1}{1 + e^{-\gamma}}.$$

Para el conjunto de parámetros $\theta = \{\mathbf{W}, \mathbf{c}^T, \mathbf{w}^T, b\}$, una función que pudiera identificar la pérdida podría ser

$$J(\theta; \mathbf{X}) = \parallel f^*(\mathbf{X}) - f(\mathbf{X}; \theta) \parallel_2^2 .$$

Note que mientras que f^* está definida para cuatro puntos, f está definida para $\mathbf{x} \in \mathcal{R}^2$.

Uno podría realizar el aprendizaje mediante descenso por gradiente mediante la formulación

$$\theta^n \leftarrow \theta^{n-1} + \rho\nabla_\theta J(\theta),$$

donde $\rho = 0.01$ es la tasa de aprendizaje. Otra arquitectura para resolver el problema del XOR podría ser al estilo de ResNet[27]. Esta podría ser construida mediante

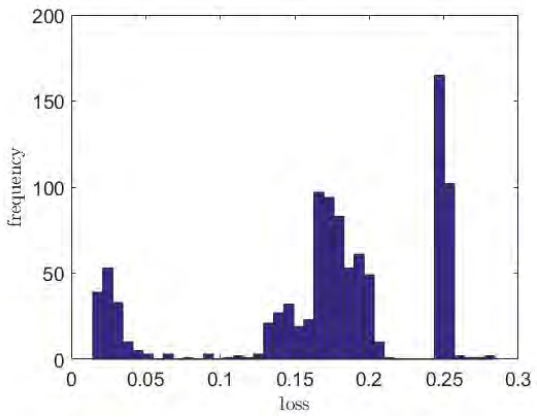
$$f(\mathbf{x}^T; \theta) = \sigma\left(\mathcal{F}(\mathbf{x}^T; \theta) + \frac{1}{\sqrt{2}}\mathbf{x}^T\mathbf{1} + b\right),$$

donde

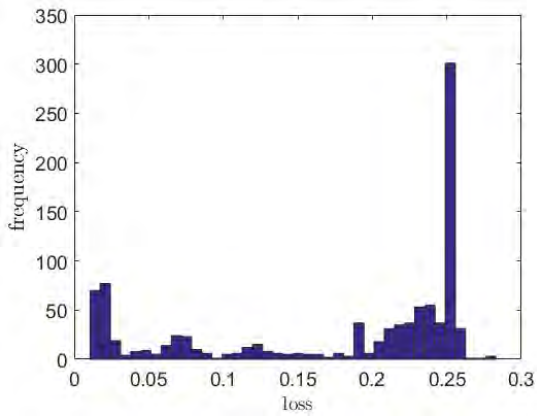
$$\mathcal{F}(\mathbf{x}^T; \theta) = \text{máx}\left\{0_{1\times 2}, \mathbf{x}^T\mathbf{W} + \mathbf{c}^T\right\}\mathbf{w}.$$

Uno puede ganar intuición sobre la naturaleza del espacio de búsqueda mediante procesos consecutivos de optimización. Así, al inicializar los parámetros $\theta = \{\mathbf{W}, \mathbf{c}, \mathbf{w}, b\}$ de forma aleatoria siguiendo una distribución normal con media cero y desviación estándar 0.35, entrenamos a la red neuronal para minimizar la función de costo. Si repetimos el proceso 1,000 veces, el valor de la función de pérdida seguiría la distribución ilustrada en la Figura 9.6. Los dos espacios de solución se presentan en la Figura 9.8. En ella, las clases se presentan por colores rojo y azul. Uno de ellos corresponde a una solución concava y otro a una solución convexa.

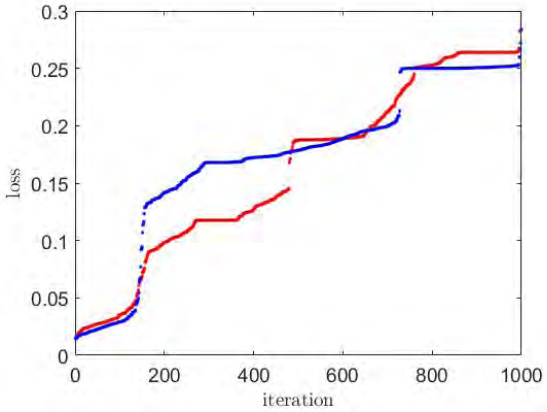
Uno puede concluir que en el proceso de minimización, la inicialización es crucial para llegar a una solución satisfactoria. Al comparar una arquitectura canónica y una arquitectura ResNet notamos que es más probable que se encuentre una solución en ResNet. No obstante, es más probable que la solución ResNet sea incorrecta. Para una muestra de inicializaciones aleatorias, la arquitectura de ResNet condujo a un mayor número de clasificadores correctos que la implementación de canónica. Por otro lado, las soluciones están repartidas a lo largo del espacio de los parámetros.



(a) Versión simple

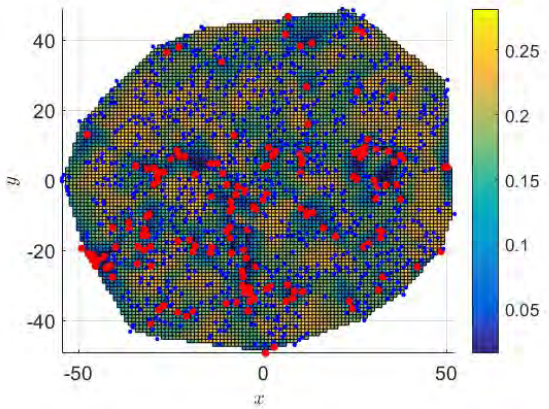


(b) versión ResNet

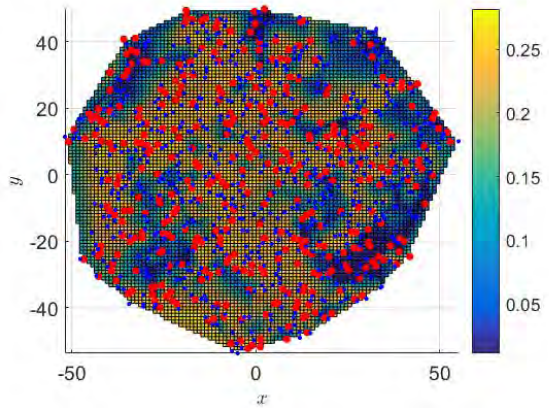


(c) Comparación de valores de histograma.

Figura 9.6: Distribución de los valores de la función de pérdida para dos arquitecturas de solución.



(a) XOR



(b) XOR ResNet

Figura 9.7: Espacio t-SNE para los valores iniciales de los parámetros.

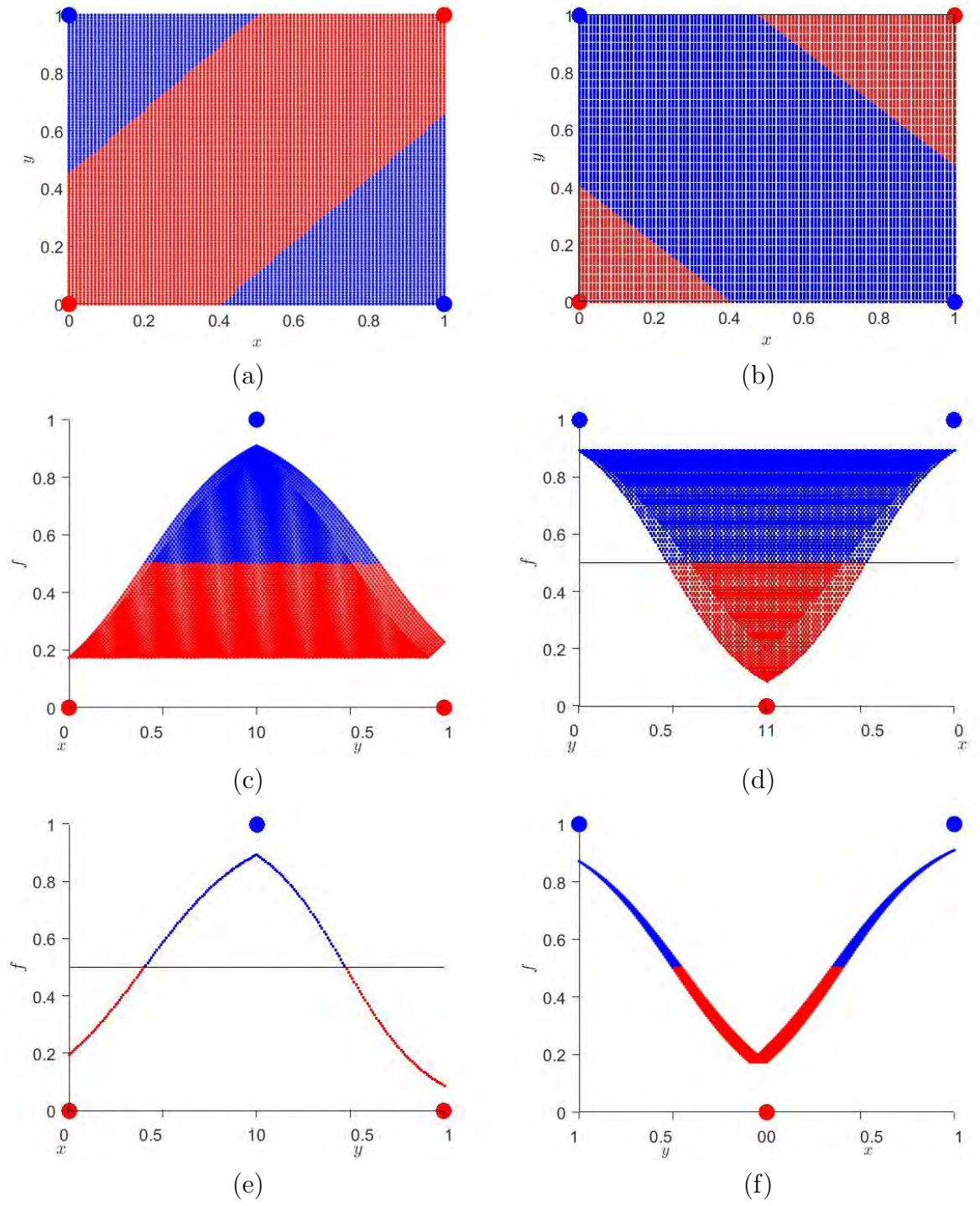


Figura 9.8: Espacios de solución para el XOR. (a), (c), (e) presentan vistas ortogonales correspondientes a un tipo de solución, mientras que (b), (d), y(f) presentan el segundo conjunto de vistas ortogonales.

Para ilustrar la distribución de los valores iniciales para los parámetros, proyectamos los parámetros θ en el espacio t-SNE, que conducen a clasificadores correctos (puntos rojos) y clasificadores incorrectos (puntos azules) (Figure 9.7). En el fondo, ilustramos el valores de pérdida interpolados correspondientes a los valores de los parámetros iniciales en el espacio t-SNE. Por su parte en ResNet, las áreas donde la pérdida es baja parecen ser más abundantes. No obstante, existen clasificadores correctos en áreas de baja pérdida y áreas de pérdida relativamente alta. Además, podemos encontrar clasificadores incorrectos en áreas de baja pérdida.

9.3.2. Esquemas de Optimización

Algunos de los esquemas de optimización para redes neuronales más utilizados incluyen los siguientes.

Descenso por gradiente estocástico (SGD). SGD toma el promedio del gradiente de un minibatch de m muestras. Es decir,

$$\tilde{\mathbf{g}} \leftarrow \frac{1}{m}\nabla_\theta \sum_i L(\mathbf{x}_i; \theta, \mathbf{y}_i), \tag{9.27}$$

y aplica el siguiente esquema de actualización de los parámetros.

$$\theta \leftarrow \theta - \epsilon\hat{\mathbf{g}}. \tag{9.28}$$

Momentum. El método de momentum acumula un promedio sobre una serie de observaciones para evitar ser atrapado en mínimos locales.

$$\tilde{\mathbf{v}} \leftarrow \alpha\mathbf{v} - \epsilon\nabla_\theta \left(\frac{1}{m} \sum_i L(\mathbf{x}_i; \theta, \mathbf{y}_i \right), \tag{9.29}$$

y aplica el esquema de actualización de los parámetros

$$\theta \leftarrow \theta - \mathbf{v}, \tag{9.30}$$

donde $\mathbf{v}$ es un vector que describe una masa unitaria moviéndose a una velocidad vectorial $\mathbf{v}$.

Momentum de Nesterov. El método de momentum de Nesterov es una variante del método anterior cuyas ecuaciones son

$$\tilde{\mathbf{v}} \leftarrow \alpha\mathbf{v} - \epsilon\nabla_\theta \left(\frac{1}{m} \sum_i L(\mathbf{x}_i; \theta + \alpha\mathbf{v}, \mathbf{y}_i \right), \tag{9.31}$$

y aplica el esquema de actualización de los parámetros

$$\theta \leftarrow \theta - \mathbf{v}, \tag{9.32}$$

donde $\mathbf{v}$ es un vector que describe una masa unitaria moviéndose a una velocidad vectorial $\mathbf{v}$.

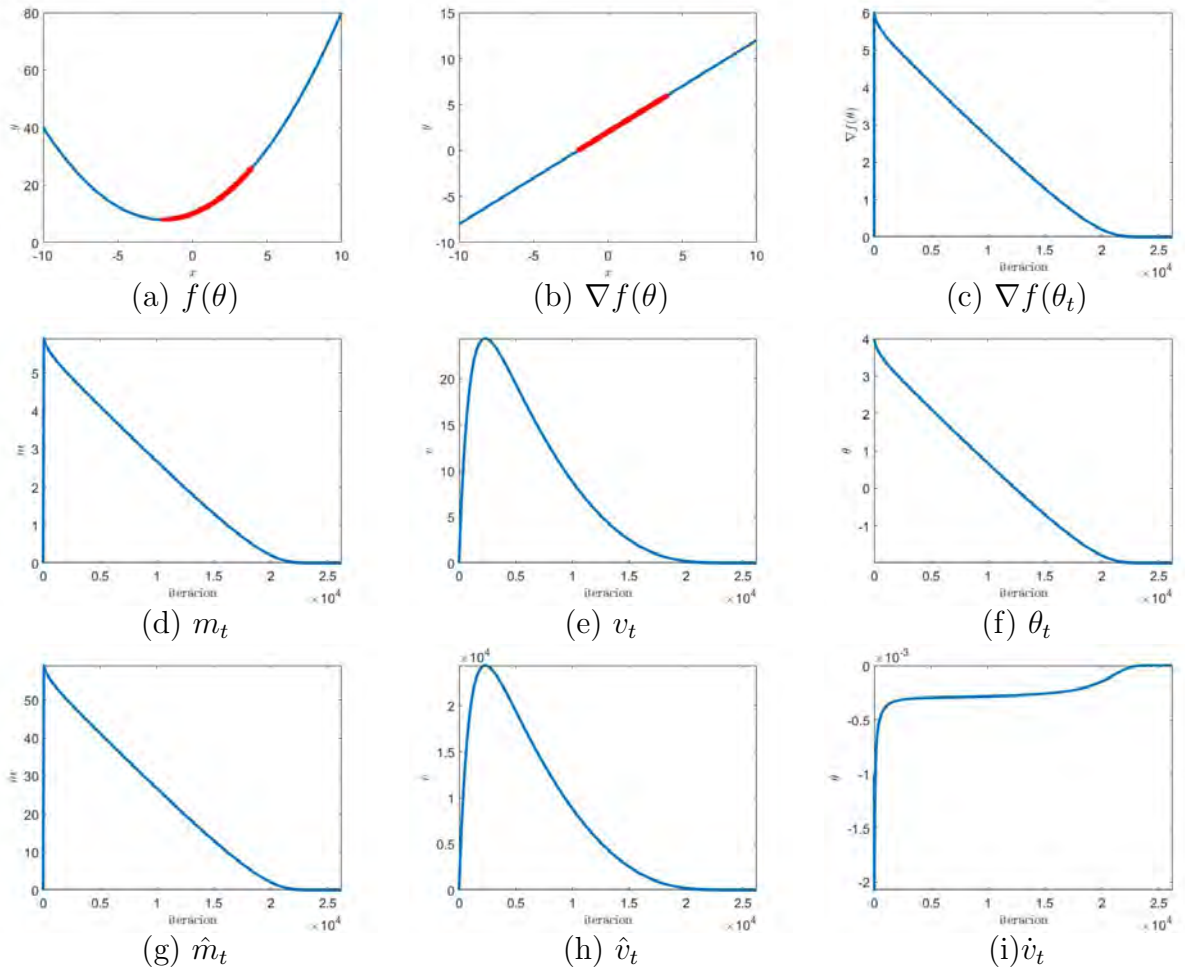
Como una forma de ilustración vamos a considerar el esquema de optimización conocido como Adams, el cual utiliza el primer y segundo momentos del gradiente[31]. El método es descrito en el Algoritmo5. Como ejemplo considere la función

$$y = ax^2 + bx + c, \tag{9.33}$$

con $a = 0.5$, $b = 2$ y $c = 10$. Supongamos que inicialmente el valor del parámetro $\theta = 4$. El algoritmo calcula el gradiente, $\mathbf{g}_k$, actualiza el estimado del primer momento m_k, actualiza el estimado del segundo momento v_k, corrige el sesgo de los momentos, $\hat{m}_k$ y $\hat{v}_k$, y actualiza el valor del parámetro mediante el esquema

$$\theta_k \leftarrow \theta_{k-1} - \alpha\hat{m}_k/(\sqrt{\hat{v}_k} + \epsilon), \tag{9.34}$$

para un valor de ϵ arbitrariamente pequeño pero mayor que cero. La Figura 9.9 muestra el ejemplo de correr el algoritmo de optimización para nuestro ejemplo. Note en las subfiguras (a) y (b) como el parámetro converge hacia la solución. El resto de las subfiguras muestra la evolución de las variables auxiliares conforme avanza la ejecuón del algoritmo.



(a) $f(\theta)$ (b) $\nabla f(\theta)$ (c) $\nabla f(\theta_t)$

(d) m_t (e) v_t (f) θ_t

(g) $\hat{m}_t$ (h) $\hat{v}_t$ (i)$\dot{v}_t$

Figura 9.9: Ilustración del comportamiento del optimizador de Adams para una curva cuadrática.

Llamada : $\theta \leftarrow$ `Optimización` (θ, α)
Entradas: La constante de aprendizaje α, el valor inicial de los parámetros θ.
Salidas : El valor óptimo de los parámetros θ.

`// hiper-parámetros para la caída exponencial de los promedios móviles`
$\beta_1 \leftarrow 0.9; \beta_2 \leftarrow 0.999;$
`// constante para evitar singularidades`
$\epsilon \leftarrow 10^{-8};$
`// inicialización del primero y segundo momento`
$m \leftarrow 0; v \leftarrow 0;$
`// inicializar parámetros para la búsqueda del óptimo`
$k \leftarrow 1; c \leftarrow$ false;
while c **do**
  $k \leftarrow k + 1;$
  `// calcula gradiente`
  $g_k \leftarrow f_p(\theta_{k-1});$
  `// actualiza el estimado del primer momento`
  $m_k = \beta_1 m_{k-1} + (1 - \beta_1) g_k;$
  `// actualiza el estimado del segundo momento`
  $v_k = \beta_2 v_{k-1} + (1 - \beta_2) g_k^2;$
  `// corrige el sesgo del primer momento`
  $\hat{m}_k \leftarrow m_k / (1 - \beta_1);$
  `// corrige el sesgo del segundo momento`
  $\hat{v}_k = v_k / (1 - \beta_2);$
  `// actualiza los parámetros`
  $\theta_k = \theta_{k-1} - \alpha \hat{m}_k / \sqrt{\hat{v}_k + \epsilon};$
  `// prueba de convergencia`
  $c \leftarrow \mid \theta_k - \theta_{k-1} \mid < \epsilon;$
end

Algorithm 5: Algoritmo para la optimización de parámetros por el método de Adams.

CAPÍTULO 10

Aprendizaje por Refuerzo

En ámbito de aprendizaje maquinal, el aprendizaje por refuerzo (*reinforcement learning*) es una técnica en la cual *agentes* aprenden basados en *recompensas*. En contraposición al aprendizaje supervisado, en lugar de conjuntos de entrenamiento, el problema se define en términos de un agente, un ambiente, un estado, las acciones y las recompensas[47]. Por ejemplo, considere el ejemplo de *blockout* ilustrado en la Figura 10.1. El propósito del juego de *blockout* es eliminar los bloques en la parte superior de la pantalla con el uso de una pelota que rebota en las paredes y raqueta. Cuando la pelota pasa por los bloques los desintegra, da una recompensa en la forma de puntos al jugador. Cuando la pelota pasa la raqueta, y sale por la parte baja de la pantalla, el jugador pierde una vida. En un cierto momento, el jugador puede mover la raqueta horizontalmente a uno u otro lado. Esto es particularmente útil cuando la pelota esta por salir del área de juego. Tocar la pelota permite que esta se mueva hacia arriba, en la dirección de los bloques.

10.1. Esquema de Aprendizaje

En aprendizaje por refuerzo hay un conjunto de acciones que eventualmente producen un estado. El objetivo del aprendizaje por refuerzo es diseñar una estrategia de acciones o política para alcanzar un estado conocido como meta. Para alcanzar la meta, típicamente nos interesa maximizar la función valor-acción[46]

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left(r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots \mid s_t = s, a_t = a, \pi\right), \tag{10.1}$$

para una constante de reducción $0 \leq \gamma \leq 1$ que refleja el plazo en el cual una recompensa r es recibida, una observación s que resulta en una acción a, bajo una política $\pi = P(a \mid s)$ la cual es producto de una acción dada cierta observación, en un tiempo t. Por ejemplo, supongamos que se tienen una recompensa $r_t = 2$, un valor de descuento $\gamma = 0.9$, y dos estados t y $t+1$ con la siguiente relación de acciones y Q-valores.

t		$t+1$	
acción	Q-valor	acción	Q-valor
α	4	α	3
β	2	β	4
γ	6	γ	1

Entonces, para la acción α en el tiempo t, los valores de actualización corresponden a

Figura 10.1: Pantalla del juego *Blockout*. El propósito del juego es eliminar los bloques en la parte superior de la pantalla con el uso de una pelota que rebota en las paredes y raqueta. Cuando la pelota pasa por los bloques los desintegra, da una recompensa en la forma de puntos al jugador. Cuando la pelota pasa la raqueta, y sale por la parte baja de la pantalla, el jugador pierde una vida. En un cierto momento, el jugador puede mover la raqueta horizontalmente a uno u otro lado. Esto es particularmente útil cuando la pelota esta por salir del área de juego. Tocar la pelota permite que esta se mueva hacia arriba, en la dirección de los bloques.

$r_t + \gamma Q(s_{t+1}, a)$
$2 + 0.9 \times 3 = 4.7$
$2 + 0.9 \times 4 = 5.6$
$2 + 0.9 \times 1 = 2.9$

de donde resulta un valor máximo de 5.6. Este valor es mayor al Q-valor de α en t. Durante el entrenamiento se sigue una política ϵ-*greedy*, la cual consiste en escoger la opción que más recompensa da, con una probabilidad de $1 - \epsilon$, ó una opción completamente aleatoria con una probabilidad de ϵ. Siguiendo este proceso para las acciones de β y γ, los nuevos Q-valores pudieran ser

t		$t+1$	
acción	Q-valor	acción	Q-valor
α	5.6	α	3
β	5.6	β	4
γ	6	γ	1

dependiendo del resultado aleatorio de la política de selección de acciones.

La combinación entre aprendizaje por refuerzo y aprendizaje profundo ha dado lugar a lo que se conoce como redes profundas Q (DQN), el cual fue introducido por Mnih *et al.*[46]. En

su propuesta, Mnih *et al.* aproximan la función $Q(s, a; \theta_i)$, donde θ_i son los pesos de la red en la iteración i, mediante una CNN. Para el entrenamiento, su DQN usa una estrategia conocida como repetición de la experiencia (*experience replay*) donde un subconjunto o *minibatch* de la experiencia del agente es utilizada para el aprendizaje. Esta experiencia se almacena en la variable $D_t = \{e_1, \ldots, e_t\}$, donde $e_t = (s_t, a_t, r_t, s_{t+1})$ para un momento en el tiempo t.

Con ello, la función de pérdida en la iteración i de define de acuerdo a la función

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')\sim U(D)} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i) \right)^2 \right], \tag{10.2}$$

donde $(s, a, r, s') \sim U(D)$ es un minibatch extraído de forma aleatoria y suponiendo una distribución D uniformemente distribuida, θ_i^- son parámetros de la red en la iteración i y θ_i son los parámetros de la red para calcular el valor Q en la iteración i. Los parámetros θ_i^- son actualizados con los parámetros θ_i cada C pasos.

En la práctica, una secuencia de acciones y observaciones , $s_t = x_1, a_1, x_2, a_2, \ldots, a_{t-1}x_t$ son alimentadas al algoritmo. El objetivo del agente es interactuar con su entorno y seleccionar sus acciones de tal forma que se maximize su recompensa en el futuro, donde

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}, \tag{10.3}$$

donde T es el momento en el tiempo en que el juego termina. La función que maximizaría el valor esperado sería

$$Q^*(s, a) = \max_{\pi} \mathbb{E} \left(R_t \mid s_t = s, a_t = a, \pi \right). \tag{10.4}$$

Sin embargo, la identidad de Bellman[3] nos dice que si el valor óptimo $Q^*(s', a')$, para una secuencia s', se conoce para todas las acciones a', entonces la estrategia óptima consiste en seleccionar la acción a' que maximiza el valor esperado $r + \gamma Q^*(s', a')$. Es decir

$$Q^*(s, a) = \mathbb{E}'_s \left(r + \gamma Q^*(s', a') \mid s, a, \pi \right). \tag{10.5}$$

El propósito de la red neuronal profunda es aproximar el valor de la función Q, mediante una serie de parámetros θ_i^- como

$$y = r + \gamma \max_{a'} Q(s', a'; \theta_i^-), \tag{10.6}$$

Así, la función de pérdida cambia con cada iteración i y puede ser definida como

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')} \left[(y - Q(s, a; \theta_i))^2 \right], \tag{10.7}$$

y su derivada es

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')} \left[\left(r + \gamma \max_{a}{}' Q(s', a'; \theta_i^-) - Q(s, a; \theta_i) \right) \nabla_{\theta_i} Q(s, a; \theta_i) \right], \tag{10.8}$$

donde el gradiente puede ser calculado mediante gradiente por descenso estocástico (SGD).

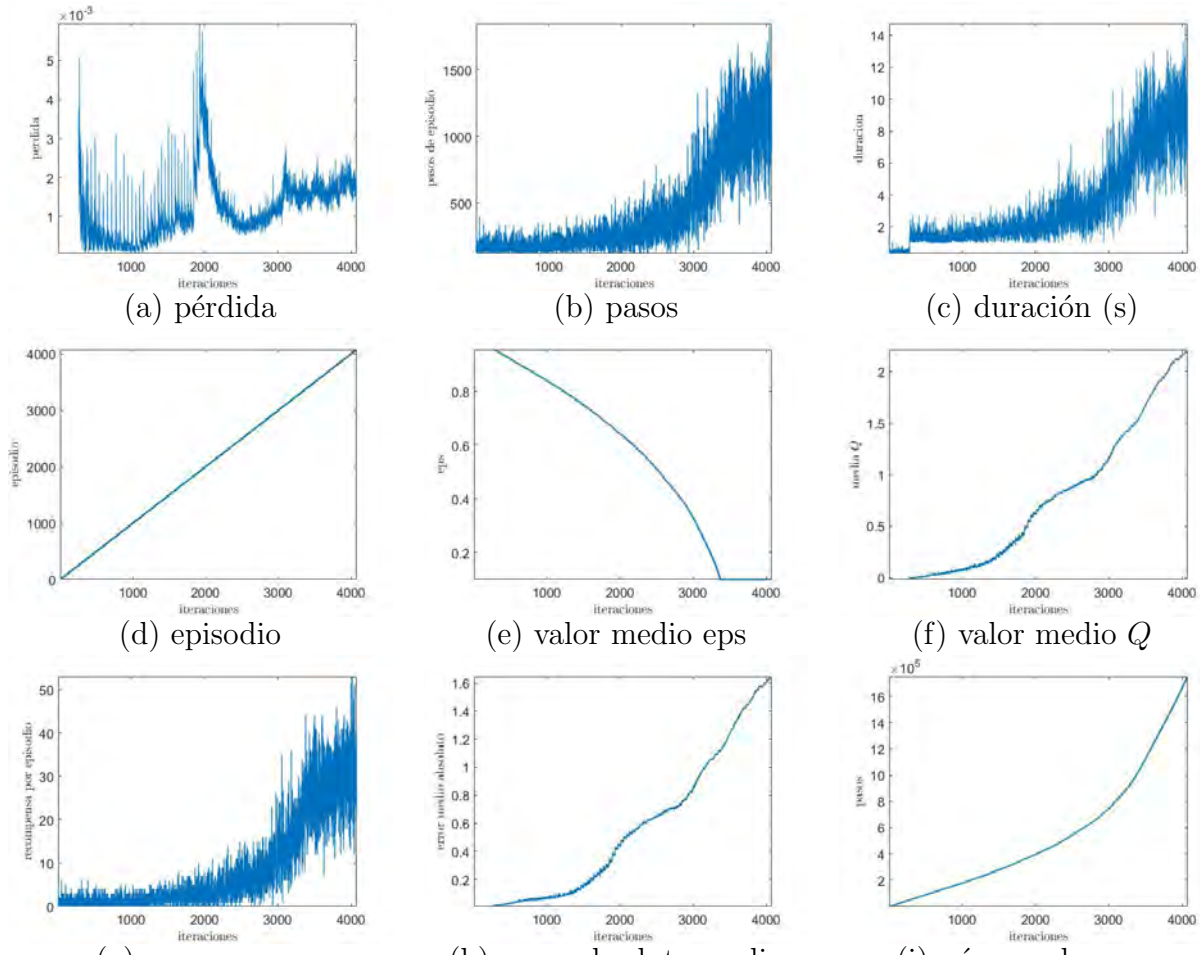
Los actuales esquemas de *reinforcement learning* siguen dos lazos principales de ejecución que corren en paralelo[54]. Durante el primer lazo, se ejecuta el juego y se colecta información. En este lazo se guarda en el *Replay Memory* el estado del juego, el correspondiente Q-valor y la recompensa. El segundo lazo se ejecuta cuando el *Replay Memory* esta llena. Primero se realiza un recálculo de los Q-valores. Enseguida se realiza un proceso de entrenamiento de la DQN para afinar los Q-valores actualizados. El conocimiento previo que se utiliza en el entrenamiento corresponde a las imágenes, el score del juego, el número de posibles acciones (pero no la correspondencia de la acción), y el conteo del número de vidas.

10.2. Implementación

OpenAI Gym is una herramienta que permite correr simulación de juegos y escenarios para aplicar aprendizaje por refuerzo. Gym proporciona una descripción del juego, incluyendo los comandos de control, las imágenes dentro de la simulación, y el inicio del juego. A continuación describimos la implementación de ejemplo que proporciona *keras-rl* para el juego de *blockout* (Figura 10.1). Por su lado, *keras-rl* es una herramienta para el aprendizaje por refuerzo construida sobre *keras*, el cual a su vez es un *front-end* para TensorFlow.

La arquitectura de la red DQN tiene la siguiente estructura. Las entradas son imágenes de 84×84 pixeles en 3 bandas de color. Luego vienen tres capas de convolución seguidas de activación ReLU. Las convoluciones generan 32, 64 y 64 filtros con tamaño de máscara 8×8, 4×4, y 3×3. Después de cada convolución los datos son submuestreados con un *stride* de 4×4, 2×2, 1×1. Enseguida de las capas de convolución se tienen capas completamente conectadas. La primera tiene 512 nodos. La segunda tiene un número de nodos igual al número de acciones del juego.

Durante el entrenamiento, las imágenes del juego de entrada son escaladas a imágenes de 84×84 y transformadas en grises. La *replay memory* guarda registro de las acciones, las recompensas, las terminales y las observaciones. La política de operación es ϵ-*greedy*. En ella, una acción es seleccionada con probabilidad eps. El valor de eps cambia de 1.0 a 0.1 en el transcurso de 1M pasos. El proceso de aprendizaje DQN se inicializa con $\gamma = 0.99$. Hay un conjunto de 50,000 pasos que se usan para inicializar el proceso. El modelo objetivo se actualiza cada 10,000 pasos, y el intervalo de entrenamiento es de 4. La red se optimiza usando Adams, y su tasa de aprendizaje es 0.00025. Se pidió al entrenamiento dar 1,750,000 pasos. Un intervalo se define como 10,000 pasos se guardan las variables que aparecen en la Figura 10.2. En total hubo 4,075 episodios, cada episodio es un juego y esta compuesto de varios pasos. La pérdida comenzó con 280 puntos de observación sin valor. Luego tuvo un pico en 2000 para finalmente establizare alrededor entre 10^{-3} y 2×10^{-3}.



(a) pérdida (b) pasos (c) duración (s)

(d) episodio (e) valor medio eps (f) valor medio Q

(g) recompensa (h) error absoluto medio (i) número de pasos

Figura 10.2: Variables de salida durante el entrenamiento de *blockout* por episodio o juego transcurrido.

CAPÍTULO 11

Redes Neuronales Recurrentes

Las Redes Neuronales Recurrentes (RNN) son una especie particular de redes neuronales donde se presentan ciclos. Esta característica los hace interesantes para el estudio de fenómenos dinámicos, *i.e.*, el lenguaje visto como una secuencia de caracteres o el sonido considerado como una secuencia de tonos.

11.1. Formulación Canónica

Las redes neuronales recurrentes (RNN) permiten el procesamiento de información secuencial. La diferencia que distingue a las RNN es el ciclo de retroalimentación (ver Figura 11.1). La RNN puede ser vista como una red tradicional si desdoblamos en tiempo su arquitectura. Para cada paso de tiempo, las ecuaciones que definen la actualización están dadas por [24]

$$\begin{aligned} \mathbf{a}^t =& \mathbf{b} + \mathbf{W}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t, \\ \mathbf{h}^t =& \tanh(\mathbf{a}^t), \\ \mathbf{o}^t =& \mathbf{c} + \mathbf{V}\mathbf{h}^t, \\ \mathbf{y}^t =& \text{softmax}(\mathbf{o}^t), \end{aligned} \tag{11.1}$$

donde $\mathbf{h}^t \in \mathcal{R}^n$ corresponde al estado oculto y que actua como el predictor de la salida, en cada iteración. En este sistema, los pesos $\mathbf{W}$, $\mathbf{U}$ y $\mathbf{V}$ y los sesgos $\mathbf{b}$ y $\mathbf{c}$ son compartidos en la red y son los parámetros que hay que optimizar. Tal como en casos anteriores, la optimización se realiza con respecto a una función de pérdida, L. Por ejemplo, tal como plantea Goodfellow *et al.*[24], si la pérdida está dada por el negativo del logaritmo de la verosimilitud tenemos

$$L = \sum_t L^t = -\sum_t \log p(\hat{\mathbf{y}}^t \mid \mathbf{x}^1, \ldots, \mathbf{x}^t). \tag{11.2}$$

El entrenamiento se realiza mediante *back-propagation* a través del tiempo (BPTT). Para ello, *back-propagation* calcula las derivadas parciales. En el caso de la última capa, la derivada está dada por

$$\frac{\partial L}{\partial L^t} = 1. \tag{11.3}$$

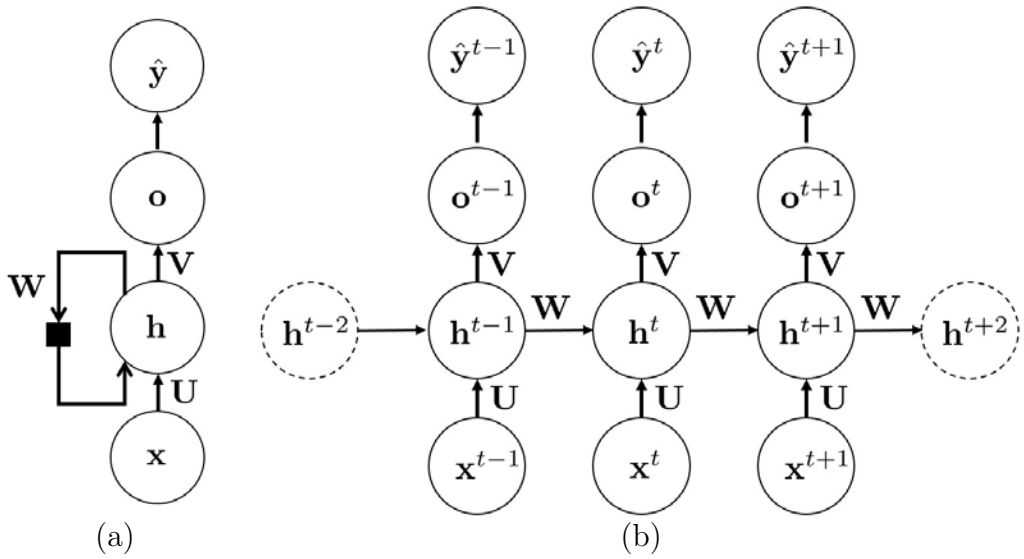


Figura 11.1: Redes Neuronales Recurrentes (RNN). La característica que distingue a la RNN es su retroalimentación, expresada con un tiempo de retraso. Aquí $\mathbf{x}$ es la entrada, $\mathbf{o}$ es la salida y $\mathbf{h}$ la(s) capa(s) oculta(s). Respectivamente, $\mathbf{W}$, $\mathbf{U}$ y $\mathbf{V}$ son los pesos asociados.

Por su lado, el gradiente de las salidas está dado por

$$\frac{\partial L}{\partial o_i^t} = \frac{\partial L}{\partial L^t}\frac{\partial L^t}{\partial o_i^t} = \hat{y}_i - y_i^t, \tag{11.4}$$

donde el gradiente de la pérdida con respecto a las salidas, $\nabla_{\mathbf{o}^\tau} L$, está dado por la conjunción de los i-resultados individuales. En el tiempo final τ, $\mathbf{h}^\tau$ tiene como único descendiente a $\mathbf{o}^\tau$. Por ello su gradiente está dado por

$$\nabla_{\mathbf{h}^\tau} L = \frac{\partial L}{\partial \mathbf{h}^\tau} = \frac{\partial \mathbf{o}^\tau}{\partial \mathbf{h}^\tau}\frac{\partial L}{\partial \mathbf{o}^\tau} = \mathbf{V}^T \nabla_{\mathbf{o}^\tau} L. \tag{11.5}$$

En las etapas del grafo más atrás en el tiempo, el gradiente de los nodos de las capas ocultas reciben la contribución de las salidas y de capas ocultas en el futuro. Es decir,

$$\begin{aligned}
\nabla_{\mathbf{h}^t} L =& \frac{\partial L}{\partial \mathbf{h}^t}, \\
=& \frac{\partial \mathbf{h}^{t+1}}{\partial \mathbf{h}^t}\frac{\partial L}{\partial \mathbf{h}^{t+1}} + \frac{\partial \mathbf{o}^t}{\partial \mathbf{h}^t}\frac{\partial L}{\partial \mathbf{o}^t}, \\
=& \mathbf{W}^T \mathrm{diag}\left(\{1-(h_i^{t+1})^2\}_i\right)\frac{\partial L}{\partial \mathbf{h}^{t+1}} + \mathbf{V}^T\frac{\partial L}{\partial \mathbf{o}^t}, \\
=& \mathbf{W}^T \mathrm{diag}\left(\{1-(h_i^{t+1})^2\}_i\right)\nabla_{\mathbf{h}^{t+1}} L + \mathbf{V}^T \nabla_{\mathbf{o}^t} L,
\end{aligned} \tag{11.6}$$

recordando que $\partial \tanh(z)/\partial z = 1 - \tanh^2(z)$, y donde $\mathrm{diag}\left(\{1-(h_i^{t+1})^2\}_i\right)$ es una matriz diagonal conteniendo los i-elementos de $\mathbf{h}^{t+1}$.

Por su lado, los parámetros correspondientes a los sesgos tienen derivadas parciales dadas

por

$$\begin{aligned}\nabla_{\mathbf{c}} L =& \frac{\partial L}{\partial \mathbf{c}} = \frac{\partial \mathbf{o}}{\partial \mathbf{c}} \frac{\partial L}{\partial \mathbf{o}}, \\ =& \sum_t \left(\frac{\partial \mathbf{o}^t}{\partial \mathbf{c}} \right)^T \frac{\partial L}{\partial \mathbf{o}^t}, \\ =& \sum_t \nabla_{\mathbf{o}^t} L,\end{aligned} \tag{11.7}$$

y

$$\begin{aligned}\nabla_{\mathbf{b}} L =& \frac{\partial L}{\partial \mathbf{b}} = \frac{\partial \mathbf{h}}{\partial \mathbf{b}} \frac{\partial L}{\partial \mathbf{h}}, \\ =& \sum_t \left(\frac{\partial \mathbf{h}^t}{\partial \mathbf{b}} \right)^T \frac{\partial L}{\partial \mathbf{h}^t}, \\ =& \sum_t \operatorname{diag}\left(\{1 - (h_i^{t+1})^2\}_i \right) \nabla_{\mathbf{h}^t} L.\end{aligned} \tag{11.8}$$

Por su lado, los parámetros correspondientes a los pesos tienen derivadas parciales dadas, para $\mathbf{V}$ por

$$\begin{aligned}\nabla_{\mathbf{V}} L =& \frac{\partial L}{\partial \mathbf{V}} = \frac{\partial L}{\partial \mathbf{o}} \frac{\partial \mathbf{o}}{\partial \mathbf{V}}, \\ =& \sum_t \sum_i \frac{\partial L}{\partial \mathbf{o}_i^t} \left(\frac{\partial \mathbf{o}_i^t}{\partial \mathbf{V}} \right), \\ =& \sum_t \left(\nabla_{\mathbf{o}^t} L \right) \left(\mathbf{h}^t \right)^T,\end{aligned} \tag{11.9}$$

para $\mathbf{W}$ por

$$\begin{aligned}\nabla_{\mathbf{W}} L =& \frac{\partial L}{\partial \mathbf{W}} = \frac{\partial L}{\partial \mathbf{h}} \frac{\partial \mathbf{h}}{\partial \mathbf{W}}, \\ =& \sum_t \sum_i \frac{\partial L}{\partial h_i^t} \left(\frac{\partial h_i^t}{\partial \mathbf{W}} \right), \\ =& \sum_t \operatorname{diag}\left(\{1 - (h_i^{t+1})^2\}_i \right) \left(\nabla_{\mathbf{h}^t} L \right) \left(\mathbf{h}^{t-1} \right)^T,\end{aligned} \tag{11.10}$$

y para $\mathbf{W}$ por

$$\begin{aligned}\nabla_{\mathbf{U}} L =& \frac{\partial L}{\partial \mathbf{U}} = \frac{\partial L}{\partial \mathbf{h}} \frac{\partial \mathbf{h}}{\partial \mathbf{U}}, \\ =& \sum_t \sum_i \frac{\partial L}{\partial h_i^t} \left(\frac{\partial h_i^t}{\partial \mathbf{U}} \right), \\ =& \sum_t \operatorname{diag}\left(\{1 - (h_i^{t+1})^2\}_i \right) \left(\nabla_{\mathbf{h}^t} L \right) \left(\mathbf{x}^t \right)^T.\end{aligned} \tag{11.11}$$

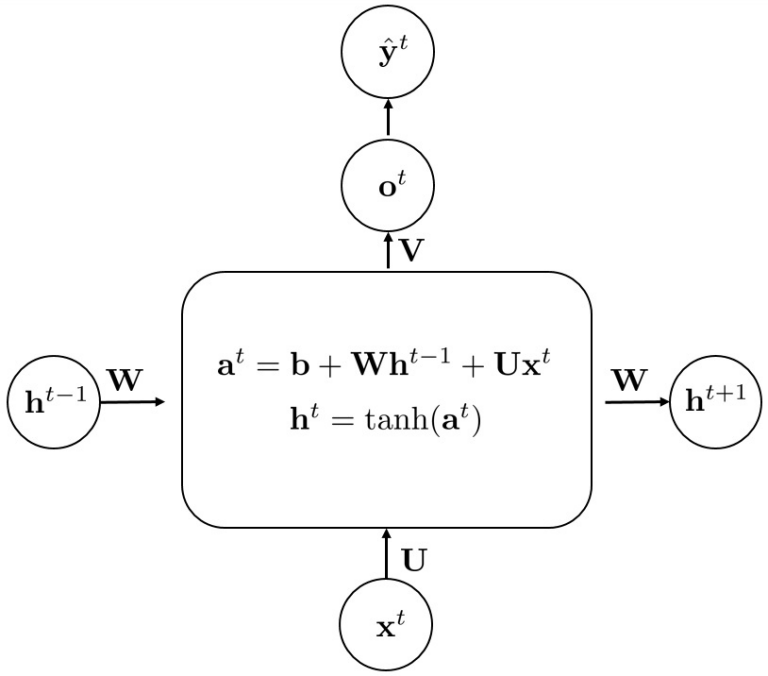


Figura 11.2: Nodo de una Redes Neuronales Recurrentes (RNN). Al interior de un node de una RNN se realiza una operación lineal que tiene como entrada el valor del nodo oculto del tiempo anterior $\mathbf{h}^{t-1}$ y el valor de la entrada en el tiempo actual $\mathbf{x}^t$. El resultado sirve como entrada para una función de activación no lineal.

El problema de las RNN es la dificultad para entrenarlas. Para entender por qué ocurre esto, considere la relación temporal en la capa oculta

$$\mathbf{h}^t = \mathbf{W}^T \mathbf{h}^{t-1}, \tag{11.12}$$

dado un cierto valor inicial para la capa oculta $\mathbf{h}^0$, para el tiempo τ se tiene la relación

$$\mathbf{h}^\tau = \left(\mathbf{W}^t\right)^T \mathbf{h}^0, \tag{11.13}$$

donde la matriz cuadrada $\mathbf{W}$ puede ser descompuesta como $\mathbf{W} = \mathbf{Q}\Lambda\mathbf{Q}^T$. Sustituyendo está relación en (11.13) tenemos

$$\mathbf{h}^\tau = \left(\mathbf{Q}^T \Lambda^t \mathbf{Q}\right) \mathbf{h}^0, \tag{11.14}$$

de donde se puede apreciar que si los eigenvalores son menores a uno, estos tienden a desaparecer, mientras que si son mayores a uno tenderán a explotar. Los métodos que entrenan una RNN pretenden mantener los eigenvalores cercanos a uno con la finalidad de evitar uno u otro fenómeno.

11.2. RNN con Compuertas

Las RNN tienen en su capa oculta la evaluación de una función lineal que recibe como entrada el valor anterior de la capa oculta $\mathbf{h}^{t-1}$ y el valor actual de la variable de entrada $\mathbf{x}^t$

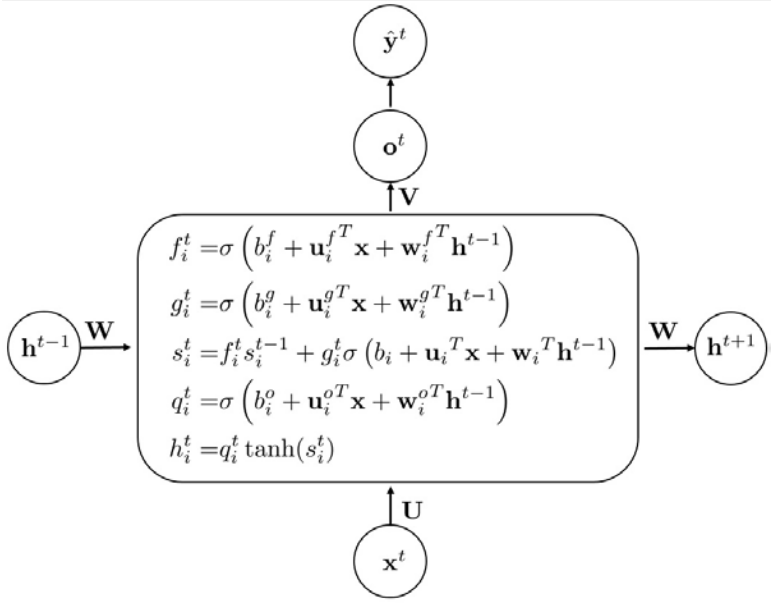


Figura 11.3: Nodo de una Redes Neuronales Recurrentes (RNN) tipo *Long-Short Term Memory* (LSTM). Estas RNN integran compuertas de olvido f_i^t (con sus respectivos pesos $\mathbf{W}^f$ y $\mathbf{U}^f$), para entrada externa g_i^t (con sus respectivos pesos $\mathbf{W}^g$ y $\mathbf{U}^g$) y para la salida q_i^t (con sus respectivos pesos $\mathbf{W}^o$ y $\mathbf{U}^o$). Las variables de entrada $\mathbf{x}^t$ y ocultas $\mathbf{h}^t = (h_1, \ldots, h_{\text{célula}})^T$ definen, junto con los pesos y compuertas, el valor del estado s_i^t, para un tiempo t y una célula i.

(ver Figura 11.2). El resultado sirve de entrada a una función de activación. La aplicación continua de estas operaciones tiene el problema de generar o desvanecimiento o saturación de los gradientes. El propósito de las compuertas es reducir las magnitudes resultantes de realizar el proceso de integración a larga escala. Las compuertas tienen pesos entrenables. Las principales formas de RNN con compuertas son las LSTM (*Long-Short Term Memory*) y las GRU (*Gated Recurrent Unit*). Ambas han mostrado un nivel similar de desempeño, aunque la GRU es considerablemente más sencilla.

La Figura 11.3 ilustra el funcionamiento de una LSTM. Las RNN tipo LSTM integran compuertas de olvido f_i^t (con sus respectivos pesos $\mathbf{W}^f$ y $\mathbf{U}^f$), para entrada externa g_i^t (con sus respectivos pesos $\mathbf{W}^g = (\mathbf{w}_1^g, \ldots, \mathbf{w}_c^g)^T$ y $\mathbf{U}^g = (\mathbf{u}_1^g, \ldots, \mathbf{u}_c^g)^T$) y para la salida q_i^t (con sus respectivos pesos $\mathbf{W}^o = (\mathbf{w}_1^o, \ldots, \mathbf{w}_c^o)^T$ y $\mathbf{U}^o = (\mathbf{u}_1^o, \ldots, \mathbf{u}_c^o)^T$). Las variables de entrada $\mathbf{x}^t$ y ocultas $\mathbf{h}^t = (h_1, \ldots, h_c)^T$ definen, junto con los pesos y compuertas, el valor del estado s_i^t, para un tiempo t y número de célula $i = 1, \ldots, c$. Las compuertas tienen una activación sigmoidal σ que responde a una combinación lineal de las entradas $\mathbf{x}^t$ y el valor anterior de la capa oculta $\mathbf{h}^{t-1}$.

La Figura 11.4 ilustra el funcionamiento de una GRU. Las RNN tipo GRU integran compuertas de actualización u_i^t (con sus respectivos pesos $\mathbf{W}^u = (\mathbf{w}_1^u, \ldots, \mathbf{w}_c^u)^T$ y $\mathbf{U}^u = (\mathbf{u}_1^u, \ldots, \mathbf{u}_c^u)^T$), y reset r_i^t (con sus respectivos pesos $\mathbf{W}^r = (\mathbf{w}_1^r, \ldots, \mathbf{w}_c^r)^T$ y $\mathbf{U}^r = (\mathbf{u}_1^r, \ldots, \mathbf{u}_c^r)^T$). Las variables de entrada $\mathbf{x}^t$, los pesos y compuertas definen el valor de la capa oculta $\mathbf{h}^t = (h_1, \ldots, h_c)^T$, para un tiempo t y una célula i. Las compuertas tienen una activación sigmoidal σ que responde a una combinación lineal de las entradas $\mathbf{x}^t$ y el valor anterior

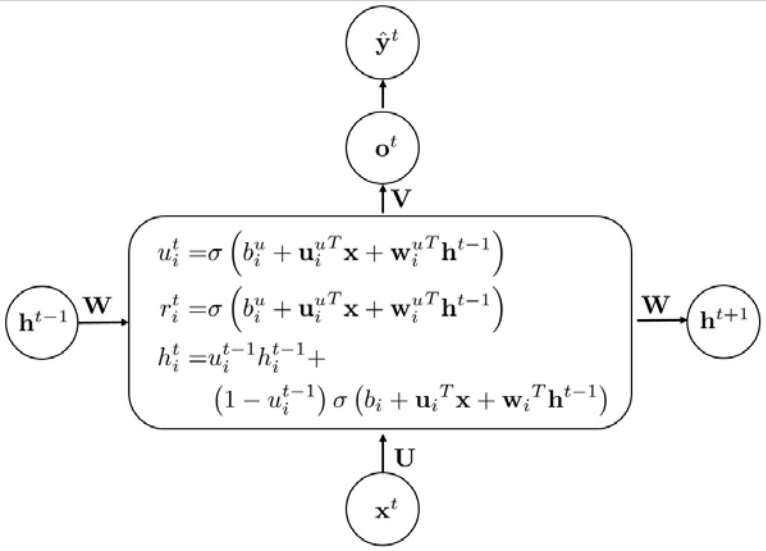


Figura 11.4: Nodo de una Redes Neuronales Recurrentes (RNN) tipo *Gated Recurrent Unit* (GRU). Estas RNN integran compuertas de actualización u_i^t (con sus respectivos pesos $\mathbf{W}^u$ y $\mathbf{U}^u$), y reset r_i^t (con sus respectivos pesos $\mathbf{W}^r$ y $\mathbf{U}^r$) . Las variables de entrada $\mathbf{x}^t$, los pesos y compuertas definen el valor de la capa oculta $\mathbf{h}^t = (h_1, \ldots, h_c)^T$, para un tiempo t y una célula i.

de la capa oculta $\mathbf{h}^{t-1}$, y, en el caso de la definición del valor de la capa oculta, la compuerta de reset.

La Figura 11.5 representa la operación de una RNN multicapa para un número variable de entradas τ. Al inicio de la operación cada capa es inicializada. Enseguida los elementos de la secuencia son alimentados, uno en cada iteración. Al final de la secuencia, la salida refleja una evaluación sobre la secuencia. Algunos ejemplos del uso de RNN incluyen el análisis de texto, y la predicción de condiciones atmosféricas.

11.3. Procesamiento de Lenguaje Natural

En un experimento probamos un dispositivo para ayudar a personas con discapacidad visual. Al término del experimento, les hicimos una entrevista para evaluar su grado de satisfacción con la interacción cuando usaban y cuando no usaban el dispositivo. Nuestro problema ahora es evaluar esas respuestas de forma automática para determinar si encontraron o no satisfacción en la interacción. Supongamos que se tiene una respuesta como: *La conversación no fue buena*. La palabra *buena* tiene una connotación positiva, pero la palabra *no* le da una connotación negativa. Para un procesamiento de lenguaje natural, las palabras de la oración se convierten a un número, *e.g.*, la posición de la palabra en un diccionario. Los números, que representan palabras, se organizan en vectores de números reales, *e.g.*, embeddings, los cuales son las entradas a la RNN. La RNN entregará una salida cuyo valor será expresado entre cero y uno, tal vez mediante una sigmoide, que representará nuestra confianza en que la oración este expresando un sentimiento negativo (con un valor cercano a cero) o positivo (con un valor cercano a uno).

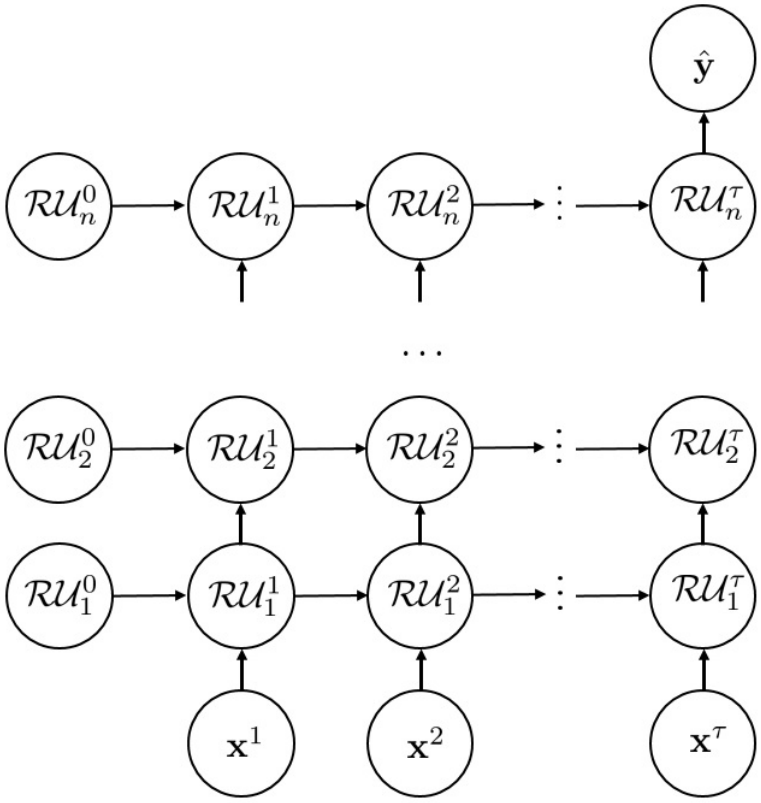


Figura 11.5: Redes Neuronales Recurrentes (RNN) multicapas. Ilustración de la operación de una RNN con n capas para un número variable de entradas. Cada nodo $\mathcal{RU}_i^t$ representa un nodo de una RNN para una entrada i en un tiempo t.

El entrenamiento de las RNN es complicado tanto por la explosión como por el desvanecimiento de los gradientes. Supóngase que se tiene textos de 100 palabras. Si inicialmente el valor del gradiente es 1.01, al final de la revisión de la cadena si el gradiente es multiplicado por si mismo 100 veces se tendrá $1.01^{100} = 2.7048$. Por otro lado, si el valor del gradiente es 0.99 se tendrá 0.366. Las RNN tipo LSTM y GRU han mostrado en la práctica tener un desempeño similar, aunque estas últimas tienen un diseño más sencillo.

En una RNN la primera capa es de embebimiento, la cual convierte un token de enteros en un vector de valores [54]. En seguida viene una sucesión de capas LSTM o GRU, cada una de esas capas con un cierto número de salidas. Esas salidas son las entradas para las siguientes capas. La capa final es alimentada a una señal de activación. El entrenamiento se realiza mediante *back-propagation* con algun métodos de optimización basado en descenso por gradiente.

11.4. Traducción de Lenguajes

Una aplicación interesante de las RNN es la traducción entre lenguajes. En el pasado, los investigadores creaban expresiones bien formadas, en donde los elementos del lenguaje, tales como verbos, sujetos y artículos tenían su lugar en la estructura de las expresiones. Hoy mediante grandes cuerpos de datos, podemos entrenar a una red neuronal recurrente a pasar de una secuencia en un lenguaje a otra secuencia en un lenguaje diferente. Para ello, por un lado se tiene un *encoder*, que mapea el texto de un lenguaje a un vector que sumariza su contenido. Este vector de sumarización alimenta a un *decoder*, el cual mapea el vector al texto en el lenguaje destino destino. Así, el texto se convierte en tokens, los cuales son embebidos. El embebimiento consiste en covertir la serie de tokens en un vector de números reales. El embebimiento tiene la propiedad interesante de que palabras con contenido semántico similar son mapeadas cerca en el espacio embebido.

11.4.1. Word2Vec

La tarea es clasificar correctamente la palabra circundada por otras tomando algunas palabras antes y algunas palabras después. Este modelo es llamado Bolsa de Palabras Continua (CBOW, en inglés). En la segunda arquitectura se utiliza la actual palabra como entrada y se busca predecir las palabras, dentro de un cierto rango, que van antes o después.

Sea el problema de transformar una entrada $\mathbf{x} \in \mathcal{R}^V$ en la salida $\mathbf{y} \in \mathcal{R}^V$, donde V es el tamaño del vocabulario, y N es el tamaño de la capa oculta. La entrada $\mathbf{x} \in \mathcal{R}^V$ está codificada como *one-hot*, *i.e.*, uno de los valores es uno y los demás cero. La arquitectura de la red tiene la forma:

$$\mathbf{x} \xrightarrow{\mathbf{U}} \mathbf{h} \xrightarrow{\mathbf{V}} \mathbf{o} \xrightarrow{\text{softmax}} \mathbf{y}. \tag{11.15}$$

Los pesos entre la capa de entrada y la capa oculta $\mathbf{h} \in \mathcal{R}^N$ son representados por una matriz $\mathbf{U} \in \mathcal{R}^{V \times N}$. Por tanto la multiplicación $\mathbf{U}^T\mathbf{x}$ básicamente representa copiar la hilera k de $\mathbf{U}$, donde k es la posición del valor 1 en $\mathbf{x}$. Entre la capa oculta $\mathbf{h}$ y la salida $\mathbf{o}$ hay, correspondientemente, una matriz $\mathbf{V}$. Enseguida, uno podría aplicar una función softmax a las salidas. En ambos casos, la función de activación es lineal.

Embebimiento de palabras es una factorización, explícita o implícita[37]. El concepto principal es la matriz de co-ocurrencia. En el esquema de redes recurrentes, se puede plantear como una red *feed-forward* entrenada con *back-propagation*, donde la función de pérdida es *entropía cruzada*.

11.4.2. Evaluación de la Calidad de la Traducción

Los trabajos actuales para evaluar la calidad de una traducción siguen usando esquemas basados en criterios antropocéntricos. Por ejemplo, Pawar y Mago[53] siguen un proceso en donde las oraciones son descompuestan en palabras. Enseguida, se determina la similaridad semántica entre pares de palabras. La similaridad de la sentencia se construye en base a la mejor relación semántica entre pares de palabras, en ambas oraciones. La propuesta que aquí se presenta se basa en la representación semántica que se logra después de interpretar la palabra en base al conjunto de sus características.

Para cada palabra w_t, para $t = 1, \ldots, T$, predice las palabras en una ventana de radio m. La función objetivo por maximizar podría ser planteada como

$$J'(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \leq j \leq m \\ j \neq 0}} p(w_{t+j} \mid w_t; \theta), \tag{11.16}$$

o equivalentemente, se busca minimizar el logaritmo de la verosimilitud

$$J(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log p(w_{t+j} \mid w_t; \theta). \tag{11.17}$$

La función que puede servir para obtener las probabilidades es softmax, definida como

$$p(\mathbf{v}_i \mid \mathbf{u}_k) = \frac{\exp\left(\mathbf{v}_i^T \mathbf{u}_k\right)}{\sum_{i=1}^{V} \exp\left(\mathbf{v}_i^T \mathbf{u}_k\right)}. \tag{11.18}$$

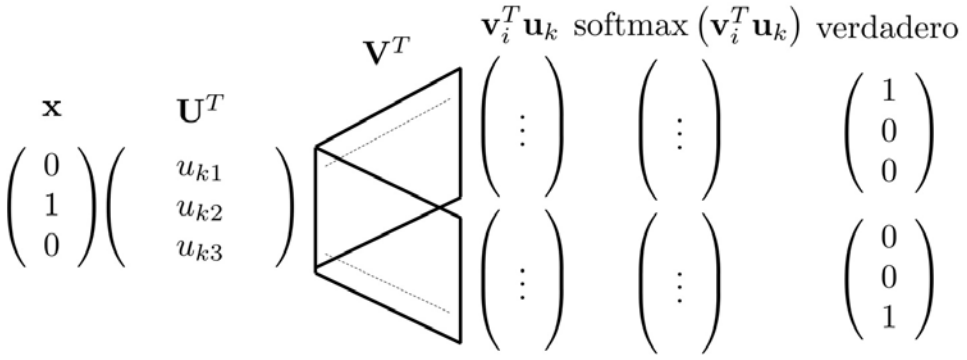


Figura 11.6: Proceso de embebimiento. Un vector $\mathbf{x}$ es multiplicado por una matriz $\mathbf{U}^T$. Como $\mathbf{x}$ está representado como un *one-hot*, el resultado corresponde a la columna $\mathbf{u}_k$ de $\mathbf{U}$. Este vector es multiplicado por la matriz $\mathbf{V}$, lo cual para cada posición equivale a multiplicarlo por la correspondiente hilera, resultando en productos $\mathbf{v}_i^T\mathbf{u}_k$. Para obtener valores de probabilidad se aplica una función softmax. Para obtener la pérdida el resultado se compara con el contexto dato por las m palabras anteriores y posteriores.

Bibliografía

[1] Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations, author=Mercer, James, journal=Philosophical Transactions of the Royal Society of London, Series A, volume=209, pages=415–446, year=1909, publisher=JSTOR.

[2] Atilim Gunes Baydin, Barak Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic Differentiation in Machine Learning: A Survey. *arXiv preprint arXiv:1502.05767*, 2015.

[3] Richard Bellman. On the Theory of Dynamic Programming. *Proceedings of the National Academy of Sciences*, 38(8):716–719, 1952.

[4] Paritosh Bhattacharya. Weibull distribution for estimating the parameters. In *Wind Energy Management*. InTech, 2011.

[5] Martin Bland and D. Bland. Statistics Notes: One and two sided tests of significance. *British Medical Journal*, 309(6949):248, 1994.

[6] Austin Blanton, Kristen Allen, Tim Miller, Nathan Kalka, and Anil Jain. A Comparison of Human and Automated Face Verification Accuracy on Unconstrained Image Sets. In *To appear in the CVPR Workshop on Biometrics*, 2016.

[7] Léon Bottou. Large-Scale Machine Learning with Stochastic Gradient Descent. In *Proceedings of COMPSTAT*, pages 177–186. Springer, 2010.

[8] Leo Breiman. Bagging Predictors. *Machine learning*, 24(2):123–140, 1996.

[9] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[10] Leo Breiman and Adele Cutler. Random forest toolbox version 5.1, 2004.

[11] Kristen Bush, Daniel R Kivlahan, Mary B McDonell, Stephan D Fihn, and Katharine A Bradley. The AUDIT Alcohol Consumption Questions (AUDIT-C): An Effective Brief Screening Test for Problem Drinking. *Archives of internal medicine*, 158(16):1789–1795, 1998.

[12] Don Cahalan, Ira Cisin, and Helen Crossley. American Drinking Practices: A National Study of Drinking Behavior and Attitudes. *Monographs of the Rutgers Center of Alcohol Studies*, 1969.

[13] Jose Carta, Penelope Ramirez, and Sergio Velazquez. A Review of Wind Speed Probability Distributions used in Wind Energy Analysis: Case Studies in the Canary Islands. *Renewable and Sustainable Energy Reviews*, 13(5):933–955, 2009.

[14] Dan Cireşan, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. A Committee of Neural Networks for Traffic Sign Classification. In *International Joint Conference on Neural Networks*, pages 1918–1921. IEEE, 2011.

[15] Christine Connolly and T. Fleiss. A Study of Efficiency and Accuracy in the Transformation from RGB to CIELAB Color Space. *IEEE Transactions on Image Processing*, 6(7):1046–1048, 1997.

[16] Antonio Criminisi and Jamie Shotton. *Decision Forests for Computer Vision and Medical Image Analysis*. Springer Science & Business Media, 2013.

[17] Richard Cutler, Thomas Edwards, Karen H Beard, Adele Cutler, Kyle Hess, Jacob Gibson, and Joshua Lawler. Random forests for classification in ecology. *Ecology*, 88(11):2783–2792, 2007.

[18] Jeffrey De Fauw, Joseph Ledsam, Bernardino Romera-Paredes, Stanislav Nikolov, Nenad Tomasev, Sam Blackwell, Harry Askham, Xavier Glorot, Brendan O'Donoghue, and Daniel Visentin. Clinically Applicable Deep Learning for Diagnosis and Referral in Retinal Disease. *Nature Medicine*, page 1, 2018.

[19] Kai-Bo Duan and Sathiya Keerthi. Which is the Best Multiclass SVM Method? An Empirical Study. In *International Workshop on Multiple Classifier Systems*, pages 278–285. Springer, 2005.

[20] Jane Elith, John R Leathwick, and Trevor Hastie. A working guide to boosted regression trees. *Journal of Animal Ecology*, 77(4):802–813, 2008.

[21] Tom Fawcett. An Introduction to ROC Analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[22] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[23] Othón González. Data-Driven Morphable Wing for Tracking and Mapping using Small UAVs. Master's thesis, Instituto Politécnico Nacional, 2016.

[24] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[25] Wayne Hall. Social Class and Survival on the SS Titanic. *Social Science and Medicine*, 22(6):687–690, 1986.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[28] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2(5):359–366, 1989.

[29] Tim Kam. Random Decision Forest. In *International Conference on Document Analysis and Recognition*, pages 14–18, 1995.

[30] Andrej Karpathy. Image Classification: Data-driven Approach, k-Nearest Neighbor, train/val/test splits, 2016.

[31] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[33] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep Learning. *Nature*, 521(7553):436–444, 2015.

[34] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[35] Yann LeCun, Léon Bottou, Genevieve Orr, and Klaus-Robert Müller. Efficient Backprop. In *Neural Networks: Tricks of the Trade*, pages 9–48. Springer, 2012.

[36] Yann LeCun and M. Ranzato. Deep Learning Tutorial. In *Tutorials in International Conference on Machine Learning*. Citeseer, 2013.

[37] Omer Levy and Yoav Goldberg. Neural Word Embedding as Implicit Matrix Factorization. In *Advances in Neural Information Processing Systems*, pages 2177–2185, 2014.

[38] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[39] Zachary Lipton, John Berkowitz, and Charles Elkan. A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv preprint arXiv:1506.00019*, 2015.

[40] Siegrid Lowel and Wolf Singer. Selection of intrinsic horizontal connections in the visual cortex by correlated neuronal activity. *Science*, 255(5041):209–212, 1992.

[41] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[42] David MacKay. *Information theory, inference and learning algorithms*. Cambridge University Press, 2003.

[43] Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Pablo Arbeláez, and Luc Van Gool. Deep Retinal Image Understanding. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 140–148. Springer, 2016.

[44] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT press, 2017.

[45] Dmytro Mishkin and Jiri Matas. All you Need is a Good Init. *arXiv preprint arXiv:1511.06422*, 2015.

[46] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei Rusu, Joel Veness, Marc Bellemare, Alex Graves, Martin Riedmiller, Andreas Fidjeland, and Georg Ostrovski. Human-Level Control through Deep Reinforcement Learning. *Nature*, 518(7540):529, 2015.

[47] Abhishek Nandy and Manisha Biswas. Reinforcement Learning with Keras, TensorFlow, and ChainerRL. In *Reinforcement Learning*, pages 129–153. Springer, 2018.

[48] Tristan Needham. A visual Explanation of Jensen's Inequality. *The American Mathematical Monthly*, 100(8):768–771, 1993.

[49] Nils Nilsson and M. Lungarella. 50 Years of AI. *Festschrift, LNAI*, 4850:9–17, 2007.

[50] Gerald North and Tatiana Erukhimova. *Atmospheric Thermodynamics: Elementary Physics and Chemistry*. Cambridge University Press, 2009.

[51] Serafim Opricovic and Gwo-Hshiung Tzeng. Compromise solution by mcdm methods: A comparative analysis of vikor and topsis. *European journal of operational research*, 156(2):445–455, 2004.

[52] Wanli Ouyang, Xiaogang Wang, Cong Zhang, and Xiaokang Yang. Factors in Finetuning Deep Model for Object Detection with Long-Tail Distribution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 864–873, 2016.

[53] Atish Pawar and Vijay Mago. Calculating the Similarity between Words and Sentences using a Lexical Database and Corpus Statistics. *arXiv preprint arXiv:1802.05667*, 2018.

[54] Magnus Erik Hvass Pedersen. TensorFlow Tutorials. `https://github.com/Hvass-Labs/TensorFlow-Tutorials`, April 2018.

[55] Hoang Pham. *Springer Handbook of Engineering Statistics*. Springer Science & Business Media, 2006.

[56] Simon Prince. *Computer Vision: Models, Learning, and Inference*. Cambridge University Press, 2012.

[57] Thomas Saaty. How to Make a Decision: The Analytic Hierarchy Process. *European Journal of Operational Research*, 48(1):9–26, 1990.

[58] Cosma Shalizi. Advanced data analysis from an elementary point of view, 2013.

[59] Claude Elwood Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

[60] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[61] Abe Sklar. Distribution Functions of n Dimensions and Margins. *Publications of the Institute of Statistics of the University of Paris*, 8:229–231, 1959.

[62] John Swets, Robyn Dawes, and John Monahan. Better Decisions through Science. *Scientific American*, page 83, 2000.

[63] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[64] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[65] Juan Terven, Bogdan Raducanu, María Elena Meza-de Luna, and Joaquín Salas. Head-Gestures Mirroring Detection in Dyadic Social Interactions with Computer Vision-based Wearable Devices. *Neurocomputing*, 175:866–876, 2016.

[66] Sergios Theodoridis, Aggelos Pikrakis, Konstantinos Koutroumbas, and Dionisis Cavouras. *Introduction to Pattern Recognition: A Matlab Approach*. Academic Press, 2010.

[67] Surya Tapas Tokdar. Laplace Approximation to the Posterior. `http://www2.stat.duke.edu/~st118/sta250/laplace.pdf`, August 2018.

[68] Carlo Tomasi and Roberto Manduchi. Bilateral Filtering for Gray and Color Images. In *International Conference on Computer Vision,*, pages 839–846. IEEE, 1998.

[69] Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

[70] David Wolpert and William Macready. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[71] Max Woodbury. Inverting Modified Matrices. *Memorandum Report*, 42(106):336, 1950.

[72] Matthew Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.

Índice alfabético