Machine Learning

Eloy Alvarado Narváez

Instituto de Estadística
Universidad de Valparaíso

Introducción

- El problema de buscar patrones
- El reconocimiento de patrones se ocupa del descubrimiento automático de regularidades en los datos mediante el uso de algoritmos, y usa estas regularidades para tomar acciones.

Ejemplo

Tomemos como ejemplo el reconocer dígitos escritos a mano.

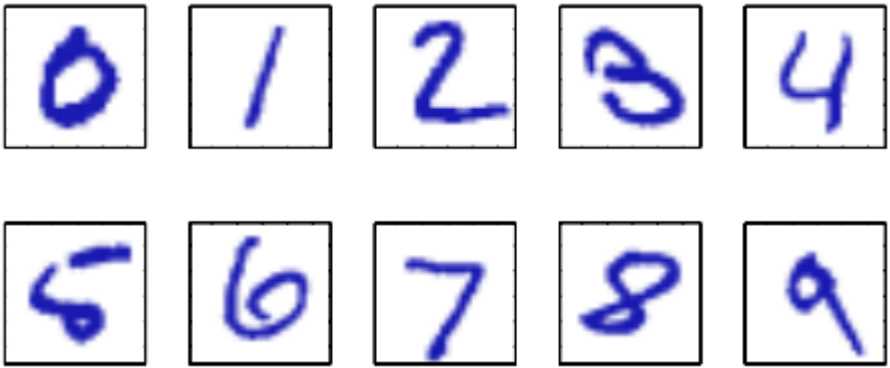


Figura 1: Ejemplos de dígitos escritos a mano tomados desde códigos postales

Estos dígitos corresponden a imágenes de 28x28 pixeles, por lo que pueden ser representados en un vector $\mathbf{x}$ que contiene 784 números reales.

El objetivo es construir una **máquina** que tome el vector $\mathbf{x}$ como entrada y produzca la identidad del dígito $0, \ldots, 9$ como salida.

Este problema es claramente no-trivial debido a la gran variedad de escrituras. Podría abordarse utilizando reglas heurísticas para distinguir los dígitos en función de las formas de los trazos, pero en la práctica, tal enfoque conduce a una proliferación de reglas y de excepciones a las reglas, etc., e invariablemente da malos resultados.

Mejores resultados pueden ser obtenidos adoptando un enfoque de **meachine learning**, en donde un conjunto grande de datos de N dígitos $\{x_1, \ldots, x_n\}$ llamados **conjunto de entrenamiento (training set)** se utiliza para ajustar los parámetros de un modelo adaptativo.

Las categorías de los dígitos en el conjunto de entrenamiento se conocen de antemano, normalmente inspeccionándolos individualmente y etiquetándolos a mano.

Podemos expresar la categoría de un dígito usando un **vector objetivo (target vector) t**, que representa la identidad del dígito correspondiente. Notar que hay un vector objetivo **t** para cada dígito de la imágen **x**.

El resultado tras aplicar el algoritmo de **machine learning** puede ser expresado como una functión $\mathbf{y}(\mathbf{x})$, que toma una nueva imagen del dígito $\mathbf{x}$ como entrada y que genera como salida un vector $\mathbf{y}$, codificada de la misma manera que los vector objetivos.

La forma exacta de la función $\mathbf{y}(\mathbf{x})$ es determinada durante la **fase de entrenamiento**, también conocida como la fase de aprendizaje, en base al conjunto de entrenamiento.

Una vez que el modelo es entrenado, este puede ser usado para identificar nuevas imágenes de dígitos, que les llamamos **conjunto de prueba (test set)**.

La habilidad de categorizar correctamente nuevos ejemplos que difieren de los utilizados en la fase de aprendizaje es conocido como **generalización**.

En la mayoría de las aplicaciones reales, las variables de entrada son típicamente preprocesadas para transformarlas a un n uevo espacio de variables donde, se espera que la problemática de reconocer patrones sea más fácil de resolver.

Por ejemplo, en el reconocimiento de dígitos escritos a mano, las imágenes de los dígitos generalmente se transforman y escalan tal que cada dígito esté contenido dentro de un cuadro de tamaño fijo. Esto reduce en gran medida la variabilidad dentro de cada clase de dígito, debido a que la localización y la escala de todos los dígitos serán las mismas, por lo que la identificación de patrones se facilitará.

La etapa de de **pre-procesamiento** es usualmente conocida como **extracción de características (feature extraction)**.

Notar que los nuevos datos, incluidos en el conjunto de entrenamiento, deben ser preprocesados de igual manera que los del conjunto de entrenamiento.

La etapa de preprocesamiento también puede ser utilizada para acelerar el cálculo del algoritmo utilizado. Se debe tener especial cuidado en esta etapa debido a que usualmente, cierta información es descartada, y si esta es imporatnte para la solución del problema, la precisión general del sistema confeccionado puede verse afectada.

Las aplicaciones en donde la entrada son los datos de entrenamiento (training set) en conjunto con sus correspondientes vectores objetivo son conocidas como **problemas de aprendizaje supervisado (supervised learning problems)**.

Los casos en donde el objetivo es asignar a cada vector de entrada una categoría, se conocen como **problemas de clasificación**.

Si se desean salidas que consisten en una o más variables continuas, entonces le llamamos **regresión**.

Las aplicaciones en donde la entrada son los datos de entrenamiento (training set) sin sus correspondientes vectores objetivos son conocidas como **problemas de aprendizaje no supervisado (unsupervised learning problems)**. Varios pueden ser los objetivos en este tipo de problemas:

- Descubrir grupos de elementos similares dentro de los datos, en este caso le llamamos **agrupamiento (clustering)**
- Estimar la distribución de los datos dentro del espacio de los datos, a esto le llamamos **estimación de densidad**
- Proyectar los datos desde un espacio multidimensional a uno de 2 o 3 dimensiones, para así poder visualizarlo, a esto le llamamos **visualización**.

Otra técnica utilizada en **machine learning** es el **aprendizaje reforzado (reinforcement learning)**, que se ocupa del problema de encontrar acciones adecuadas para tomar en una situación específica con el fin de maximizar una recompensa.

En este caso, el algoritmo de aprendizaje no recibe ejemplos de resultados óptimos (como se tienen en el aprendizaje supervisado), sino que debe descubrirlos mediante un proceso de prueba y error.

Optimización no lineal

La forma estándar de un problema de optimización no lineal es:

$$
\begin{aligned}
&\min_{x} f(x) \\
&\text{donde } g_1(x) \leq 0 \\
&\quad \vdots \\
&\quad g_l(x) \leq 0 \\
&\quad h_1(x) = 0 \\
&\quad \vdots \\
&\quad h_m(x) = 0
\end{aligned}
$$

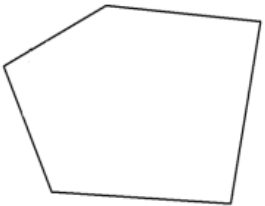
$f(x)$ le llamamos la función objetivo, usualmente a minimizar. Todas las otras restricción son de la forma $\leq$ o $=$.

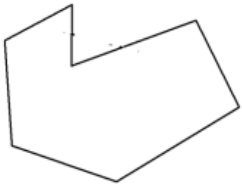
Conjunto convexo

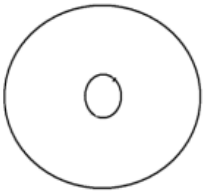
El problema **general** de optimización no lineal (donde, f, g y h pueden ser cualquier función) es extremadamente dificil de resolver. Sin embargo, si la función objetivo y las restricción son lo suficientemente *buenas*, existen algoritmos eficientes para encontrar un mínimo global.

Una de estas *buenas* condiciones, es la **convexidad**.

Existen dos definiciones para convexidad, una para conjuntos y otra para funciones. Intuitivamente, un conjunto convexo no tiene ningún agujero.







Una definición más precisa es:

Para dos puntos cualesquiera del conjunto, la línea recta que conecta esos dos puntos también se encuentra en el conjunto.

Especificamente, El conjunto X es convexo si, para cualquier $x_1 \in X, x_2 \in X$, y $\lambda \in [0,1]$, el punto $\lambda x_1 + (1-\lambda)x_2 \in X$ (este punto es una combinación convexa de x_1 y x_2).

- ¿El plano $X = \{(x,y,z) : 3x + 4y - 3z = 1\}$ es convexo?
- ¿Es la región $X = \{(x,y) : x^2 + y^2 \geq 1\}$ convexa?

Para mostrar que un conjunto es convexo, se debe mostrar que toda combinación convexa de dos puntos en el conjunto está dentro del conjunto.

Para mostrar que un conjunto no es convexo, basta mostrar un caso en donde no suceda.

Funciones convexas

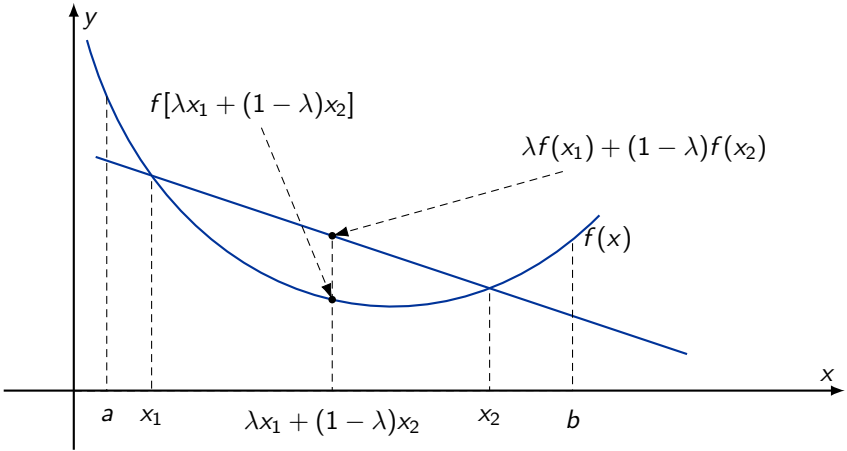
Una definición clásica que se da en cálculo (aunque acotada), es que una función unidimensional, diferenciable dos veces, es convexa si $f''(x) \geq 0$ en todo punto.

Ahora, generalizaremos este definición a más dimensiones, y a funciones que no son dos veces diferenciables.

Una función $f : X \to \mathbb{R}$ es **convexa** si, para cada $x_1, x_2 \in X$ y cada $\lambda \in (0, 1)$,

$$f((1-\lambda)x_1 + \lambda x_2)) \leq (1-\lambda)f(x_1) + \lambda f(x_2)$$

Si la desigualdad es estricta, entonces se llama **estrictamente convexa**.



- ¿Es $f(x) = |x|$ convexa?

Esta definición puede ser difícil de manejar, por lo que hay una caracterización alternativa.

Si la función es diferenciable, la convexidad puede ser caracterizada en términos de rectas tangentes a la función.

La función f es convexa si está sobre todas sus rectas tangentes.

Matemáticamente, si f es diferenciable en su dominio, entonces f es convexa si y solo si

$$f(x_2) \geq f(x_1) + f'(x_1)(x_2 - x_1)$$

para todo $x_1, x_2 \in X$.

- ¿Es x^2 convexa?

Si f es dos veces diferenciable en su dominio, entonces f es convexa si y solo si $f''(x) \geq 0$ en todas partes.

Cuando f es una función de múltiples variables, las condiciones de convexidad que involucran la primera y segunda derivada deben cambiar.

El análogo a la primera derivada es el **vector gradiente**.

$$\nabla f = [\partial f/\partial x_1 \quad \partial f/\partial x_2 \cdots \partial f/\partial x_n]^T$$

El análogo de la segunda derivada es la **matrix Hessiana**.

$$H_f = \begin{bmatrix} \partial^2 f/\partial x_1^2 & \partial^2 f/\partial x_1 \partial x_2 & \cdots & \partial^2 f/\partial x_1 \partial x_n \\ \partial^2 f/\partial x_2 \partial x_1 & \partial^2 f/\partial x_2^2 & \cdots & \partial^2 f/\partial x_2 \partial x_n \\ \vdots & \vdots & \ddots & \vdots \\ \partial^2 f/\partial x_n \partial x_1 & \partial^2 f/\partial x_n \partial x_2 & \cdots & \partial^2 f/\partial x_n^2 \end{bmatrix}$$

Para funciones multidimensionales dos veces diferenciables, f es convexa si cualquier de estas condiciones equivalentes se satisface.

1. Para todo x_1 y x_2 en X

$$f(\lambda x_2 + (1-\lambda)x_1) \leq \lambda f(x_2) + (1-\lambda)f(x_1)$$

2. Para todo x_1 y x_2 en X.

$$f(x_2) \geq f(x_1) + \nabla f(x_1)^T (x_2 - x_1)$$

3. Para todo $x \in X, H(x)$ es semidefinida positiva (esto es, $y^T H(x) y \geq 0$ para todos los vectores y).

Hay ciertas propiedades que se cumplen para las funciones convexas:

- Cualquier función lineal es convexa
- Un múltiplo no negativo de una función convexa es convexa
- La suma de funciones convexas es convexa
- La composición de funciones convexas es convexa.

Un problema de optimización convexa, es un problema de optimización en donde la función objetivo es una función convexa, y la región factible es un conjunto convexo.

Método de Lagrange

La idea del método de Lagrange o más usualmente conocido como multiplicadores de Lagrange, es mover las restricciones hacia la función objetivo, y luego resolver como si fuese un problema sin restricciones.

$$\begin{aligned} \text{mín} \quad & -x_1 - x_2 \\ \text{sujeto a} \quad & x_1^2 + x_2^2 - 1 = 0 \end{aligned}$$

¿Cómo solucionamos este problema?

Multiplicamos la restricción por λ y luego la agregamos a la función objetivo para formar la función lagrangiana:

$$\mathcal{L}(x_1, x_2, \lambda) = -x_1 - x_2 + \lambda(x_1^2 + x_2^2 - 1)$$

Los puntos estacionarios de esta función son los puntos en donde todas sus derivadas parciales son cero.

$$\frac{\partial \mathcal{L}}{\partial x_1} = -1 + 2\lambda x_1 = 0$$
$$\frac{\partial \mathcal{L}}{\partial x_x} = -1 + 2\lambda x_2 = 0$$
$$\frac{\partial \mathcal{L}}{\partial \lambda} = x_1^2 + x_2^2 - 1 = 0$$

Notar que la tercera ecuación nos entrega las restricciones iniciales.

Estas ecuaciones se resuelven cuando $x_1 = x_2 = \lambda = 1/\sqrt{2}$.

Así, la solución óptima del problema original es $x_1 = x_2 = 1/\sqrt{2}$

Si hay más de una restricción, se introduce un multiplicador adicional diferente para cada una de estas.

Tarea

Considere el siguiente problema de optimización

$$\begin{aligned} \text{mín} \quad & x^2 + y^2 + z^2 \\ \text{sujeto a} \quad & x^2 + y^2 - z^2 = 0 \\ & x - 2z - 3 = 0 \end{aligned}$$

Ejemplo

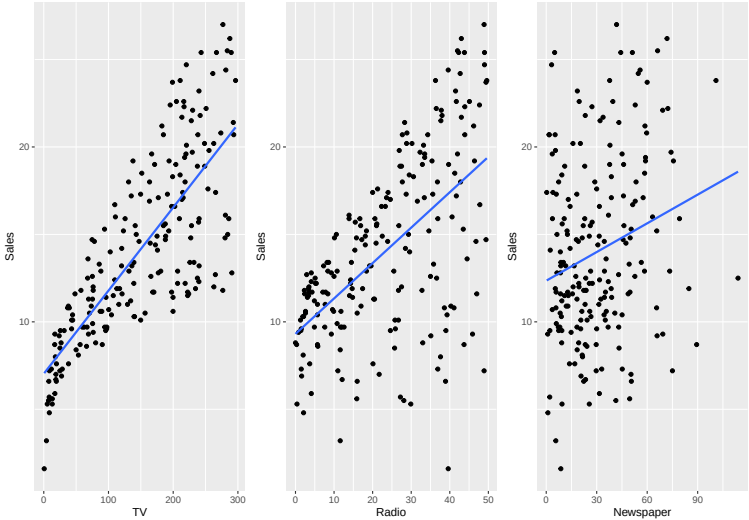
Se desea mejorar las ventas de un producto en particular. El siguiente conjunto de datos contiene datos de las ventas de aquel producto en 200 mercados diferentes, junto con el presupuesto de publicidad para el producto en cada uno de los mercados para 3 medios de publicidad: TV, radio y diario.

```
library(ISLR)
library(ggplot2)
library(gridExtra)
Advertising <- read.csv("./db/Advertising.csv")
head(Advertising)
```

```
##   X    TV Radio Newspaper Sales
## 1 1 230.1  37.8      69.2  22.1
## 2 2  44.5  39.3      45.1  10.4
## 3 3  17.2  45.9      69.3   9.3
## 4 4 151.5  41.3      58.5  18.5
## 5 5 180.8  10.8      58.4  12.9
## 6 6   8.7  48.9      75.0   7.2
```

```
p1<- ggplot(data = Advertising, mapping = aes(x = TV, y = Sales))+
    geom_point() + geom_smooth(method = "lm", se = FALSE)
p2<- ggplot(data = Advertising, mapping = aes(x = Radio, y = Sales))+
    geom_point() + geom_smooth(method = "lm", se = FALSE)
p3<- ggplot(data = Advertising, mapping = aes(x = Newspaper, y = Sales))+
    geom_point() + geom_smooth(method = "lm", se = FALSE)
```

```
grid.arrange(p1, p2, p3, nrow = 1)
```



En este ejemplo, los presupuestos son las variables de entrada (**input**) mientras que las ventas es la variable de salida (**output**). Usualmente denotaremos a las variables de entrada por la letra X, así X_1 es el presupuesto en televisión, X_2 en Radio y X_3 en periódicos.

Estas variables de entregada también se le conocen como **predictores, variables independientes, *features*** o simplemente **variables**.

La variable respuesta **Sales** es usualmente llamada **respuesta** o **variable dependiente**, y se denota por la leta Y.

En general, supongamos que observamos una variable respuesta cuantitative Y y p diferentes predictores $X_1, \ldots, X_p$. Asumiremos que existe algún tipo de relación entre Y y $X = (X_1, X_2, \ldots, X_p)$ que puede ser escrito de forma general como

$$Y = f(X) + \varepsilon$$

Donde f es una función fija de $X_1, \ldots, X_p$ y ε es un error aleatorio, que es independiente de X y tiene media cero. En lo anterior, f representa la información sistemática que X provee sobre Y.

Aprendizaje estadístico

El aprendizaje estadístico refiere al conjunto de herramientas y enfoques para **estimar** f.

¿Para qué estimar f?

Predicción

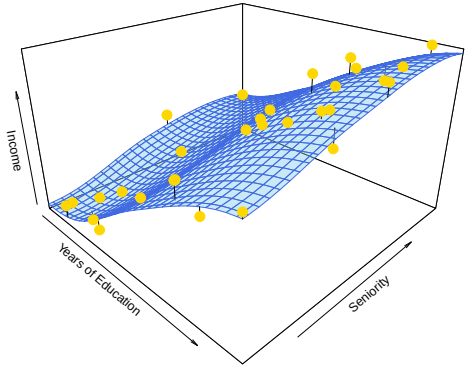
En muchas situaciones, un conjunto de variables de entrada X son fácilmente obtenibles, pero las salidas Y tienen difícil acceso. Bajo esta configuración, debido a que el promedio de los errores tiene media cero, podemos predecir Y usando:

$$\hat{Y} = \hat{f}(X)$$

donde $\hat{f}$ representa nuestra estimación para f e $\hat{Y}$ representa la predicción obtenida para Y. En este contexto, $\hat{f}$ es usualmente tratada como una **caja negra**, en el sentido que no estamos usualmente preocupados con la forma exacta de $\hat{f}$, si es que esta entrega predicciones precisas de Y.

```r
library(plot3D)
Income2<- read.csv("./db/Income2.csv")
# Ajuste
fit_2_3_loess <- loess(Income ~ Education + Seniority, data = Income2)
# Predicción de valores
x.pred <- seq(min(Income2$Education), max(Income2$Education), length.out = 30)
y.pred <- seq(min(Income2$Seniority), max(Income2$Seniority), length.out = 30)
xy     <- expand.grid(Education = x.pred, Seniority = y.pred)
z.pred <- matrix(predict(fit_2_3_loess, newdata = xy), nrow = 30, ncol = 30)
```

```
Income2 %>%
  scatter3D(
    type = "p",
    x = Income2$Education,
    y = Income2$Seniority,
    z = Income2$Income,
    colvar = NA, pch = 19, col = "gold", cex = 1.75,
    phi = 25, theta = 45, expand = 0.6,
    xlab = "Years of Education", ylab = "Seniority", zlab = "Income",
    panel.first = scatter3D(x = Income2$Education,y = Income2$Seniority,
    z = Income2$Income,colvar = NA, col = "black", add = T,
    surf = list(x = x.pred, y = y.pred, z = z.pred,
    fit = predict(fit_2_3_loess), facets = T, col = "skyblue",
    border = "royalblue", alpha = 0.45)))
```


Income
Years of Education
Seniority

Consideremos que un estimador $\hat{f}$ y un conjunto de variables X entregan la predicción $\hat{Y} = \hat{f}(X)$. Asumiendo que $\hat{f}$ y X son fijos, entonces se tiene:

$$\begin{aligned}\mathbb{E}(Y - \hat{Y})^2 &= \mathbb{E}(f(X) + \varepsilon - \hat{f}(X))^2 \\ &= \underbrace{[f(X) - \hat{f}(X)]^2}_{\text{Reducible}} + \underbrace{\mathbb{V}(\varepsilon)}_{\text{Irreducible}}\end{aligned}$$

Nosotros nos concentraremos en técnicas para estimar f con el fin de poder minimizar el error reducible.

Inferencia

Usualmente estamos interesados en entender la forma en que Y se ve afectada conforme $X_1, \ldots, X_p$ cambia. En este tipo de situaciones, deseamos estimar f, pero nuestro objetivo no es necesariamente hacer predicciones para Y. En cambio, se quiere entender la relación entre X e Y, por lo que ya no podemos tratar $\hat{f}$ como una caja negra, debido a que para poder explicar el fenómeno debemos tener una **forma exacta**. Usualmente nos preguntamos:

- ¿Qué predictores están asociados con la respuesta?
- ¿Cuál es la relación entre la respuesta y cada predictor?
- ¿La relación entre Y y cada predictor ser explicada adecuadamente usando una ecuación lineal o la relación es más complicada?

¿Cómo estimamos f?

A lo largo del curso, veremos enfoques lineales y no lineales para estimar f. Estos métodos usualmente comparten ciertas características.

En general, la mayoría de las técnicas de aprendizaje estadístico pueden ser categorizadas como **paramétricas** o **no-paramétricas**.

Métodos paramétricos

Este enfoque tiene dos pasos y se base en modelos que reducen el problema de estimar f a estimar un conjunto de parámetros.

Pros

- Es mucho más fácil que ajustar una función arbitraria cualquiera

Contras

- El modelo usualmente no seguirá la forma real de f
- Si el ajuste está muy lejano a la forma real, la estimación será mala
- Se puede caer en sobreajuste

¿Cuales serían los pasos de un enfoque paramétrico?

1. Asumir la forma de f
2. Realizar un proceso que ajuste el conjunto de datos (**training set**) para el modelo

Métodos no paramétricos

El enfoque no paramétrico se caracteriza por no asumir la forma de f, pero en lugar de eso intenta obtener una estimación de f que sea lo más cercano al conjunto de datos sin llegar a un sobreajuste.

Pros

- Al no asumir nada sobre f, estos métodos permiten un vasto rango de formas que se ajustan con precisión a f

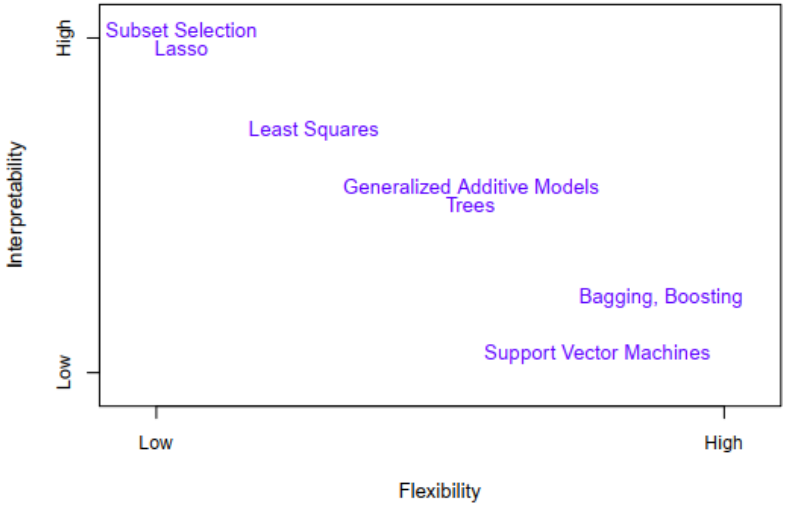
Contras

- Un gran número de datos es necesario para estimar de forma precisa f, mucho más que bajo un enfoque paramétrico.

Compensación entre precisión vs interpretabilidad

Como sabemos hay métodos de aprendizaje estadístico que son menos flexibles que otros, por ejemplo la regresión lineal. Sin embargo, existen razones para escoger estas metodologías en vez de una más flexible.

- Si la inferencia es nuestro principal objetivo, los modelos más restrictivos son recomendados debido a que la relación entre X e Y es fácilmente interpretable.
- Métodos más flexibles usualmente llegar a estimación más complejas que dificultan el análisis de alguna relación individual entre un predictor y la variable respuesta.
- Incluso cuando la predicción es el único objetivo, modelos más restrictivos pueden entregar mayor precisión que la mayoría de los métodos más flexible, debido a que estos últimos pueden sobreajustar.


Subset Selection
Lasso
Least Squares
Generalized Additive Models
Trees
Bagging, Boosting
Support Vector Machines
High
Low
Interpretability
Low
High
Flexibility

Teorema del No-Free-Lunch

¿Por qué no simplemente elegimos el **mejor** método para todos los problemas?

El teorema de No-Free-Lunch establece que todos los algoritmos de optimización se desempeñan igualmente bien cuando su desempeño es promediado sobre todas las funciones objetivos posibles.

Compromiso sesgo-varianza

Una de las herramientas que tenemos para cuantificar que tan bueno es nuestro ajuste es el Error cuadrático medio, lo notamos por sus siglas en inglés **MSE**. Para un valor x_0 dado, es posible mostrar que el error cuadrático medio se puede descomponer de la forma

$$\mathbb{E}\left(y_0 - \hat{f}(x_0)\right)^2 = \mathbb{V}(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + \mathbb{V}(\varepsilon)$$

En donde el lado izquierdo representa el error cuadrado medio esperado cuando se estima f y se evalúan en el punto x_0.

De la ecuación anterior se desprende que para minimizar el error cuadrático medio se debe seleccionar una metodología que simultáneamente logre una varianza baja y un bajo sesgo.

A esta relación le llamamos un compromiso, debido a que es fácil obtener un método con extremadamente bajo sesgo pero varianza alta o un modelo con baja varianza pero alto sesgo.

Como regla general, si se utilizan metodologías más flexibles, la varianza crecerá y el sesgo disminuirá.

Métodos supervisados

Como hemos mencionado a lo largo del curso, una regresión lineal simple asume que la variable respuesta Y es **cuantitativa**, pero en muchas situaciones esta es **cualitativa** (también referida como categórica). En lo que sigue, veremos métodos para predecir respuestas cualitativas, más comúnmente llamado **clasificación**.

Existen mucha técnicas de clasificación o **clasificadores**, que se pueden usar para predecir una variable cualitativa. Entre ellos se encuentras:

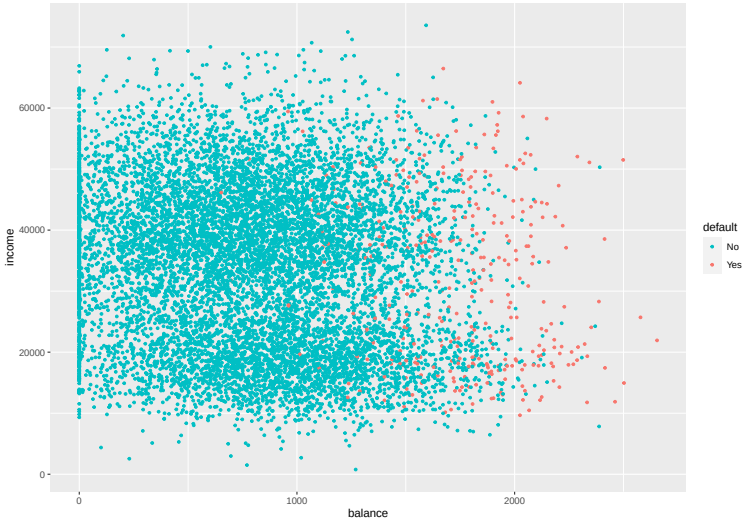
- Regresión logística
- Análisis discriminante lineal
- *k-NN (k- nearest neighbors / k-vecinos cercanos)*
- Modelos generalizados aditivos
- Árboles y bosques aleatorios
- Boosting
- SVM

Ejemplo

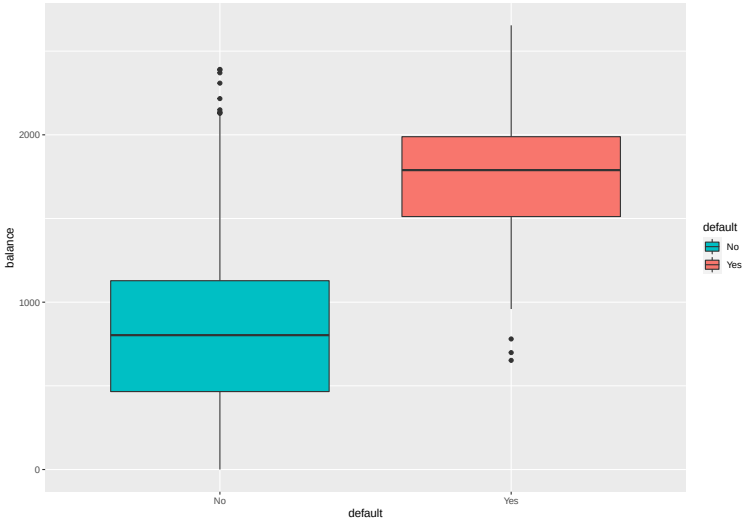
```
data<-Default
head(data)
```

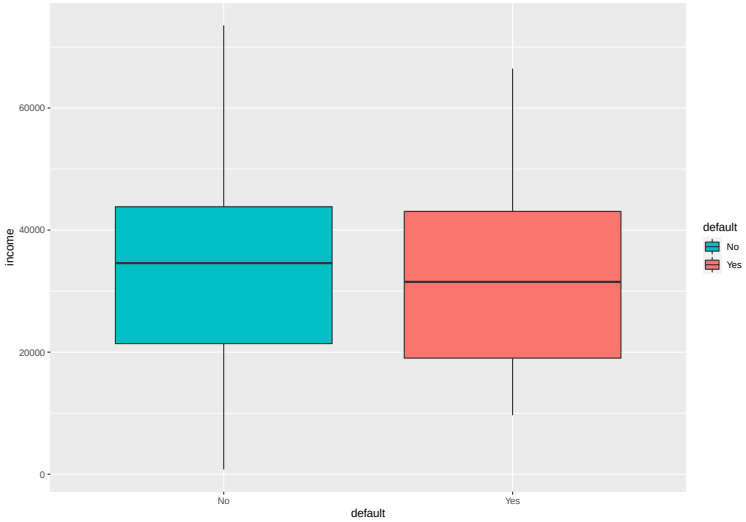
```
##   default student   balance    income
## 1      No      No  729.5265 44361.625
## 2      No     Yes  817.1804 12106.135
## 3      No      No 1073.5492 31767.139
## 4      No      No  529.2506 35704.494
## 5      No      No  785.6559 38463.496
## 6      No     Yes  919.5885  7491.559
```

```
ggplot(data) +
 aes(x = balance, y = income, colour = default) +
 geom_point(shape = "bullet", size = 1.5) +
 scale_color_hue(direction = -1) +
 theme_gray()
```



Universidad de Valparaíso CHILE


balance
2000
1000
0
No
Yes
default
default
No
Yes


60000
40000
20000
0
income
No
Yes
default
default
No
Yes

¿Por qué no usar una regresión lineal?

Supongamos que se intenta predecir la condición médica de un paciente en la sala de emergencia con base a sus síntomas. Para simplificar, imaginemos que sólo que tienen 3 posibles diagnósticos: accidente cardiovascular, sobredosis y ataque epiléptico. Por lo que podríamos clasificar la variable respuesta como

$$Y = \begin{cases} 1 & \text{si Accidente cardiovascular} \\ 2 & \text{si Sobredosis} \\ 3 & \text{si Ataque epiléptico} \end{cases}$$

Usando esta codificación, se puede usar el método de mínimos cuadrados para ajustar una regresión lineal para predecir Y en base a los predictores $X_1, \ldots, X_p$.

Desafortunadamente, esta codificación implica un ordenamiento de las salidas, estableciendo sobredosis entre accidente cardiovascular y Ataque epiléptico, e inherentemente afirmando que la diferencia entre categorías contiguas son la misma.

Es claro notar que si usamos otra codificación, el ajuste de regresión lineal obtenido será diferente al primero. En general, no hay una forma natural de convertir una variable respuesta cualitativa con más de dos niveles en una variable cuantitativa que esté lista para hacer una regresión lineal.

En el caso de variable respuesta binaria, la situación es algo más favorable, debido a que si se cambia la codificación, el ajuste de regresión obtenido será el mismo. Sin embargo, el método de mínimos cuadrados no tiene sentido, provocando que algunas de nuestras estimación estén fuera del intervalo [0,1], haciendo difícil la interpretación de las probabilidades.

Lo anterior debido a que se puede mostrar que el $X\hat{\beta}$ obtenido con la regresión lineal con codificación binaria, es simplemente una estimación de $\mathbb{P}(\text{ Sobredosis })$ si la codificación es

$$Y = \begin{cases} 0 & \text{si Accidente cardiovascular} \\ 1 & \text{si Sobredosis} \end{cases}$$

Regresión logística

Usando el mismo conjunto de datos `Default`, donde la variable respuesta `default` cae dentro de dos categorías `Yes` y `No`. En vez de modelar la respuesta Y directamente, la **regresión logística** modela la probabilidad que Y pertenezca a una categoría particular.

Para el conjunto de datos `Default`, la regresión logística modela la probabilidad de que haya default (morosidad). Por ejemplo, la probabilidad de default dado cierto `balance` puede ser escrito como

$$\mathbb{P}(\text{default} = \text{Yes}|\text{balance})$$

Los valores de esta probabilidad, que la abreviamos como $p(\text{balance})$, estarán entre 0 y 1. Por lo que para un valor particular de `balance`, se puede hacer una predicción para `default`. Por ejemplo, se podría predecir que `default=Yes` para cualquier individuo cuyo $p(\text{balance}) > 0.5$. Alternativamente, si una compañía quisiese ser más conservador en la predicción, podría definir $p(\text{balance}) > 0.1$.

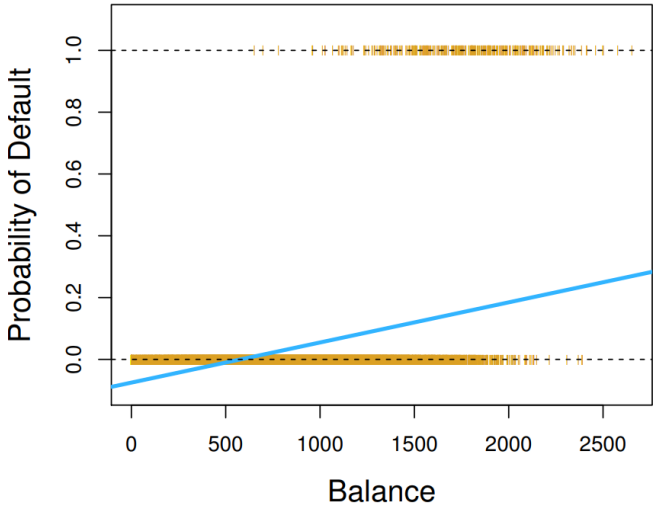
Modelo logístico

¿Cómo deberíamos modelar la relación entre $p(X) = \mathbb{P}(Y = 1|X)$ y X?

Podemos utilizar un enfoque de regresión lineal para representar estar probabilidades, esto es:

$$p(X) = \beta_0 + \beta_1 X$$

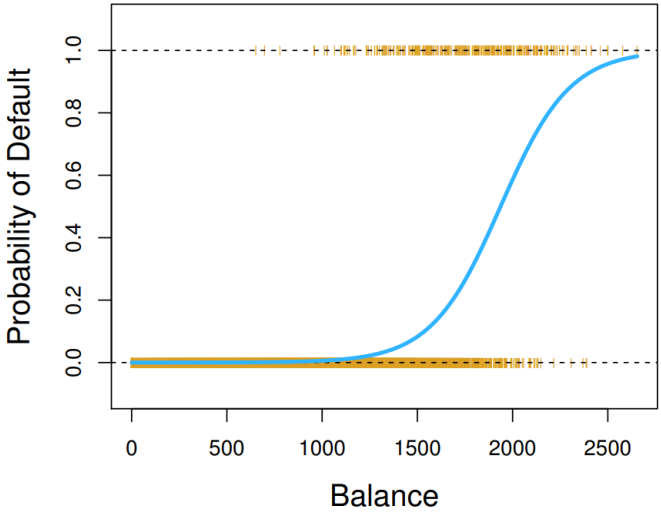
Si usamos este enfoque para predecir `default=Yes` usando `balance`, entonces obtendremos el siguiente modelo (izquierda).


Probability of Default
0.0
0.2
0.4
0.6
0.8
1.0
0
500
1000
1500
2000
2500
Balance

Para evitar lo anterior, debemos modelar $p(X)$ usando una función que entregue salidas entre 0 y 1 para todos los valores de X. Muchas funciones cumplen estas condiciones. En una **regresión logística**, usamos la *función logística*.

$$p(X) = \frac{\exp(\beta_0 + \beta_1 X)}{1 + \exp(\beta_0 + \beta_1 X)}$$

Para ajustar el modelo anterior, usamos máxima verosimilitud


Probability of Default
0.0
0.2
0.4
0.6
0.8
1.0
0
500
1000
1500
2000
2500
Balance

Manipulando un poco la fórmula anterior, se tiene que

$$\frac{p(X)}{1-p(X)} = \exp(\beta_0 + \beta_1 X)$$

La cantidad $\frac{p(X)}{1-p(X)}$ se le llaman **odds**, que pueden toman cualquier valor en $\mathbb{R}^+$. Valores cercanos a cero y tendiendo a infinito, indican muy baja y alta probabilidad de `default`, respectivamente.

Tomando el logaritmo en ambos lados, se tiene

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

a esta cantidad la llamamos **log-odds** o **logit**. Notamos que el modelo de regresión logística tiene un logit lineal en X.

Estimación de los coeficientes de regresión

Los coeficiente β_0 y β_1 en la ecuación

$$p(X) = \frac{\exp(\beta_0 + \beta_1 X)}{1 + \exp(\beta_0 + \beta_1 X)}$$

son desconocidos, por lo que deben ser estimados basándose en los datos de entrenamiento. Si bien podríamos ocupar una metodología de métodos cuadrados no lineales para ajustar el modelo:

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X$$

La metodología de máxima verosimilitud es usualmente preferida, debido a que tiene mejores propiedades estadísticas.

Formalmente, definimos la **función de verosimilitud** como:

$$\ell(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'}))$$

Las estimaciones $\hat{\beta}_0$ y $\hat{\beta}_1$ son escogidos para maximizar la función de verosimilitud.

Ejemplo

```
logit <- glm(default ~ balance, data = data, family = "binomial")
summary(logit)
```

```
##
## Call:
## glm(formula = default ~ balance, family = "binomial", data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2697  -0.1465  -0.0589  -0.0221   3.7589
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.065e+01  3.612e-01  -29.49   <2e-16 ***
## balance      5.499e-03  2.204e-04   24.95   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1596.5  on 9998  degrees of freedom
## AIC: 1600.5
##
## Number of Fisher Scoring iterations: 8
```

Predicciones

Una vez que los coeficientes han sido estimados, lo que resta es calcular la probabilidad de `default` para una `balance` dado. Por ejemplo, la predicción para una persona con balance \$1000 es

$$\hat{p}(X) = \frac{\exp(-10.65 + 0.0055 \times 1000)}{1 + \exp(-10.65 + 0.0055 \times 1000)} \approx 0.00576$$

que es bajo 1 %. En contraste con alguien que adeuda \$2000, en cuyo casi $\hat{p}(X) = 0.586$.

Si utilizamos *dummy variables* para el predictor `student` codificado como 0 y 1. tendremos el siguiente ajuste

```
logit_dummy<-glm(default ~ student, data = data, family = "binomial")
summary(logit_dummy)
```

```
##
## Call:
## glm(formula = default ~ student, family = "binomial", data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2970  -0.2970  -0.2434  -0.2434   2.6585
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.50413    0.07071  -49.55  < 2e-16 ***
## studentYes   0.40489    0.11502    3.52 0.000431 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 2908.7  on 9998  degrees of freedom
## AIC: 2912.7
##
## Number of Fisher Scoring iterations: 6
```

Así, podemos calcular las probabilidades

$$\mathbb{P}\left(\text{default=Yes} \mid \text{student=Yes}\right) = \frac{\exp(-3.5041 + 0.4049 \times 1)}{1 + \exp(-3.5041 + 0.4049 \times 1)} \approx 0.0431$$

y,

$$\mathbb{P}\left(\text{default=Yes} \mid \text{student=No}\right) = \frac{\exp(-3.5041 + 0.4049 \times 0)}{1 + \exp(-3.5041 + 0.4049 \times 0)} \approx 0.0292$$

Regresión logística múltiple

Ahora consideramos el problema de predecir una respuesta binaria usando múltiples predictores. La extensión natural del modelo de regresión es

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

donde $X = (X_1, \ldots, X_p)$ son p predictores. La ecuación anterior la podemos reescribir como

$$p(X) = \frac{\exp(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p)}{1 + \exp(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p)}$$

Al igual que antes, usamos método de máxima verosimilitud para estimar β

Ejemplo

```
logit2 <- glm(default ~ balance + student + income, data = data,
              family = "binomial")
summary(logit2)
```

```
##
## Call:
## glm(formula = default ~ balance + student + income, family = "binomial",
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4691  -0.1418  -0.0557  -0.0203   3.7383
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
## balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
## studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
## income       3.033e-06  8.203e-06   0.370  0.71152
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1571.5  on 9996  degrees of freedom
## AIC: 1579.5
##
## Number of Fisher Scoring iterations: 8
```

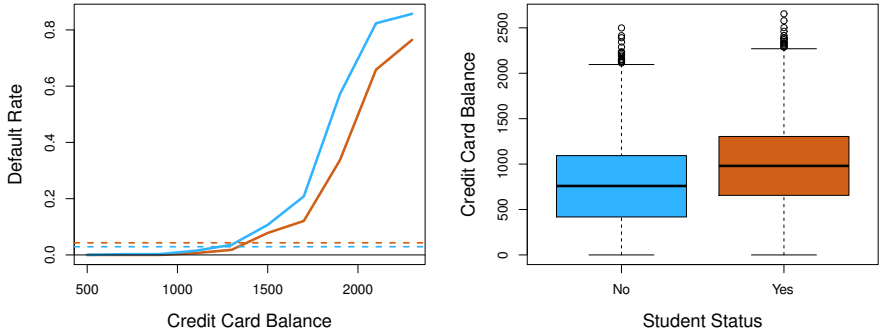


Figura 2: Tasas de default, Estudiantes en naranjo, No-Estudiantes en azul.

Regresión logística para > 2 clases en la respuesta

En el caso en que tengamos más de dos clases en la variable respuesta, es posible extender la regresión lineal. En el ejemplo de determinación de diagnóstico en una sala de emergencia se tenían las categorías accidente cardiovascular, sobredosis y ataque epiléptico, por lo que se desearía modelar

$$\mathbb{P}(Y = \text{ acc. card. } |X)$$

y

$$\mathbb{P}(Y = \text{ sobredosis } |X)$$

siendo el remanente,

$$\mathbb{P}(Y = \text{ ataque epiléptico } |X) = 1 - \mathbb{P}(Y = \text{ acc. card } |X) - \mathbb{P}(Y = \text{ sobredosis } |X)$$

Si bien es posible la extensión, en la práctica no es frecuentemente usado, pues se prefiere realizar un **análisis discriminante**.

Análisis discriminante lineal

La regresión logística que vimos antes involucra modelar directamente $\mathbb{P}(Y = k|X = x)$ usando la función logística dada por

$$p(X) = \frac{\exp(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p)}{1 + \exp(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p)}$$

para el caso de dos clases en la variable respuesta. En lo que sigue, consideramos una manera alternativa y menos directa para estimar estas probabilidades. En esta metodología, modelamos la distribución de los predictores X por separado en cada una de las categorías de la variable respuesta (Y), y luego usamos el teorema de Bayes para convertir estos resultados en estimaciones de $\mathbb{P}(Y = k|X = x)$.

Cuando estas distribuciones se asumen normales, la forma de este modelo es muy similar a una regresión logística.

Teorema de Bayes para clasificación

Supongamos que queremos clasificar una observación entre K clases, donde $K \geq 2$. Esto es, que la variable respuesta Y puede tomar K posibles valores distintos y no-ordenados.

Sea π_k la probabilidad *apriori* que una observación escogida aleatoriamente provenga de la clase $k-$ésima. Sea $f_k(X) = \mathbb{P}(X = x|Y = k)$ la **función de densidad** de X para una observación que proviene de la clase $k-$ésima. Luego, por el teorema de Bayes se tiene

$$\mathbb{P}(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

al igual que antes usamos la notación $p_k(X) = \mathbb{P}(Y = k|X)$.

La idea general, es estimar no estimar $p_k(X)$ directamente, sino estimar π_k y f_k para obtener lo deseado.

Usualmente π_k es fácil de obtener si se tiene una muestra aleatoria de Y, pues obtenemos estas estimaciones como las proporciones de cada clase.

En cambio, estimar $f_k(X)$ tiende a ser más difícil, a menos que se asuman formas simples para las densidades.

Llamamos a la cantidad $p_k(x)$ la probabilidad *posterior* que una observación $X = x$ pertenezca a la clase $k-$ésima.

Análisis discriminante lineal con $p = 1$

Primero asumiremos que $p = 1$, es decir, sólo tenemos un predictor. Deseamos obtener una estimación para $f_k(x)$ para utilizarlo en la ecuación

$$\mathbb{P}(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

y así poder estimar $p_k(x)$. Para poder estimar f_k, primero debemos asumir su forma, por lo que asumiremos que f_k es *Gaussiana*. Por lo que,

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2\right)$$

donde μ_k y σ_k^2 son la media y la varianza de la clase $k-$ésima. Por ahora, asumiremos que $\sigma_1^2 = \cdots = \sigma_K^2 = \sigma^2$

Por lo anterior, se tendrá

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(x-\mu_k)^2\right)}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(x-\mu_l)^2\right)}$$

El clasificador Bayesiano asigna una observacion $X = x$ a la clase que su $p_k(x)$ es más grande. Si tomamos el logaritmo y arreglamos términos en la expresión anterior, se tiene que el proceso es equivalente a asignar la observación a la clase en la que

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log \pi_k$$

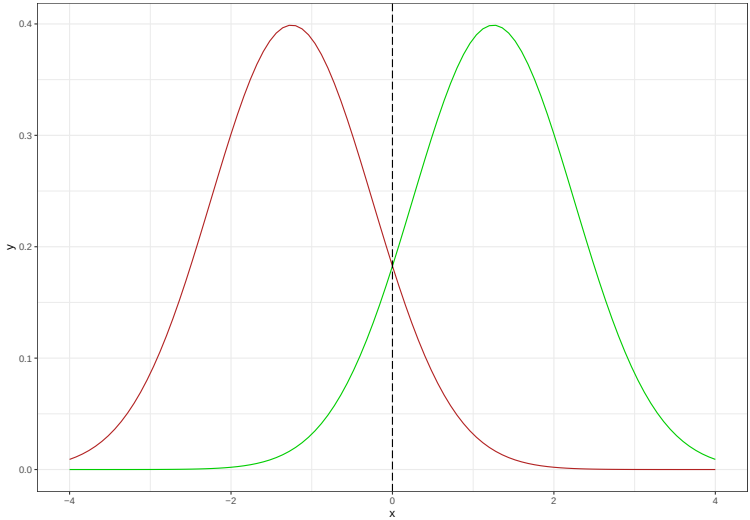
es más grande.

Por ejemplo, si $K = 2$ Y $\pi_1 = \pi_2$, entonces el clasificador Bayesiano asigna una observación a la clase 1 si $2x(\mu_1 - \mu_2) > \mu_1^2 - \mu_2^2$ y a la clase 2 en caso contrario. En este caso, el límite de decisión de Bayes (*Bayes decision boundary*) corresponde al punto donde

$$x = \frac{\mu_1^2 - \mu_2^2}{2(\mu_1 - \mu_2)} = \frac{\mu_1 + \mu_2}{2}$$

Llamamos a este punto el punto (o área) en donde la clasificación es ambigua.

```
p4<-ggplot(data.frame(x = c(-4, 4)), aes(x)) +
stat_function(fun = dnorm, args = list(mean = -1.25, sd = 1),
              color = "firebrick") +
stat_function(fun = dnorm, args = list(mean = 1.25, sd = 1), color = "green3") +
geom_vline(xintercept = 0, linetype = "longdash") +
theme_bw()
```


0.4
0.3
0.2
0.1
0.0
y
−4
−2
0
2
4
x

El análisis discriminante lineal (LDA) aproxima el clasificador bayesiano ingresando estimaciones para pi_k, μ_k y σ^2 en $\delta_k(x)$. Particularmente, las siguientes estimaciones son usadas.

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

y,

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=K} (x_i - \hat{\mu}_k)^2$$

donde n es el número total de observaciones en el conjunto de entrenamiento, n_k es el número de observaciones en el conjunto de entrenamiento en la clase $k-$ésima.

En el caso de que no tengamos información de $\pi_1, \ldots, \pi_K$, el análisis discriminante lineal estima π_k usando la proporción de las observaciones en el conjunto de entrenamiento que pertenece a la clase $k-$ésima. Esto es,

$$\hat{\pi}_k = \frac{n_k}{n}$$

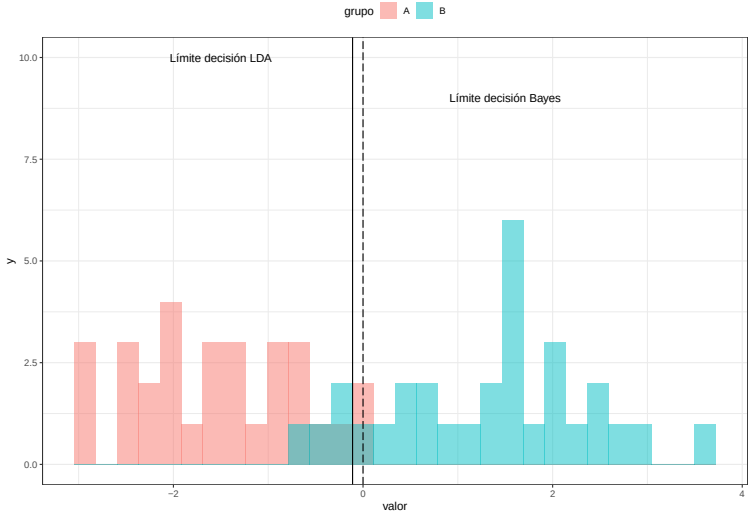
El clasificador **LDA** reemplaza las estimaciones anteriores en $\delta_k(x)$ y asigna una observación $X = x$ a la clase en la cual

$$\hat{\delta}_k = x\frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + \log \hat{\pi}_k$$

es más grande. El nombre de **lineal** viene de la linealidad de la *función discriminante* $\hat{\delta}_k$ para x.

```r
set.seed(411)
grupo_a <- rnorm(n = 30, mean = -1.25, sd = 1)
grupo_b <- rnorm(n = 30, mean = 1.25, sd = 1)
datos <- data.frame(valor = c(grupo_a, grupo_b),
                    grupo = rep(c("A","B"), each = 30))

p5<-ggplot(data = datos, aes(x = valor, fill = grupo)) +
geom_histogram(alpha = 0.5, position = "identity") +
geom_vline(xintercept = 0, linetype = "longdash") +
geom_vline(xintercept = (mean(grupo_a) + mean(grupo_b))/2)  +
annotate(geom = "text", x = 1.5, y = 9, label = "Límite decisión Bayes") +
annotate(geom = "text", x = -1.5, y = 10, label = "Límite decisión LDA") +
theme_bw() +
theme(legend.position = "top")
```


grupo
A
B
Límite decisión LDA
Límite decisión Bayes
10.0
7.5
5.0
2.5
0.0
y
−2
0
2
4
valor

Análisis discriminante liean con $p > 1$

tbd