

Capítulo 4. Classification

Diego Felipe Bobadilla Restrepo

Karol Andrea Ujueta Rojas

Yosef Shmuel Guevara Salamanca

June 1, 2021

Ejercicios

1. Using a little bit of algebra, prove that (4.2) is equivalent to (4.3). In other words, the logistic function representation and logit representation for the logistic regression model are equivalent.

$$p(X) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}} \quad (4.2)$$

$$\frac{p(X)}{1-p(X)} = e^{\beta_0 + \beta_1 x} \quad (4.23)$$

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

$$p(X)(1 + \exp(\beta_0 + \beta_1 x)) = \exp(\beta_0 + \beta_1 x)$$

$$p(X) = \exp(\beta_0 + \beta_1 x) - p(X) \exp(\beta_0 + \beta_1 x)$$

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

2. It was stated in the text that classifying an observation to the class for which (4.12) is largest is equivalent to classifying an observation to the class for which (4.13) is largest. Prove that this is the case. In other words, under the assumption that the observations in the k th class are drawn from a $N(\mu_k, \sigma^2)$ distribution, the Bayes' classifier assigns an observation to the class for which the discriminant function is maximized.

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x - \mu_k)^2)}{\sum_{t=1}^k \pi_t \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x - \mu_t)^2)} \quad (4.12)$$

$$\delta_k(x) = x \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k) \quad (4.13)$$

Bajo el supuesto de que las observaciones de una clase k tienen distribución normal de la forma $N(\mu_k, \sigma^2)$ entonces $f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2)$.

Usando el teorema de Bayes se quiere encontrar un k que maximice la probabilidad $p_k(x) = P_r(Y = k|X = x)$. Esto se puede reescribir de la siguiente forma:

$$\begin{aligned}
p_k(x) &= P(Y = k|X = x) \\
&= \frac{P(X = x|Y = k) \cdot P(Y = k)}{P(X = x)} \\
&= \frac{P(Y = k) \cdot P(X = x|Y = k)}{\sum_{l=1}^K P(X = x|Y = l) \cdot P(Y = l)} \\
&= \frac{\pi_k \cdot f_k(x)}{\sum_{l=1}^K \pi_l \cdot f_l(x)} \quad \text{donde } \pi_k = P(Y = k) \\
&= \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x - \mu_k)^2)}{\sum_{l=1}^K \pi_l \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2)} \\
&= \frac{\pi_k \exp(-\frac{1}{2\sigma^2}(x - \mu_k)^2)}{\sum_{l=1}^K \pi_l \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2)}
\end{aligned}$$

Si $a > b$ entonces $\log(a) > \log(b)$. Esto quiere decir que si se aplica el algoritmo a la ecuación (4.12) el k para el cual la ecuación se maximiza debe ser el mismo.

$$\begin{aligned}
\log(p_k(x)) &= \log \left(\frac{\pi_k \cdot \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2)}{\sum_{l=1}^K \pi_l \cdot \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2)} \right) \\
&= \log \left(\pi_k \cdot \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2) \right) - \log \left(\sum_{l=1}^K \pi_l \cdot \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2) \right) \\
&= \log(\pi_k) - \frac{1}{2\sigma^2}(x - \mu_j)^2 - \log \left(\sum_{l=1}^K \pi_l \cdot \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2) \right)
\end{aligned}$$

Se observa que el término $\log \left(\sum_{l=1}^K \pi_l \cdot \exp(-\frac{1}{2\sigma^2}(x - \mu_l)^2) \right)$ será igual para cualquier k del que se quiera saber $\log(p_k(x))$. Entonces se quiere encontrar un k para maximizar $\log(\pi_k) - \frac{1}{2\sigma^2}(x - \mu_j)^2$

$$\begin{aligned}
&\log(\pi_k) - \frac{1}{2\sigma^2}(x - \mu_j)^2 \\
&= \log(\pi_k) - \frac{(x^2 + \mu_k^2 - 2\mu_k x)}{2\sigma^2} \\
&= \log(\pi_k) - \frac{\mu_k^2}{2\sigma^2} + \frac{\mu_k x}{\sigma^2} - \frac{x^2}{2\sigma^2}
\end{aligned}$$

Aquí también se observa que el término $\frac{x^2}{2\sigma^2}$ será igual para cualquier k . Por lo tanto, se quiere maximizar:

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

Esta es la misma ecuación (4.13)

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

Por lo tanto, escoger un k que maximice $p_k(x)$ es equivalente a escoger un k que maximice $\delta_k(x)$

3. This problem relates to the QDA model, in which the observations within each class are drawn from a normal distribution with a class-specific mean vector and a class specific covariance matrix. We consider the simple case where $p = 1$; i.e. there is only one feature. Suppose that we have K classes, and that if an observation belongs to the k th class then X comes from a one-dimensional normal distribution, $X \sim N(\mu_k, \sigma_k^2)$. Recall that the density function for the one-dimensional normal distribution is given in (4.11). Prove that in this case, the Bayes' classifier is not linear. Argue that it is in fact quadratic. Hint: For this problem, you should follow the arguments laid out in Section 4.4.2, but without making the assumption that $\sigma_1^2 = \sigma_2^2 = \dots = \sigma_K^2$.

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

$$\delta_k(x) = -\frac{1}{2}(x - \mu_k)^T \frac{1}{\sum_k} (x - \mu_k) - \frac{1}{2} \log \left| \sum \right| + \log(\pi_k)$$

Como en este caso $p=1$ y X provienen de una distribución normal univariada, se puede simplificar la expresión de la siguiente forma:

$$\delta_k(x) = -\frac{1}{2}(x - \mu_k) \frac{1}{\sigma_k} (x - \mu_k) - \frac{1}{2} \log |\sigma_k| + \log(\pi_k)$$

$$= -\frac{1}{2\sigma_k}(x - \mu_k)^2 - \frac{1}{2} \log |\sigma_k| + \log(\pi_k)$$

Como se logra evidenciar, el clasificador de Bayes no es lineal al encontrarse un término cuadrático para X .

4. When the number of features p is large, there tends to be a deterioration in the performance of KNN and other local approaches that perform prediction using only observations that are near the test observation for which a prediction must be made. This phenomenon is known as the curse of dimensionality, and it ties into the fact that curse of dimensionality parametric approaches often perform poorly when p is large. We will now investigate this curse.

- (a) Suppose that we have a set of observations, each with measurements on $p = 1$ feature, X . We assume that X is uniformly (evenly) distributed on $[0, 1]$. Associated with each observation is a response value. Suppose that we wish to predict a test observation's response using only observations that are within 10% of the range of X closest to that test observation. For instance, in order to predict the response for a test observation with $X = 0.6$, we will use observations in the range $[0.55, 0.65]$. On average, what fraction of the available observations will we use to make the prediction?

Como X tiene distribución uniforme y toma valores en el intervalo $[0, 1]$ entonces si se tiene un x de prueba se tomarán valores en el intervalo $[x-0.05, x+0.05]$ para hacer la predicción de la variable de respuesta entonces la probabilidad de que un valor se encuentre en ese intervalo es:

$$P(X \in [x-0.05, x+0.05]) = \int_{x-0.05}^{x+0.05} \frac{1}{(x+0.05) - (x-0.05)} dx = 0.1$$

Por lo tanto la proporción que se usa para hacer la predicción es 10%.

- (b) Now suppose that we have a set of observations, each with measurements on $p = 2$ features, X_1 and X_2 . We assume that (X_1, X_2) are uniformly distributed on $[0, 1] \times [0, 1]$. We wish to predict a test observation's response using only observations that are within 10% of X_2 closest to that test observation. For instance, in order to predict the response for a test observation with $X_1 = 0.6$ and $X_2 = 0.35$, we will use observations in the range $[0.55, 0.65]$ for X_1 and in the range $[0.3, 0.4]$ for X_2 . On average, what fraction of the available observations will we use to make the prediction?

Como en el caso anterior, 10% de las observaciones satisfacen las condiciones que se piden para X_1 y 10% para X_2 pero la proporción de observaciones que satisfacen las condiciones para X_1 y X_2 simultáneamente se puede encontrar asumiendo que estas son independientes. En ese caso se pueden multiplicar las probabilidades de cada variable $0.1 * 0.1 = 0.01$, es decir, solo el 1% de las observaciones están en el intervalo $[x_i - 0.05, x_i + 0.05]$ con $i = 1, 2$ y esa es la fracción que se debe tomar para hacer las predicciones.

- (c) Now suppose that we have a set of observations on $p = 100$ features. Again the observations are uniformly distributed on each feature, and again each feature ranges in value from 0 to 1. We wish to predict a test observation's response using observations within the 10% of each feature's range that is closest to that test observation. What fraction of the available observations will we use to make the prediction? Siguiendo los pasos anteriores entonces sería $0.1^{100} = 1e - 100$,

$$\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = \log\left(\frac{p_1(x)}{p_2(x)}\right) = c_0 + c_1x, \quad (4.24)$$

where c_0 and c_1 are functions of μ_1, μ_2 , and σ^2 . From (4.4), we know that in logistic regression,

$$\log\left(\frac{p_1}{1-p_1}\right) = \beta_0 + \beta_1x. \quad (4.25)$$

at a drawback of KNN when p is large ns “near” any given test observation. nes incrementa, la probabilidad de que X_i con $i = 1, 2, \dots, p$ se acerca cada vez

$$P(X_1 \leq x_1, X_2 \leq x_2, \dots, X_p \leq x_p) = \lim_{p \rightarrow \infty} (0.1)^p = 0$$

En una muestra donde p es muy grande los KNN no estarán muy cercanos simplemente porque la probabilidad de que existan sus vecinos es 0, no existen, para todas las p variables.

- (e) Now suppose that we wish to make a prediction for a test observation by creating a p -dimensional hypercube centered around the test observation that contains, on average, 10% of the training observations. For $p = 1, 2$, and 100, what is the length of each side of the hypercube? Comment on your answer.

Note: A hypercube is a generalization of a cube to an arbitrary number of dimensions. When $p = 1$, a hypercube is simply a line segment, when $p = 2$ it is a square, and when $p = 100$ it is a 100-dimensional cube.

para $p = 1$ el lado del hypercubo mide $0.1^1 = 0.1$.

para $p = 2$ el lado del hypercubo mide $0.1^{\frac{1}{2}} = 0.316$.

para $p = 100$ el lado del hypercubo mide $0.1^{\frac{1}{100}} = 0.977$.

5. We now examine the differences between LDA and QDA.

- (a) If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?

Cuando el límite de Bayes es lineal QDA ajustará mejor a las observaciones de entrenamiento, mientras que LDA tendrá un ajuste más cercano a las observaciones de prueba, esto debido a los sobre ajustes que puede presentar QDA para este conjunto de observaciones.

- (b) If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

En este caso de forma similar al análisis anterior, QDA ajustará mejor al conjunto de observaciones de entrenamiento, pero además también tendrá un mejor ajuste para las observaciones de prueba.

- (c) In general, as the sample size n increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?

Entre mayor sea la razón entre el número de parámetros respecto al número de observaciones, mayor será la incidencia del sobreajuste para LDA, por lo tanto la relación de QDA con respecto a LDA mejora. De forma general para grandes conjuntos de observaciones el mejor ajuste lo presenta QDA.

- (d) True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.

Como se mencionó anteriormente, cuando el límite de decisión de Bayes es lineal, el mejor ajuste para los datos de prueba será al utilizar LDA y por lo tanto el enunciado es falso. Por otro lado también es importante considerar la cantidad de observaciones o el conjunto muestral, ya que QDA presenta sobre ajuste para cantidades pequeñas.

6. Suppose we collect data for a group of students in a statistics class with variables X_1 =hours studied, X_2 =undergrad GPA, and Y = receive an A. We fit a logistic regression and produce estimated coefficient,

$$\hat{\beta}_0 = -6, \hat{\beta}_1 = 0.05, \hat{\beta}_2 = 1$$

- (a) Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class.

La ecuación que determina la probabilidad estimada para este ejercicio esta dada por:

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2}} = \frac{e^{-6+0.05X_1+X_2}}{1 + e^{-6+0.05X_1+X_2}}$$

Al reemplazar por los valores dados tenemos que:

$$\hat{p}(X) = \frac{e^{-6+0.05(40)+(3.5)}}{1 + e^{-6+0.05(40)+(3.5)}} = 0.3775$$

Es decir que la probabilidad de que un estudiante que estudie 40 horas con un GPA de 3.5 obtenga una A es del 37.75%

- (b) How many hours would the student in part (a) need to study to have a 50% chance of getting an A in the class?

Para calcular el numero de horas que el estudiante debe estudiar para obtener una calificación de A se debe despejar X_1 de la siguiente ecuación.

$$\frac{e^{-6+0.05(X_1)+(3.5)}}{1 + e^{-6+0.05(X_1)+(3.5)}} = 0.5$$

Primero:

$$e^{-6+0.05(X_1)+(3.5)} = 0.5 (1 + e^{-6+0.05(X_1)+(3.5)})$$

Luego multiplicamos por 10 en ambos lados de la ecuación tal que:

$$10e^{-6+0.05X_1+3.5} = 5 (1 + e^{-6+0.05X_1+3.5})$$

Entonces nos queda que:

$$10e^{-6+0.05X_1+3.5} = (5 + 5e^{-6+0.05X_1+3.5})$$

Ahora agrupamos de tal forma que:

$$10e^{-6+0.05X_1+3.5} - 5e^{-6+0.05X_1+3.5} = 5$$

Operando tenemos:

$$5e^{-6+0.05X_1+3.5} = 5$$

Al pasar 5 a dividir nos queda que:

$$e^{-6+0.05X_1+3.5} = \frac{5}{5}$$

Al sacar logaritmo natural a ambos lados de la ecuaciones tenemos que:

$$\ln(e^{-6+0.05X_1+3.5}) = \ln(1)$$

Cuya resultante es una ecuación lineal de la que despejaremos X_1 .

$$-6 + 0.05X_1 + 3.5 = 0$$

Finalmente tenemos que la cantidad de horas que el estudiante debe estudiar es:

$$X_1 = \frac{2.5}{0.05} = 50$$

7. Suppose that we wish to predict whether a given stock will issue a dividend this year (“Yes” or “No”) based on X , last year’s percent profit. We examine a large number of companies and discover that the mean value of X for companies that issued a dividend was $\bar{X} = 10$, while the mean for those that didn’t was $\bar{X} = 0$. In addition, the variance of X for these two sets of companies was $\hat{\sigma}^2 = 36$. Finally, 80% of companies issued dividends. Assuming that X follows a normal distribution, predict the probability that a company will issue a dividend this year given that its percentage profit was $X = 4$ last year.

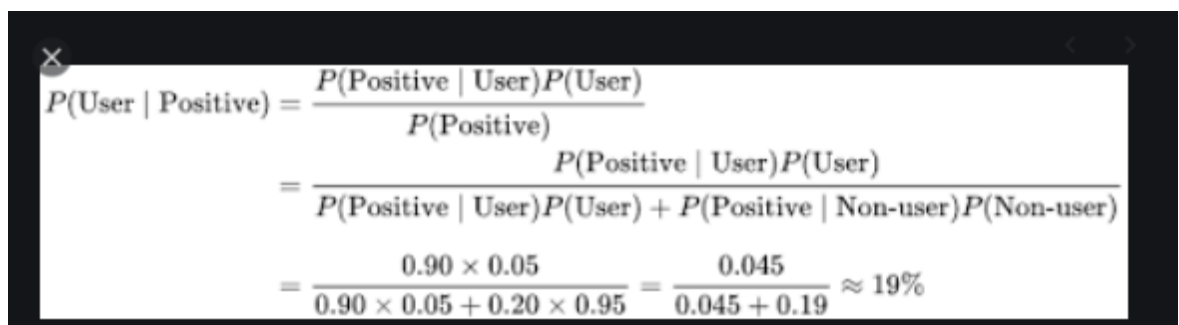
Hint: Recall that the density function for a normal random variable is

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2}$$

You will need to use Bayes’ theorem.

Para la varianza constante $\sigma_1^2 = \dots = \sigma_k^2$ tenemos que:

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{1}{2\sigma^2}(x - \mu_k)^2)}{\sum_{t=1}^k \pi_t \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{1}{2\sigma^2}(x - \mu_t)^2)}$$



The screenshot shows a calculator interface with the following steps for calculating $P(\text{User} | \text{Positive})$:

$$P(\text{User} | \text{Positive}) = \frac{P(\text{Positive} | \text{User})P(\text{User})}{P(\text{Positive})}$$

$$= \frac{P(\text{Positive} | \text{User})P(\text{User})}{P(\text{Positive} | \text{User})P(\text{User}) + P(\text{Positive} | \text{Non-user})P(\text{Non-user})}$$

$$= \frac{0.90 \times 0.05}{0.90 \times 0.05 + 0.20 \times 0.95} = \frac{0.045}{0.045 + 0.19} \approx 19\%$$

Reemplazando por los valores del ejercicio tenemos que:

$$p_{yes}(4) = \frac{0.8 \exp(-\frac{1}{2(36)}(4-10)^2)}{0.8 \exp(-\frac{1}{2(36)}(4-10)^2) + (1-0.8) \exp(-\frac{1}{2(36)}(4-0)^2)}$$

Calculando finalmente tenemos que:

$$p_{yes}(4) = 0.752$$

En otras palabras la probabilidad de que una empresa emita dividendos este año dado que su porcentaje de rentabilidad del año pasado fue $X=4$ es del 75.2%.

8. Suppose that we take a data set, divide it into equally-sized training and test sets, and then try out two different classification procedures. First we use logistic regression and get an error rate of 20% on the training data and 30% on the test data. Next we use 1-nearest neighbors (i.e. $K = 1$) and get an average error rate (averaged over both test and training data sets) of 18%. Based on these results, which method should we prefer to use for classification of new observations? Why?

A pesar que para la regresión logística la tasa de error en los datos de entrenamiento es considerablemente elevada y mayor que para los KNN con $K=1$ que puede estar cerca del 0% (Ya que el promedio de los errores de prueba y de entrenamiento es de 18%), la tasa de error para los datos de prueba en KNN será superior al 30% observado en la regresión. Por lo tanto la regresión logística es mejor por tener menor tasa de error menor para las observaciones de prueba.

9. This problem has to do with odds.

- (a) On average, what fraction of people with an odds of 0.37 of defaulting on their credit card payment will in fact default?

Tenemos que el odd ratio esta dado por:

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

Y queremos calcular la probabilidad de impago para:

$$\frac{p_{Impago}}{1-p_{Impago}} = 0.37$$

Para nuestro caso tenemos que:

$$\frac{p_{Impago}}{1-p_{Impago}} = 0.37$$

Debemos entonces dejar la probabilidad de impago:

$$p_{Impago} = 0.37(1 - p_{Impago})$$

Entonces:

$$p_{Impago} = 0.37 - 0.37p_{Impago}$$

Luego agrupando de un solo lado la probabilidad de impago:

$$p_{Impago} + 0.37p_{Impago} = 0.37$$

Es decir que:

$$p_{Impago}(1 + 0.37) = 0.37$$

Finalmente nos queda que la probabilidad de impago para un odds de 0.37 es:

$$p_{Impago} = \frac{0.37}{1 + 0.37} = 0.27$$

- (b) Suppose that an individual has a 16% chance of defaulting on her credit card payment. What are the odds that she will default?

Para este caso tenemos que:

$$\frac{p_{Impago}}{1 - p_{Impago}} = \frac{0.16}{1 - 0.16} = 0.1905$$

Es decir que el odds de impago es de 0.19 aproximadamente.

10. This question should be answered using the Weekly data set, which is part of the ISLR package. This data is similar in nature to the Smarket data from this chapter's lab, except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of.

```
library(ISLR)
data(Weekly)
attach(Weekly)
```

Acerca del dataframe:

Un marco de datos con 1089 observaciones sobre las siguientes 9 variables.

Año: El año en que se registró la observación

Lag1: Porcentaje de retorno de la semana anterior

Lag2: Porcentaje de rendimiento de las dos semanas anteriores

Lag3: Rendimiento porcentual de las 3 semanas anteriores

Lag4: Rendimiento porcentual de 4 semanas anteriores

Lag5: Rendimiento porcentual de las 5 semanas anteriores

Volumen: Volumen de acciones negociadas (número medio de acciones diarias negociadas en miles de millones)

Hoy: Rendimiento porcentual de esta semana

Dirección: Factor con niveles de bajada y subida que indica si el mercado ha tenido una rentabilidad positiva o negativa en una semana determinada

- (a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

```
summary(Weekly)
```

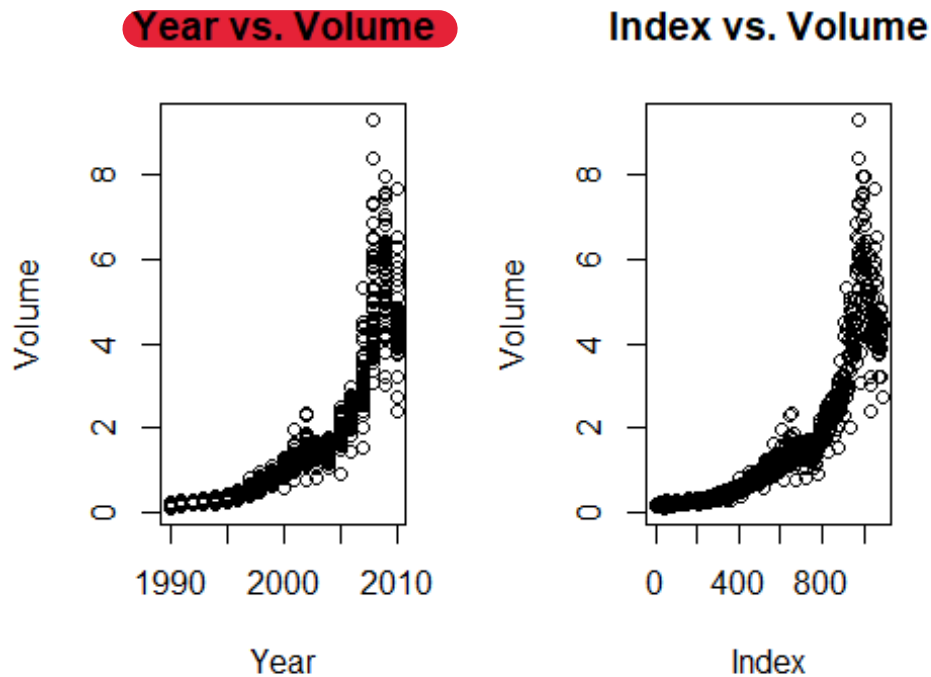
```
##      Year      Lag1      Lag2      Lag3
## Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
## 1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
## Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
## Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
## 3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
## Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
##      Lag4      Lag5      Volume      Today
## Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950
## 1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540
## Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410
## Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499
## 3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050
## Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260
## Direction
## Down:484
## Up  :605
```

```
##
##
##
##

round(cor(subset(Weekly, select = -Direction)),3)

##      Year   Lag1   Lag2   Lag3   Lag4   Lag5 Volume Today
## Year    1.000 -0.032 -0.033 -0.030 -0.031 -0.031  0.842 -0.032
## Lag1   -0.032  1.000 -0.075  0.059 -0.071 -0.008 -0.065 -0.075
## Lag2   -0.033 -0.075  1.000 -0.076  0.058 -0.072 -0.086  0.059
## Lag3   -0.030  0.059 -0.076  1.000 -0.075  0.061 -0.069 -0.071
## Lag4   -0.031 -0.071  0.058 -0.075  1.000 -0.076 -0.061 -0.008
## Lag5   -0.031 -0.008 -0.072  0.061 -0.076  1.000 -0.059  0.011
## Volume  0.842 -0.065 -0.086 -0.069 -0.061 -0.059  1.000 -0.033
## Today  -0.032 -0.075  0.059 -0.071 -0.008  0.011 -0.033  1.000

par(mfrow=c(1,2))
plot(Year,Volume,main="Year vs. Volume")
plot(Volume, main="Index vs. Volume")
```



La forma general del modelo de regresión logística es

$$y_i = E(y_i) + \varepsilon_i \quad (13.6)$$

donde las observaciones y_i son variables aleatorias independientes de Bernoulli, cuyos valores esperados son

$$\begin{aligned} E(y_i) &= \pi_i \\ &= \frac{\exp(\mathbf{x}_i' \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta})} \end{aligned} \quad (13.7)$$

- b) Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

```
glm.fit = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, family =
binomial )
summary(glm.fit)
```

Sea $\hat{\beta}$ el estimado final de los parámetros del modelo que se obtiene con el algoritmo anterior. Si son correctas las hipótesis del modelo, se puede demostrar que, en forma asintótica,

$$E(\hat{\beta}) = \beta \quad y \quad Var(\hat{\beta}) = (X'V^{-1}X)^{-1} \quad (13.10)$$

```
##
## Call:
## glm(formula = Direction
##      Volume, family = binomial)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106  0.0019 **
## Lag1        -0.04127    0.02641  -1.563  0.1181
## Lag2         0.05844    0.02686   2.175  0.0296 *
## Lag3        -0.01606    0.02666  -0.602  0.5469
## Lag4        -0.02779    0.02646  -1.050  0.2937
## Lag5        -0.01447    0.02638  -0.549  0.5833
## Volume       -0.02274    0.03690  -0.616  0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

El Lag2 presenta el único coeficiente significativo pues su p-value < 0.05.

- (c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

Este comando nos informa como R generara La variable dummy para Direction

```
contrasts(Direction)
```

```
##      Up
## Down  0
## Up    1
```

Se usa La función predict para determinar si el mercado subira o bajara, en este caso se usan Los valore de entrenamiento para el calculo.

```

glm.probs = predict(glm.fit ,type ="response")
glm.probs[1:10]

##           1           2           3           4           5           6           7
8
## 0.6086249 0.6010314 0.5875699 0.4816416 0.6169013 0.5684190 0.5786097 0.51
51972
##           9          10
## 0.5715200 0.5554287

# Convierte las probabilidades en etiquetas para saber si el mercado o bajara
.

len = length(glm.probs) # 1089
glm.pred = rep("Down", len ) # Crea un vector con 1089 "Down"
glm.pred[glm.probs > 0.5] = "Up" # Intercambia Los Down por up si prob > 50%
table(glm.pred,Direction) # Crea La matrix de confusión

##           Direction
## glm.pred Down  Up
##      Down   54  48
##      Up   430 557

```

La diagonal principal indica los elementos correctamente catalogados, es decir que nuestro modelo predice correctamente que 54 semanas el mercado estara a la baja y 557 al alta.

```

(54+557)

## [1] 611

mean(glm.pred == Direction)

## [1] 0.5610652

```

En otras palabras nuestro modelo clasifico correctamente 611 semanas es decir el 56.11% de las veces.

```

100-56.11

## [1] 43.89

```

Pero puesto que se entrenó el modelo con la totalidad de los datos quiere decir que tenemos una tasa error de entrenamiento del 43.89%

- d) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).

```

train = (Year<2009)
Weekly.2009 = Weekly[!train,]
Direction.2009 = Direction[!train]
dim(Weekly.2009)

```

```
## [1] 104 9

glm.fit.2009 = glm(Direction ~ Lag2, family = "binomial", subset=train)
summary(glm.fit.2009)

##
## Call:
## glm(formula = Direction ~ Lag2, family = "binomial", subset = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.536  -1.264   1.021   1.091   1.368
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.20326    0.06428   3.162  0.00157 **
## Lag2        0.05810    0.02870   2.024  0.04298 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1350.5  on 983  degrees of freedom
## AIC: 1354.5
##
## Number of Fisher Scoring iterations: 4

glm.probs.2009 = predict(glm.fit.2009, Weekly.2009, type="response")
glm.probs.2009[1:10]

##      986      987      988      989      990      991      992
993
## 0.5261291 0.6447364 0.4862159 0.4852001 0.5197667 0.5401255 0.6233482 0.48
09930
##      994      995
## 0.4512204 0.4848808

len.2009 = length(glm.probs.2009)
glm.pred.2009 = rep("Down", len.2009)
glm.pred.2009[glm.probs.2009 > 0.5] = "Up"
table(glm.pred.2009, Direction.2009)

##              Direction.2009
## glm.pred.2009 Down Up
##           Down   9  5
##           Up   34 56

mean(glm.pred.2009 == Direction.2009) #tasa de error

## [1] 0.625
```

Gracias a entrenar el modelo sobre los datos de entrenamiento los resultados de nuestras predicciones mejoran en un 6%, al clasificar correctamente 65 semanas es decir el 62.5% de las veces, con una tasa de error del $100 - 62.5 = 37.5\%$

```
56/(34+56)
```

```
## [1] 0.6222222
```

Por otro lado, la matriz de confusión muestra el 62.2% de las semanas en las que nuestro modelo predijo correctamente una subida en el mercado. Esto sugiere una posible estrategia de compra en los días en que el modelo predice un mercado creciente mercado, y evitar las operaciones en los días en que se predice un descenso.

(e) Repeat (d) using **LDA**.

```
library(MASS)

lda.fit = lda(Direction ~ Lag2, subset=train)
lda.fit

## Call:
## lda(Direction ~ Lag2, subset = train)
##
## Prior probabilities of groups:
##      Down      Up
## 0.4477157 0.5522843
##
## Group means:
##      Lag2
## Down -0.03568254
## Up    0.26036581
##
## Coefficients of linear discriminants:
##      LD1
## Lag2 0.4414162
```

La salida del modelo `lda` indica que $\hat{\pi}_1 = 0.448$ y $\hat{\pi}_2 = 0.552$, es decir que el 44.8% de las semanas corresponden a aquellas en las cuales el mercado estará a la baja.

class: Contiene Los predictores LDA sobre el movimiento del mercado

posterior: Es La matriz con Las columnas kth que contienen La probabilidad posterior

x: contiene Los discriminareos nombrados anteriormente.

```
lda.pred = predict(lda.fit,Weekly.2009)
names(lda.pred)
```

```
## [1] "class"      "posterior" "x"
```

```
lda.class=lda.pred$class
table(lda.class,Direction.2009)
```

```
##          Direction.2009
## lda.class Down Up
##      Down    9  5
##      Up     34 56

mean(lda.class == Direction.2009)

## [1] 0.625

## porcentaje de semanas en con subida del mercado acertadas

56/(56+5)

## [1] 0.9180328
```

Al aplicar un ajuste por discriminantes lineares tenemos que nuestro modelo predijo correctamente el 62.5% de las semanas, con una tasa de error del 37.5% y que este a su vez predijo que un correctamente un 91.8% de las veces que el mercado incrementaria.

Al aplicar un umbral del 50% para las predicciones posteriores de que el mercado caera tenemos que

```
sum(lda.pred$posterior[,1]>=0.5)

## [1] 14
```

Es decir que habrá 14 semanas en las que el mercado decaerá.

```
sum(lda.pred$posterior[,1]<0.5)

## [1] 90
```

Y un total de 90 semanas en las que no lo hará es decir es decir se predice que para el 86.54% de las semanas posteriores el mercado no bajará.

De manera gráfica podemos comparar que en las probabilidades posteriores inferiores al 50% que el mercado bajara se pronostica que el mercado subira.

```
lda.pred$posterior[1:20,1]

##      986      987      988      989      990      991      992
993
## 0.4736555 0.3558617 0.5132860 0.5142948 0.4799727 0.4597586 0.3771117 0.51
84724
##      994      995      996      997      998      999     1000
1001
## 0.5480397 0.5146118 0.5504246 0.3055404 0.4268160 0.3637275 0.4034316 0.42
56310
##     1002     1003     1004     1005
## 0.4277053 0.4548626 0.4308002 0.3674066

lda.class[1:20]
```



```
## [1] Up Up Down Down Up Up Up Down Down Down Down Up Up Up
Up
## [16] Up Up Up Up Up
## Levels: Down Up
```

(f) Repeat (d) using **QDA**.

```
qda.fit = qda(Direction ~ Lag2, subset = train)
qda.fit

## Call:
## qda(Direction ~ Lag2, subset = train)
##
## Prior probabilities of groups:
## Down Up
## 0.4477157 0.5522843
##
## Group means:
## Lag2
## Down -0.03568254
## Up 0.26036581
```

La salida del modelo **qda** indica que $\hat{\pi}_1 = 0.448$ y $\hat{\pi}_2 = 0.552$, es decir que el 55.2% de los días corresponden a aquellos en los cuales el mercado estará a la alza.

```
qda.class = predict(qda.fit, Weekly.2009)$class
table(qda.class, Direction.2009)

## Direction.2009
## qda.class Down Up
## Down 0 0
## Up 43 61

mean(qda.class == Direction.2009)

## [1] 0.5865385
```

El modelo **qda**, predice correctamente el 58.65% de los casos, es decir 4 puntos porcentuales menos que los modelos **glm** y **lda**, pues este fue incapaz de clasificar correctamente por lo menos una sola vez en la que el mercado estaría a la baja.

(g) Repeat (d) using **KNN with K = 1**.

```
library(class)

train.X = as.matrix(Lag2[train])
test.X = as.matrix(Lag2[!train])
train.Direction <- Direction[train]
set.seed(1)
pred.knn <- knn(train.X, test.X, train.Direction, k = 1)
table(pred.knn, Direction.2009)

## Direction.2009
## pred.knn Down Up
```

```
##      Down   21 30
##      Up    22 31

mean(pred.knn == Direction.2009)

## [1] 0.5
```

Usando $K = 1$ nuestro modelo predice correctamente los movimiento del mercado en un 50%.

```
set.seed(1)
pred.knn <- knn(train.X, test.X, train.Direction, k = 3)
table(pred.knn, Direction.2009)

##      Direction.2009
## pred.knn Down Up
##      Down    16 20
##      Up     27 41

mean(pred.knn == Direction.2009)

## [1] 0.5480769
```

Al disminuir la flexibilidad del modelo al aumentar a $K = 3$, vemos que este se ajusta un poco mejor prediciendo correctamente el 54.8% de las veces el movimiento del mercado.

(h) Which of these methods appears to provide the best results on this data?

En este caso particular los mejores modelos para predecir el comportamiento del mercado están dados por la regresión logística y por el análisis lineal de discriminantes pues estos otorgan clasifican correctamente el 62.5% de los movimientos del mercado. con una tasa de error de apenas el 37.5%, mientras que el modelo de discriminante cuadrático tiene una tasa de error del 41.35% y el KNN con $K = 3$ tiene una tasa de error del 45.2%.

(i) Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables, method, and associated confusion matrix that appears to provide the best results on the held out data. Note that you should also experiment with values for K in the KNN classifier.

```
# Logistic regression with Direction ~ Lag1 + Lag2
fit.glm3 <- glm(Direction ~ Lag1 + Lag2, data = Weekly, family = binomial, subset = train)
probs3 <- predict(fit.glm3, Weekly.2009, type = "response")
pred.glm3 <- rep("Down", length(probs3))
pred.glm3[probs3 > 0.5] = "Up"
table(pred.glm3, Direction.2009)

##      Direction.2009
## pred.glm3 Down Up
##      Down     7  8
##      Up     36 53

mean(pred.glm3 == Direction.2009)

## [1] 0.5769231
```

```

# LDA with Direction ~ Lag1 + Lag2
fit.lda2 <- lda(Direction ~ Lag1 + Lag2, data = Weekly, subset = train)
pred.lda2 <- predict(fit.lda2, Weekly.2009)
lda.class2=pred.lda2$class
table(lda.class2, Direction.2009)

##           Direction.2009
## lda.class2 Down Up
##           Down      7  8
##           Up       36 53

mean(pred.lda2$class == Direction.2009)

## [1] 0.5769231

# QDA with Direction ~ Lag1 + Lag2
fit.qda2 <- qda(Direction ~ Lag1 + Lag2, data = Weekly, subset = train)
pred.qda2 <- predict(fit.qda2, Weekly.2009)
table(pred.qda2$class, Direction.2009)

##           Direction.2009
##           Down Up
## Down      7 10
## Up       36 51

mean(pred.qda2$class == Direction.2009)

## [1] 0.5576923

# KNN k =5
pred.knn2 <- knn(train.X, test.X, train.Direction, k = 5)
table(pred.knn2, Direction.2009)

##           Direction.2009
## pred.knn2 Down Up
##           Down  15 20
##           Up   28 41

mean(pred.knn2 == Direction.2009)

## [1] 0.5384615
pred.knn3 <- knn(train.X, test.X, train.Direction, k = 10) # KNN k = 10
table(pred.knn3, Direction.2009)

##           Direction.2009
## pred.knn3 Down Up
##           Down  19 19
##           Up   24 42

mean(pred.knn3 == Direction.2009)

## [1] 0.5865385

```

11. this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the Auto data set.

```
data(Auto)
attach(Auto)
```

- (a) Create a binary variable, mpg01, that contains a 1 if mpg contains a value above its median, and a 0 if mpg contains a value below its median. You can compute the median using the median() function. Note you may find it helpful to use the data.frame() function to create a single data set containing both mpg01 and the other Auto variables.

```
median(mpg)
```

```
## [1] 22.75
```

```
mpg01 <- ifelse(Auto$mpg > median(Auto$mpg), 1, 0)
```

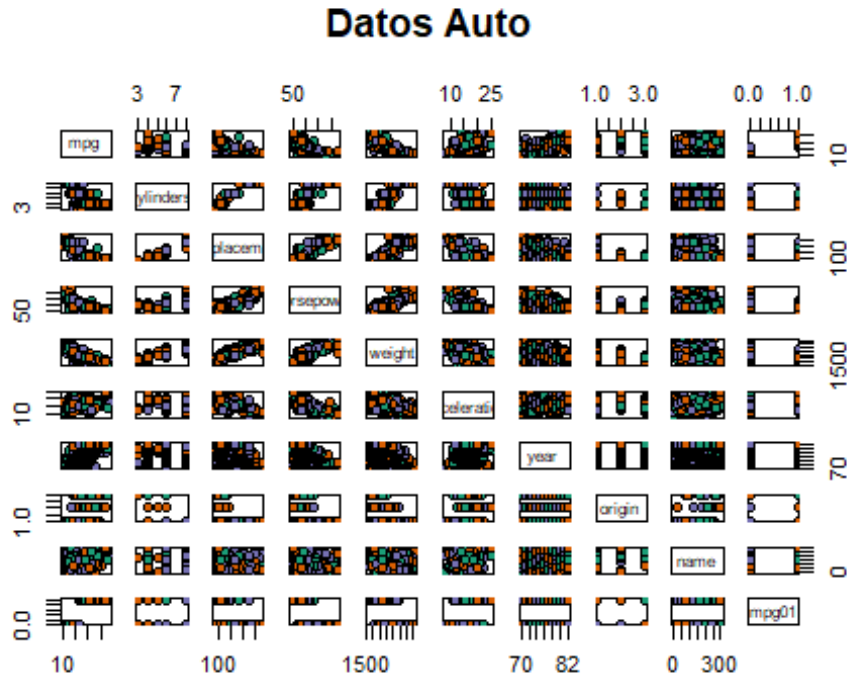
```
mod <- data.frame(Auto,mpg01)
```

```
summary(mod)
```

```
##      mpg      cylinders      displacement      horsepower
weight
## Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.
:1613
## 1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st
Qu.:2225
## Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median
:2804
## Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean
:2978
## 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd
Qu.:3615
## Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.
:5140
##
##      acceleration      year      origin      name
## Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador      : 5
## 1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto       : 5
## Median :15.50   Median :76.00   Median :1.000   toyota corolla   : 5
## Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin      : 4
## 3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet       : 4
## Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette: 4
##                                     (Other)           :365
##
##      mpg01
## Min.   :0.0
## 1st Qu.:0.0
## Median :0.5
## Mean   :0.5
## 3rd Qu.:1.0
## Max.   :1.0
##
```

- (b) Explore the data graphically in order to investigate the association between mpg01 and the other features. Which of the other features seem most likely to be useful in predicting mpg01? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

```
pairs(mod[1:10],
      main = "Datos Auto",
      pch = 21,
      bg = c("#1b9e77", "#d95f02", "#7570b3"))
```

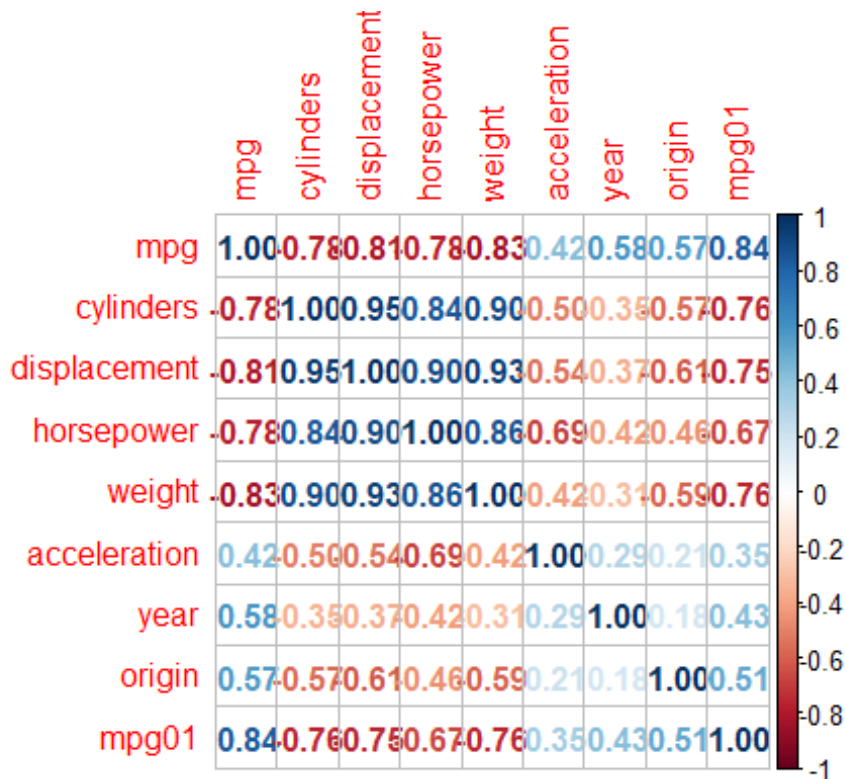


```
mod$name = NULL
mod.cor <- cor(mod, method = "pearson")
round(mod.cor, digits = 2)
```

```
##          mpg cylinders displacement horsepower weight acceleration
year
## mpg          1.00      -0.78         -0.81         -0.78  -0.83          0.42
0.58
## cylinders    -0.78        1.00          0.95          0.84   0.90         -0.50 -
0.35
## displacement -0.81        0.95          1.00          0.90   0.93         -0.54 -
0.37
## horsepower   -0.78        0.84          0.90          1.00   0.86         -0.69 -
0.42
## weight       -0.83        0.90          0.93          0.86   1.00         -0.42 -
0.31
## acceleration 0.42        -0.50         -0.54         -0.69  -0.42          1.00
0.29
```

```
## year      0.58    -0.35    -0.37    -0.42   -0.31      0.29
1.00
## origin    0.57    -0.57    -0.61    -0.46   -0.59      0.21
0.18
## mpg01     0.84    -0.76    -0.75    -0.67   -0.76      0.35
0.43
##          origin mpg01
## mpg          0.57  0.84
## cylinders    -0.57 -0.76
## displacement -0.61 -0.75
## horsepower   -0.46 -0.67
## weight       -0.59 -0.76
## acceleration  0.21  0.35
## year         0.18  0.43
## origin       1.00  0.51
## mpg01        0.51  1.00

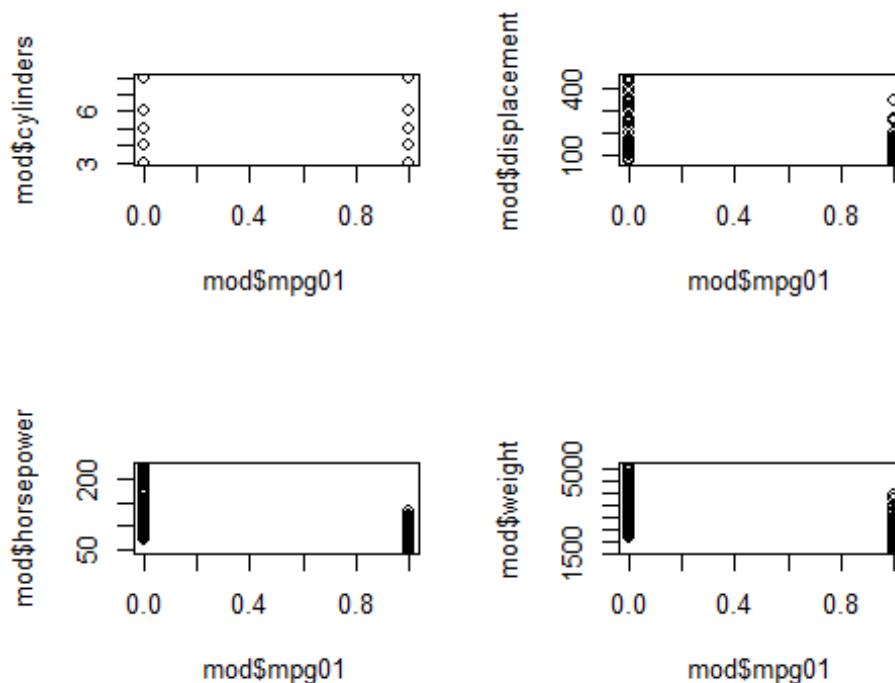
corrplot(mod.cor, method = "number")
```



Como se puede observar, las mayores correlaciones de la covariable mpg01 se presentan con respecto a cylinders, displasment, weight y horsepower, con valores negativos de -0.76, -0.76, -0.75 y -0.67 respectivamente. A continuación, se pueden observar graficamente la relación entre mpg01 y cada una de ellas.

```
par(mfrow=c(2,2))
plot(mod$mpg01, mod$cylinders)
plot(mod$mpg01, mod$displacement)
```

```
plot(mod$mpg01, mod$horsepower)
plot(mod$mpg01, mod$weight)
```



(c) Split the data into a training set and a test set.

```
set.seed(1)
trainid <- sample(1:nrow(mod), nrow(mod)*0.7 , replace=F)
train <- mod[trainid,]
test <- mod[-trainid,]
```

(d) Perform LDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

Las variables que presentan una mayor asociación con respecto a mpg01 son cylinders, displacement, weight y horsepower como se evidenció anteriormente en el análisis de correlaciones.

```
mod.lda <- lda(mpg01~cylinders+weight+displacement+horsepower, data=train)
mod.lda

## Call:
## lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = train)
##
## Prior probabilities of groups:
##      0      1
## 0.4927007 0.5072993
##
## Group means:
##  cylinders  weight displacement horsepower
```

```
## 0  6.777778 3611.052      271.9333  129.13333
## 1  4.187050 2342.165      116.8129   79.27338
##
## Coefficients of linear discriminants:
##                      LD1
## cylinders      -0.3962357999
## weight         -0.0008321338
## displacement  -0.0047630097
## horsepower     0.0061919395

pred.lda <- predict(mod.lda, test)
table(pred.lda$class, test$mpg01)

##
##      0  1
## 0 50  3
## 1 11 54

mean(pred.lda$class != test$mpg01)

## [1] 0.1186441
```

El error de prueba es 11.86% aproximadamente.

- (e) Perform QDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```
mod.qda <- qda(mpg01~cylinders + weight + displacement + horsepower,
data=train)
pred.qda <- predict(mod.qda, test)
table(pred.qda$class, test$mpg01)

##
##      0  1
## 0 52  5
## 1  9 52

mean(pred.qda$class != test$mpg01)

## [1] 0.1186441
```

El error de prueba en este caso es de 11.86% aproximadamente igual que en el caso del análisis lineal discriminante (LDA), a pesar que la matriz de confusión es distinta.

- (f) Perform logistic regression on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```
mod.logit <- glm(mpg01~cylinders + weight + displacement + horsepower,
data=train, family=binomial)
summary(mod.logit)

##
## Call:
## glm(formula = mpg01 ~ cylinders + weight + displacement + horsepower,
```



```

##      family = binomial, data = train)
##
## Deviance Residuals:
##      Min        1Q      Median        3Q        Max
## -2.4794  -0.1963   0.1056   0.3508   3.3756
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  11.725290   2.147421   5.460 4.76e-08 ***
## cylinders     0.056770   0.419131   0.135  0.8923
## weight       -0.001931   0.000817  -2.364  0.0181 *
## displacement -0.014718   0.009904  -1.486  0.1373
## horsepower   -0.041518   0.017821  -2.330  0.0198 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 379.79  on 273  degrees of freedom
## Residual deviance: 144.49  on 269  degrees of freedom
## AIC: 154.49
##
## Number of Fisher Scoring iterations: 7

prob.mod.logit <- predict(mod.logit, test, type="response")
pred.logit <- rep(0, length(prob.mod.logit))
pred.logit[pred.mod.logit > 0.5] <- 1
table(pred.logit, test$mpg01)

##
## pred.logit  0  1
##           0 53  3
##           1  8 54

mean(pred.logit != test$mpg01)

## [1] 0.09322034

```

El Error de prueba para la regresión logit fue de 9.32% aproximadamente.

- (g) Perform KNN on the training data, with several values of K, in order to predict mpg01. Use only the variables that seemed most associated with mpg01 in (b). What test errors do you obtain? Which value of K seems to perform the best on this data set?

```

train.X <- cbind(train$cylinders, train$weight, train$displacement,
train$horsepower)
test.X <- cbind(test$cylinders, test$weight, test$displacement,
test$horsepower)

knn.pred_1 <- knn(train.X, test.X, train$mpg01, k=1)
table(knn.pred_1, test$mpg01)

```

```
##
## knn.pred_1  0  1
##           0 51  6
##           1 10 51

mean(knn.pred_1 != test$mpg01)

## [1] 0.1355932
```

La tasa de error de prueba para K=1 es de 13.56%. Ahora se busca obtener este error para K=10, K=20, K=30, K=50 y K=100

```
knn.pred_10 <- knn(train.X, test.X, train$mpg01, k=10)
table(knn.pred_10, test$mpg01)

##
## knn.pred_10  0  1
##           0 49  5
##           1 12 52

mean(knn.pred_10 != test$mpg01)

## [1] 0.1440678

knn.pred_20 <- knn(train.X, test.X, train$mpg01, k=20)
table(knn.pred_20, test$mpg01)

##
## knn.pred_20  0  1
##           0 51  6
##           1 10 51

mean(knn.pred_20 != test$mpg01)

## [1] 0.1355932

knn.pred_30 <- knn(train.X, test.X, train$mpg01, k=30)
table(knn.pred_30, test$mpg01)

##
## knn.pred_30  0  1
##           0 52  7
##           1  9 50

mean(knn.pred_30 != test$mpg01)

## [1] 0.1355932

knn.pred_50 <- knn(train.X, test.X, train$mpg01, k=50)
table(knn.pred_50, test$mpg01)

##
## knn.pred_50  0  1
```

```
##           0 50  7
##           1 11 50

mean(knn.pred_50 != test$mpg01)

## [1] 0.1525424

knn.pred_100 <- knn(train.X, test.X, train$mpg01, k=100)
table(knn.pred_100, test$mpg01)

##
## knn.pred_100  0  1
##              0 50  7
##              1 11 50

mean(knn.pred_100 != test$mpg01)

## [1] 0.1525424
```

El menor error de las opciones analizadas resultan ser para $K=20$ con un error de 12.71%

12. This problem involves writing functions.

- (a) Write a function, `Power()`, that prints out the result of raising 2 to the 3rd power. In other words, your function should compute 2^3 and print out the results. Hint: Recall that x^a raises x to the power a . Use the `print()` function to output the result.

```
Power <- function() {print(2^3)}
Power()

## [1] 8
```

- (b) Create a new function, `Power2()`, that allows you to pass any two numbers, x and a , and prints out the value of x^a . You can do this by beginning your function with the line

```
Power2 = function (x,a){
```

You should be able to call your function by entering, for instance,

```
Power2 (3 ,8)
```

on the command line. This should output the value of 3^8 , namely, 6561

```
Power2 <- function(x, a) {
  x^a
}

Power2(3, 8)

## [1] 6561
```

- (c) Using the `Power2()` function that you just wrote, compute 10^3 , 81^7 , and 131^3 .

```
Power2(10, 3)
```

```
## [1] 1000

Power2(8, 17)

## [1] 2.2518e+15

Power2(131, 3)

## [1] 2248091
```

- (d) Now create a new function, `Power3()`, that actually returns the result x^a as an R object, rather than simply printing it to the screen. That is, if you store the value x^a in an object called `result` within your function, then you can simply `return()` this `return()` result, using the following line:

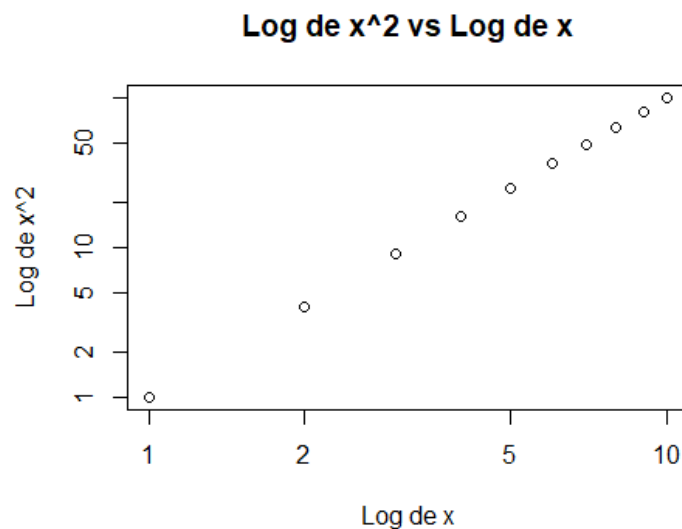
`return (result)`

The line above should be the last line in your function, before the `}` symbol.

```
Power3 <- function(x , a) {
  rta <- x^a
  return(rta)
}
```

- (e) Now using the `Power3()` function, create a plot of $f(x) = x^2$. The x-axis should display a range of integers from 1 to 10, and the y-axis should display x^2 . Label the axes appropriately, and use an appropriate title for the figure. Consider displaying either the x-axis, the y-axis, or both on the log-scale. You can do this by using `log="x"`, `log="y"`, or `log="xy"` as arguments to the `plot()` function.

```
x <- 1:10
plot(x, Power3(x, 2), log = "xy", xlab = "Log de x", ylab = "Log de x^2",
main = "Log de x^2 vs Log de x")
```



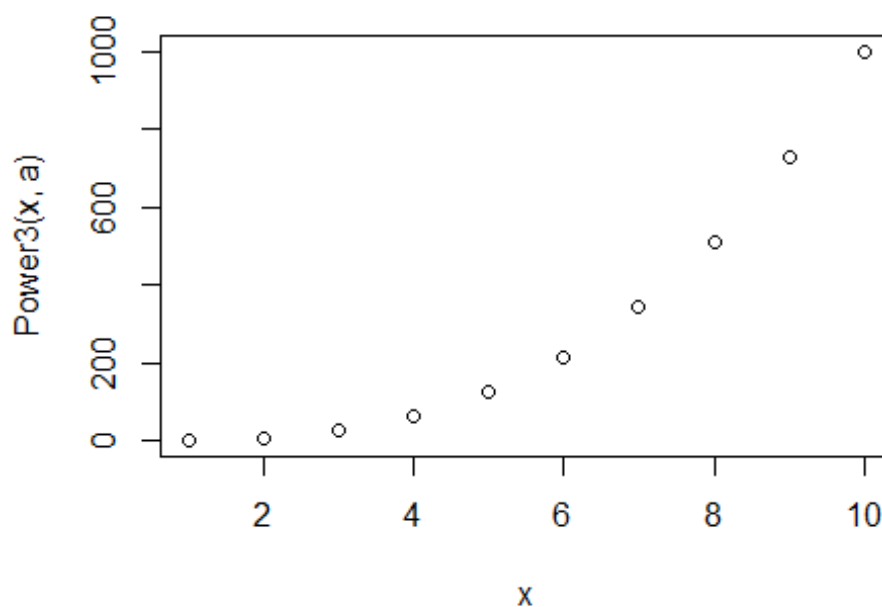
- (f) Create a function, `PlotPower()`, that allows you to create a plot of x against x^a for a fixed a and for a range of values of x . For instance, if you call

`PlotPower(1:10,3)`

then a plot should be created with an x-axis taking on values 1, 2, ..., 10, and a y-axis taking on values $1^3, 2^3, \dots, 10^3$.

```
PlotPower <- function(x, a) {  
  plot(x, Power3(x, a))  
}
```

```
PlotPower(1:10, 3)
```



13. Using the **Boston** data set, fit classification models in order to predict whether a given suburb has a crime rate above or below the median. Explore logistic regression, LDA, and KNN models using various subsets of the predictors. Describe your findings.

```
library("ISLR")
library("MASS")
data(Boston)
```

Acerca del dataframe:

crim: tasa de criminalidad per cápita por ciudad.

zn: proporción de suelo residencial con lotes de más de 25.000 pies cuadrados.

indus: proporción de acres comerciales no minoristas por ciudad.

chas Variable ficticia del río Charles (= 1 si el tramo limita con el río; 0 en caso contrario).

nox: concentración de óxidos de nitrógeno (partes por 10 millones).

rm: número medio de habitaciones por vivienda.

age: proporción de unidades ocupadas por sus propietarios construidas antes de 1940.

dis: media ponderada de las distancias a cinco centros de empleo de Boston.

rad: índice de accesibilidad a las carreteras radiales.

tax: tasa de impuesto sobre la propiedad de valor total por 10.000 dólares.

ptratio: proporción de alumnos por profesor por ciudad.

black: $1000(B_k - 0,63)^2$ donde B_k es la proporción de negros por ciudad.

lstat: Porcentaje población estratos bajos (porcentaje).

medv: valor medio de las viviendas ocupadas por sus propietarios, en 1000 dólares.

Puesto que se va analizar sobre la variable crimen se generará un summary de esta.

```
summary(Boston$crim)
```

```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## 0.00632  0.08204  0.25651  3.61352  3.67708 88.97620
```

Se separan los datos mediante la mediana, aquellos valores superiores a la mediana de la criminalidad son clasificados como 1 los demás como 0.

```
c.bool <- ifelse(Boston$crim > median(Boston$crim), 1, 0)
datos <- data.frame(Boston, c.bool)
round(cor(Boston),3)
```

```
##      crim      zn  indus   chas   nox    rm   age   dis   rad
tax
## crim      1.000 -0.200  0.407 -0.056  0.421 -0.219  0.353 -0.380  0.626
0.583
## zn       -0.200  1.000 -0.534 -0.043 -0.517  0.312 -0.570  0.664 -0.312 -
```

```

0.315
## indus    0.407 -0.534  1.000  0.063  0.764 -0.392  0.645 -0.708  0.595
0.721
## chas    -0.056 -0.043  0.063  1.000  0.091  0.091  0.087 -0.099 -0.007 -
0.036
## nox      0.421 -0.517  0.764  0.091  1.000 -0.302  0.731 -0.769  0.611
0.668
## rm      -0.219  0.312 -0.392  0.091 -0.302  1.000 -0.240  0.205 -0.210 -
0.292
## age      0.353 -0.570  0.645  0.087  0.731 -0.240  1.000 -0.748  0.456
0.506
## dis     -0.380  0.664 -0.708 -0.099 -0.769  0.205 -0.748  1.000 -0.495 -
0.534
## rad      0.626 -0.312  0.595 -0.007  0.611 -0.210  0.456 -0.495  1.000
0.910
## tax      0.583 -0.315  0.721 -0.036  0.668 -0.292  0.506 -0.534  0.910
1.000
## ptratio  0.290 -0.392  0.383 -0.122  0.189 -0.356  0.262 -0.232  0.465
0.461
## black   -0.385  0.176 -0.357  0.049 -0.380  0.128 -0.274  0.292 -0.444 -
0.442
## lstat    0.456 -0.413  0.604 -0.054  0.591 -0.614  0.602 -0.497  0.489
0.544
## medv    -0.388  0.360 -0.484  0.175 -0.427  0.695 -0.377  0.250 -0.382 -
0.469
##          ptratio  black  lstat  medv
## crim      0.290 -0.385  0.456 -0.388
## zn        -0.392  0.176 -0.413  0.360
## indus      0.383 -0.357  0.604 -0.484
## chas      -0.122  0.049 -0.054  0.175
## nox        0.189 -0.380  0.591 -0.427
## rm        -0.356  0.128 -0.614  0.695
## age        0.262 -0.274  0.602 -0.377
## dis       -0.232  0.292 -0.497  0.250
## rad        0.465 -0.444  0.489 -0.382
## tax        0.461 -0.442  0.544 -0.469
## ptratio    1.000 -0.177  0.374 -0.508
## black     -0.177  1.000 -0.366  0.333
## lstat      0.374 -0.366  1.000 -0.738
## medv     -0.508  0.333 -0.738  1.000

```

Para trabajar sobre los datos de la criminalidad se decide trabajar sobre aquellos que presentan las 8 correlaciones más altas con la variable criminalidad.

```

sort(cor(Boston)[1,],decreasing = T)

##          crim          rad          tax          lstat          nox          indus
## 1.000000000  0.62550515  0.58276431  0.45562148  0.42097171  0.40658341
##          age          ptratio          chas          zn          rm          dis
## 0.35273425  0.28994558 -0.05589158 -0.20046922 -0.21924670 -0.37967009

```

```
##          black          medv
## -0.38506394 -0.38830461
```

Separados por los siguientes grupos de variables:

A. rad + tax + lstat + nox B. indus + age + ptratio + chas

Para crear los datos de entrenamiento y de prueba se creará el subset train que contendrá el 80% de los datos del dataset, los demás serán tomados para el subset test.

```
round(nrow(datos)*.8,0)

## [1] 405

# Train y test data
train <- subset(datos[1:405,])
test  <- subset(datos[406:nrow(datos),])
```

Se realiza el ajuste del modelo con el grupo de Variables A:

```
glm.fit <- glm(c.bool~ rad + tax +lstat +nox, data=train, family=binomial)
summary(glm.fit)

##
## Call:
## glm(formula = c.bool ~ rad + tax + lstat + nox, family = binomial,
##      data = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.3235  -0.3416  -0.1010   0.1379   2.6470
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -22.500792   2.652463  -8.483  < 2e-16 ***
## rad          0.602009   0.132987   4.527 5.99e-06 ***
## tax         -0.003499   0.003100  -1.129   0.259
## lstat        0.010863   0.033451   0.325   0.745
## nox         38.493384   4.975554   7.737 1.02e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 548.94  on 404  degrees of freedom
## Residual deviance: 218.95  on 400  degrees of freedom
## AIC: 228.95
##
## Number of Fisher Scoring iterations: 8
```

Se descartan las variables lstat y tax, pues su p-value > 0.05, por lo que se vuelve a ajustar el modelo solo con rad y nox.


```

glm.fit <- glm(c.bool~rad+nox, data=train, family=binomial)
summary(glm.fit)

##
## Call:
## glm(formula = c.bool ~ rad + nox, family = binomial, data = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.4205  -0.3655  -0.1221   0.1674   2.6775
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -22.2599     2.5945  -8.580  < 2e-16 ***
## rad          0.5393     0.1153   4.679 2.88e-06 ***
## nox          36.8225     4.4321   8.308  < 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: 548.94  on 404  degrees of freedom
## Residual deviance: 220.30  on 402  degrees of freedom
## AIC: 226.3
##
## Number of Fisher Scoring iterations: 8

logit.prob <- predict(glm.fit, test, type="response")
logit.pred <- ifelse(logit.prob > 0.5, 1, 0)
table(logit.pred, test$c.bool)

##
## logit.pred  0  1
##           0  5  0
##           1 10 86

cat("\n")

pc <- round(mean(logit.pred == test$c.bool)*100,2)
te <- round((1-mean(logit.pred == test$c.bool))*100,2)
cbind(pc,te)

##      pc  te
## [1,] 90.1 9.9

```

El modelo que ocupa las variables rad + nox clasifica correctamente 91 suburbios y si este suburbio está por encima de la tasa de criminalidad de la mediana el 90.1% de las veces, con una tasa de error del 9.9%.

Se realiza el ajuste del modelo con el grupo de Varibales B:

```

glm.fit <- glm(c.bool~indus+age+ptratio+chas, data=train, family=binomial)
summary(glm.fit)

##
## Call:
## glm(formula = c.bool ~ indus + age + ptratio + chas, family = binomial,
##      data = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.3797  -0.6048  -0.2900   0.5296   2.7136
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.833696   1.192362  -4.054 5.04e-05 ***
## indus        0.116972   0.026965   4.338 1.44e-05 ***
## age          0.046411   0.006782   6.843 7.76e-12 ***
## ptratio      0.005350   0.063050   0.085  0.932
## chas         0.320982   0.467470   0.687  0.492
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 548.94  on 404  degrees of freedom
## Residual deviance: 342.02  on 400  degrees of freedom
## AIC: 352.02
##
## Number of Fisher Scoring iterations: 5

```

Se descartan las variables ptratio y chas, pues su p-value > 0.05, por lo que se vuelve a ajustar el modelo solo con indus y age.

```

glm.fit <- glm(c.bool~indus+age, data=train, family=binomial)
summary(glm.fit)

##
## Call:
## glm(formula = c.bool ~ indus + age, family = binomial, data = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.4077  -0.6090  -0.2922   0.5253   2.7083
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.733088   0.468450 -10.104 < 2e-16 ***
## indus        0.119702   0.026300   4.551 5.33e-06 ***
## age          0.046439   0.006759   6.871 6.37e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 548.94 on 404 degrees of freedom
## Residual deviance: 342.49 on 402 degrees of freedom
## AIC: 348.49
##
## Number of Fisher Scoring iterations: 5

logit.prob <- predict(glm.fit, test, type="response")
logit.pred <- ifelse(logit.prob > 0.5, 1, 0)
table(logit.pred, test$c.bool)

##
## logit.pred  0  1
##           0  5  8
##           1 10 78

cat("\n")

pc <- round(mean(logit.pred == test$c.bool)*100,2)
te <- round((1-mean(logit.pred == test$c.bool))*100,2)
cbind(pc,te)

##           pc      te
## [1,] 82.18 17.82
```

El modelo que ocupa las variables indus + age clasifica correctamente 82 suburbios y si este suburbio esta por encima de la tasa de criminalidad de la mediana el 82.12% de las veces, con una tasa de error del 17.82%.

LDA models

```
lda.fit <- lda(c.bool~rad+nox, data=train)
lda.fit

## Call:
## lda(c.bool ~ rad + nox, data = train)
##
## Prior probabilities of groups:
##           0           1
## 0.5876543 0.4123457
##
## Group means:
##           rad           nox
## 0  4.189076 0.4635324
## 1 10.598802 0.6298922
##
## Coefficients of linear discriminants:
##           LD1
## rad  0.0391204
## nox 11.2491973
```

```

cat("\n")

lda.fit.pred <- predict(lda.fit, test)$class
table(lda.fit.pred, test$c.bool)

##
## lda.fit.pred  0  1
##              0  5  0
##              1 10 86

cat("\n")

pc <- round(mean(lda.fit.pred == test$c.bool)*100,2)
te <- round((1-mean(lda.fit.pred == test$c.bool))*100,2)
cbind(pc,te)

##      pc  te
## [1,] 90.1 9.9

```

El modelo que ocupa las variables rad + nox clasifica correctamente 91 suburbios y si este suburbio esta por encima de la tasa de criminalidad de la mediana el 90.1% de las veces, con una tasa de error del 9.9%. Es decir que el 58.76% de los suburbios corresponden a aquellos en los cuales la criminalidad estará por debajo de la mediana.

```

# LDA models

lda.fit <- lda(c.bool~indus+age, data=train)
lda.fit

## Call:
## lda(c.bool ~ indus + age, data = train)
##
## Prior probabilities of groups:
##      0      1
## 0.5876543 0.4123457
##
## Group means:
##      indus      age
## 0  6.406639 49.58824
## 1 13.965629 85.70120
##
## Coefficients of linear discriminants:
##      LD1
## indus 0.09518815
## age   0.02931271

cat("\n")

lda.fit.pred <- predict(lda.fit, test)$class
table(lda.fit.pred, test$c.bool)

```

```
##
## lda.fit.pred  0  1
##              0  4  6
##              1 11 80

cat("\n")

pc <- round(mean(lda.fit.pred == test$c.bool)*100,2)
te <- round((1-mean(lda.fit.pred == test$c.bool))*100,2)
cbind(pc,te)

##          pc      te
## [1,] 83.17 16.83
```

El modelo que ocupa las variables indus + age clasifica correctamente 84 suburbios y si este suburbio esta por encima de la tasa de criminalidad de la mediana el 83.17% de las veces, con una tasa de error del 16.83%. Es decir que el 58.76% de los suburbios corresponden a aquellos en los cuales la criminalidad estará por debajo de la mediana.

```
# QDA models
qda.fit <- qda(c.bool~rad+nox, data=train)
qda.fit

## Call:
## qda(c.bool ~ rad + nox, data = train)
##
## Prior probabilities of groups:
##          0          1
## 0.5876543 0.4123457
##
## Group means:
##          rad          nox
## 0  4.189076 0.4635324
## 1 10.598802 0.6298922

cat("\n")

qda.fit.pred <- predict(qda.fit, test)$class
table(qda.fit.pred,test$c.bool)

##
## qda.fit.pred  0  1
##              0  0  0
##              1 15 86

cat("\n")

pc <- round(mean(qda.fit.pred == test$c.bool)*100,2)
te <- round((1-mean(qda.fit.pred == test$c.bool))*100,2)
cbind(pc,te)

##          pc      te
## [1,] 85.15 14.85
```

El modelo que ocupa las variables rad + nox clasifica correctamente 86 suburbios y si este suburbio esta por encima de la tasa de criminalidad de la mediana el 85.15% de las veces, con una tasa de error del 14.85%.Es decir que el 58.76% de los suburbios corresponden a aquellos en los cuales la criminalidad estará por debajo de la mediana.

```
# QDA models
qda.fit <- qda(c.bool~indus+age, data=train)
qda.fit

## Call:
## qda(c.bool ~ indus + age, data = train)
##
## Prior probabilities of groups:
##      0      1
## 0.5876543 0.4123457
##
## Group means:
##      indus      age
## 0  6.406639 49.58824
## 1 13.965629 85.70120

cat("\n")

qda.fit.pred <- predict(qda.fit, test)$class
table(qda.fit.pred,test$c.bool)

##
## qda.fit.pred  0  1
##              0  4  8
##              1 11 78

cat("\n")

pc <- round(mean(qda.fit.pred == test$c.bool)*100,2)
te <- round((1-mean(qda.fit.pred == test$c.bool))*100,2)
cbind(pc,te)

##      pc      te
## [1,] 81.19 18.81
```

El modelo que ocupa las variables indus + age clasifica correctamente 82 suburbios y si este suburbio esta por encima de la tasa de criminalidad de la mediana el 81.19% de las veces, con una tasa de error del 18.81%.Es decir que el 58.76% de los suburbios corresponden a aquellos en los cuales la criminalidad estará por debajo de la mediana.

A continuación se realizara el análisis por KNN de del grupo de variables rad + nox:

```
library(class)
train.X1 <- cbind(train$rad, train$nox)
test.X1 <- cbind(test$rad, test$nox)
set.seed(1)
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=1)
print("K=1")
```

```
## [1] "K=1"
mean(knn1.pred == test$c.bool)*100
## [1] 92.07921
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=5)
print("K=5")
## [1] "K=5"
mean(knn1.pred == test$c.bool)*100
## [1] 92.07921
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=10)
print("K=10")
## [1] "K=10"
mean(knn1.pred == test$c.bool)*100
## [1] 92.07921
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=20)
print("K=20")
## [1] "K=20"
mean(knn1.pred == test$c.bool)*100
## [1] 92.07921
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=50)
print("K=50")
## [1] "K=50"
mean(knn1.pred == test$c.bool)*100
## [1] 92.07921
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=100)
print("K=100")
## [1] "K=100"
mean(knn1.pred == test$c.bool)*100
## [1] 97.0297
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=200)
print("K=200")
## [1] "K=200"
```

```
mean(knn1.pred == test$c.bool)*100
```

```
## [1] 68.31683
```

Se aprecia que para las variables rad + nox el modelo clasifica correctamente los suburbios cuando su criminalidad esta por encima de la media el 92.08% de las veces desde $k = 1 \dots k = 50$, donde sufre un cambio de 5 puntos porcentuales con $k = 100$ hasta lograr el 97.03% de aciertos, y decreciendo al el 68.32% cuando $k = 200$

A continuación, se realizara el análisis por KNN de del grupo de variables indus + age:

```
train.X1 <- cbind(train$indus, train$age)
```

```
test.X1 <- cbind(test$indus, test$age)
```

```
set.seed(1)
```

```
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=1)
```

```
print("K=1")
```

```
## [1] "K=1"
```

```
mean(knn1.pred == test$c.bool)*100
```

```
## [1] 81.18812
```

```
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=5)
```

```
print("K=5")
```

```
## [1] "K=5"
```

```
mean(knn1.pred == test$c.bool)*100
```

```
## [1] 80.19802
```

```
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=10)
```

```
print("K=10")
```

```
## [1] "K=10"
```

```
mean(knn1.pred == test$c.bool)*100
```

```
## [1] 77.22772
```

```
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=20)
```

```
print("K=20")
```

```
## [1] "K=20"
```

```
mean(knn1.pred == test$c.bool)*100
```

```
## [1] 73.26733
```

```
knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=50)
```

```
print("K=50")
```

```
## [1] "K=50"
```



```
mean(knn1.pred == test$c.bool)*100
## [1] 76.23762

knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=100)
print("K=100")
## [1] "K=100"

mean(knn1.pred == test$c.bool)*100
## [1] 75.24752

knn1.pred <- knn(train.X1, test.X1, train$c.bool, k=200)
print("K=200")
## [1] "K=200"

mean(knn1.pred == test$c.bool)*100
## [1] 75.24752
```

Se aprecia que para las variables indus + age el modelo clasifica correctamente los suburbios cuando su criminalidad esta por encima de la media el 81.20% cuando $k = 1$ y luego decrece lentamente para $k > 1$