Business Analytics - ETC3250 2018 - Lab 5 Solutions

Souhaib Ben Taieb

23 March 2018

Exercice 1

Read and run the code in Sections 4.6.1 to 4.6.6 of ISLR.

Assignment - Question 1

Exercise 7 in chapter 4.7 of ISLR.

Let $p_k(x)$ be the probability that a company will $(k = 1)$ or will not $(k = 0)$ issue a dividend this year given that its percentage profit was x last year.

Using Bayes theorem and since we assume X follows a normal distribution, we can write:

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x-\mu_k)^2)}{\sum_{l=1}^k \pi_l \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x-\mu_l)^2)}, \quad k = 1, 2.$$

Then, using $\pi_1 = .8$, $\sigma = 6$, $\mu_1 = 10$ and $\mu_2 = 0$, we have

$$p_1(x) = \frac{0.80 \exp(-\frac{1}{2*36}(x-10)^2)}{0.80 \exp(-\frac{1}{2*36}(x-10)^2) + 0.20 \exp(-\frac{1}{2*36}x^2)}.$$

Finally, since $x = 4$, we have

$$p_1(4) \approx 75\%.$$

Assignment - Question 2

Exercise 8 in chapter 4.7 of ISLR.

$E_{\text{train}} = 0.20$, $E_{\text{test}} = 0.30$ and $E_{\text{avg}} = 0.25$.

$E_{\text{train}} = x_1$, $E_{\text{test}} = x_2$ and $E_{\text{avg}} = 0.18$.

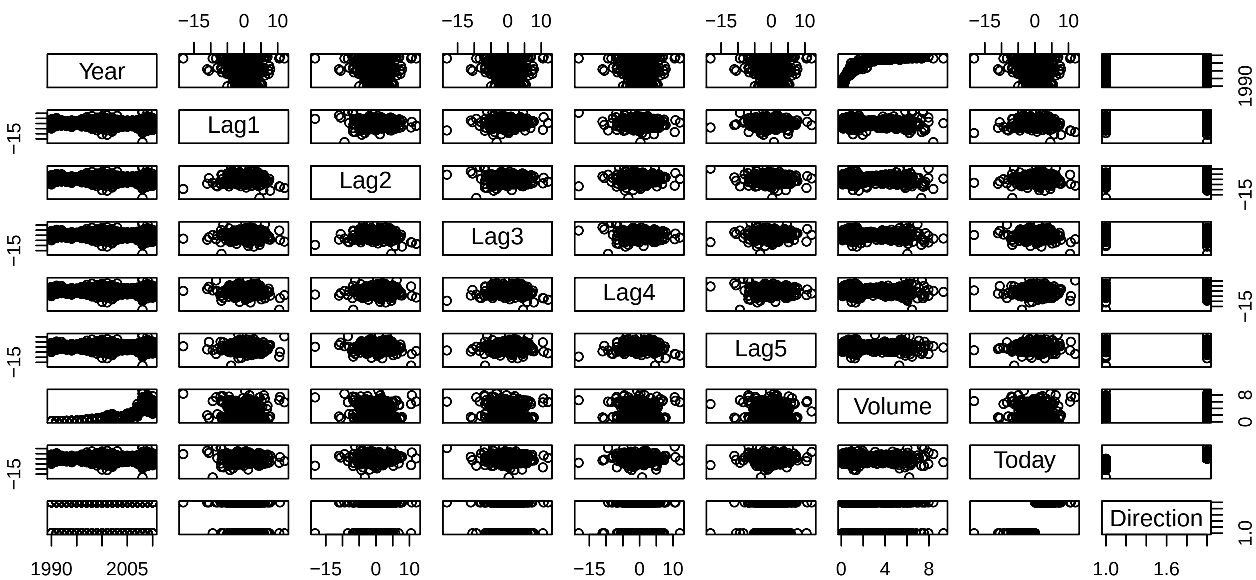
I would prefer logistic regression since the test error rate is not too far from the training error rate. For 1-nearest neighbors, which is typically a high variance classifier, we could obtain an average error rate of 0.18 with $x_1 = 0$ and $x_2 = 0.36$, which is an overfitting classifier with test error rate larger than 0.30.

Assignment - Question 3

Exercise 10 in chapter 4.7 of ISLR.

```r
library(ISLR)
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
```

```r
#  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
#  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
#  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
#  Median :  0.2380   Median :  0.2340   Median :1.00268
#  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
#  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
#  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
#      Today          Direction
#  Min.   :-18.1950   Down:484
#  1st Qu.: -1.1540   Up  :605
#  Median :  0.2410
#  Mean   :  0.1499
#  3rd Qu.:  1.4050
#  Max.   : 12.0260
pairs(Weekly)
```




```r
attach(Weekly)
```

```r
glm.fit <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly, family = binomial)
summary(glm.fit)
#
# Call:
# glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
#     Volume, family = binomial, data = Weekly)
#
# 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
```

```
glm.probs <- predict(glm.fit, type = "response")
glm.pred <- rep("Down", length(glm.probs))
glm.pred[glm.probs > 0.5] <- "Up"
table(glm.pred, Weekly$Direction)
#
# glm.pred Down  Up
#     Down   54  48
#     Up    430 557
```

```
train <- (Year <= 2008)
Weekly.test <- Weekly[!train, ]
glm.fit <- glm(Direction ~ Lag2, data = Weekly, family = binomial, subset = train)
glm.probs <- predict(glm.fit, Weekly.test, type = "response")
glm.pred <- rep("Down", length(glm.probs))
glm.pred[glm.probs > 0.5] <- "Up"
Direction.test <- Direction[!train]
table(glm.pred, Direction.test)
#         Direction.test
# glm.pred Down Up
#     Down    9  5
#     Up     34 56
mean(glm.pred == Direction.test)
# [1] 0.625
```

```
library(MASS)
lda.fit <- lda(Direction ~ Lag2, data = Weekly, subset = train)
lda.pred <- predict(lda.fit, Weekly.test)
table(lda.pred$class, Direction.test)
#       Direction.test
#         Down Up
#   Down    9  5
#   Up     34 56
mean(lda.pred$class == Direction.test)
# [1] 0.625
```

```
qda.fit <- qda(Direction ~ Lag2, data = Weekly, subset = train)
```

```r
qda.class <- predict(qda.fit, Weekly.test)$class
table(qda.class, Direction.test)
#          Direction.test
# qda.class Down Up
#      Down    0  0
#      Up     43 61
mean(qda.class == Direction.test)
# [1] 0.5865385
```

```r
library(class)
train.X <- as.matrix(Lag2[train])
test.X <- as.matrix(Lag2[!train])
train.Direction = Direction[train]
knn.pred = knn(train.X, test.X, train.Direction, k = 100, prob = T)
table(knn.pred, Direction.test)
#         Direction.test
# knn.pred Down Up
#     Down   10 13
#     Up     33 48
```

```r
glm.fit <- glm(Direction ~ Lag2:Lag1, data = Weekly, family = binomial, subset = train)
glm.probs <- predict(glm.fit, Weekly.test, type = "response")
glm.pred <- rep("Down", length(glm.probs))
glm.pred[glm.probs > 0.5] <- "Up"
Direction.0910 <- Direction[!train]
table(glm.pred, Direction.test)
#         Direction.test
# glm.pred Down Up
#     Down    1  1
#     Up     42 60
mean(glm.pred == Direction.test)
# [1] 0.5865385
```

```r
lda.fit = lda(Direction ~ Lag2:Lag1, data = Weekly, subset = train)
lda.pred = predict(lda.fit, Weekly.test)
mean(lda.pred$class == Direction.test)
# [1] 0.5769231
```

```r
for(k in seq(1, 100, by = 10)){
  knn.pred = knn(train.X, test.X, train.Direction, k = k, prob = T)
  print(table(knn.pred, Direction.test))
}
#         Direction.test
# knn.pred Down Up
#     Down   21 30
#     Up     22 31
#         Direction.test
# knn.pred Down Up
#     Down   18 21
#     Up     25 40
#         Direction.test
# knn.pred Down Up
#     Down   19 21
#     Up     24 40
#         Direction.test
```

```
# knn.pred Down Up
#     Down   19 25
#     Up     24 36
#         Direction.test
# knn.pred Down Up
#     Down   22 24
#     Up     21 37
#         Direction.test
# knn.pred Down Up
#     Down   20 23
#     Up     23 38
#         Direction.test
# knn.pred Down Up
#     Down   20 18
#     Up     23 43
#         Direction.test
# knn.pred Down Up
#     Down   13 14
#     Up     30 47
#         Direction.test
# knn.pred Down Up
#     Down   10 10
#     Up     33 51
#         Direction.test
# knn.pred Down Up
#     Down   10 11
#     Up     33 50
```

Assignment - Question 4

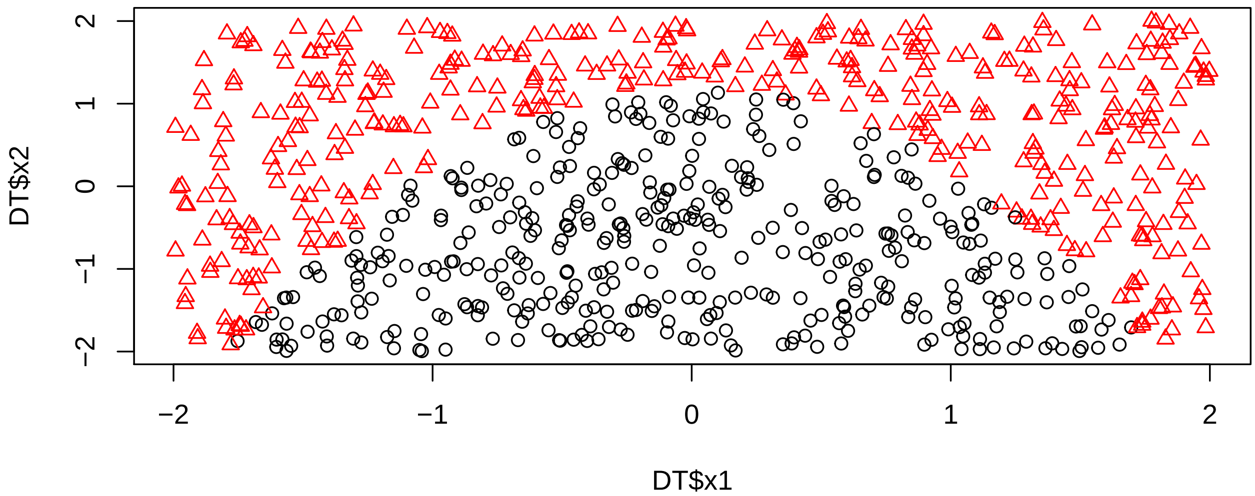
Download the file "data_lab5.Rdata" which contains two datasets D1 and D2, each with n = 800 points, $x \in \mathbb{R}^2$ and $y \in \{0, 1\}$.

- (1) Plot the data D1 with the class labels given by y. Run logistic regression, using the glm function in R. What is the training misclassification rate?

```
library(ggplot2)
load("data_lab5.Rdata")

# Choose which dataset you want to analyze:
DT <- D1
# DT <- D2

#qplot(x1, x2, data = D1, col = ifelse(y == 0, "red", "blue"))
plot(DT$x1, DT$x2, col = as.integer(DT$y)+1L, pch = as.integer(DT$y)+1L)
```




```
glm.fits <- glm(y~x1+x2, data = DT , family = binomial)
glm.probs <- predict(glm.fits, DT, type = "response")
glm.pred = rep(0, nrow(DT))
glm.pred[glm.probs >.5] = 1
mean(glm.pred==DT$y)
# [1] 0.765
```

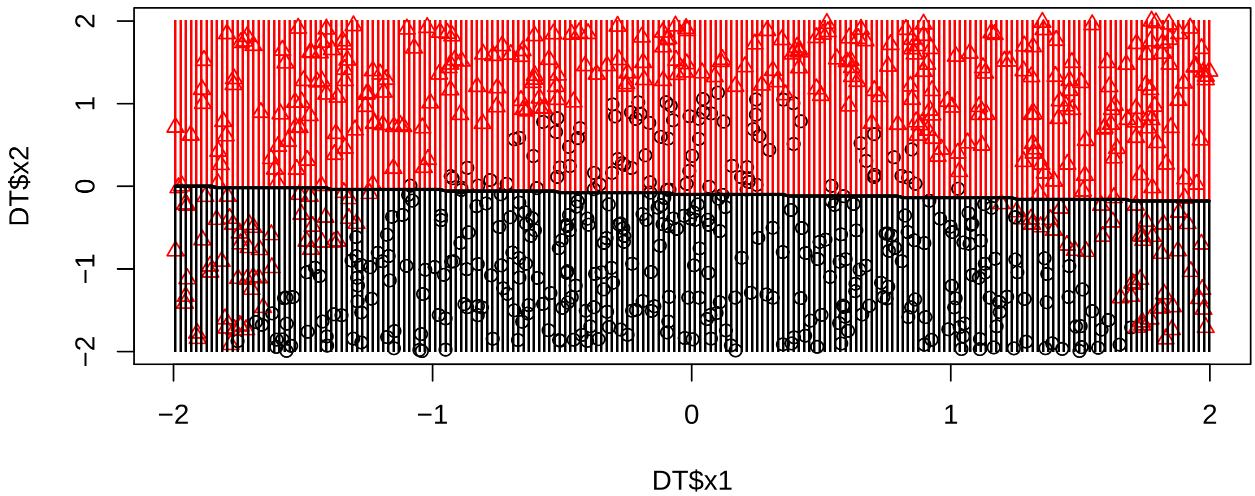
- (2) Draw the decision boundary in $\mathbb{R}_2$ of the logistic regression model from (1), on top of your plot from (1). What shape is it? Does this boundary look like it adequately separates the classes?

```
  # building the grid
resolution = 200
r <- sapply(DT[, -3], range, na.rm = TRUE)
xs <- seq(r[1,1], r[2,1], length.out = resolution)
ys <- seq(r[1,2], r[2,2], length.out = resolution)
g <- cbind(rep(xs, each=resolution), rep(ys, time = resolution))
colnames(g) <- colnames(r)
g <- as.data.frame(g)

# predictions for all points in the grid
p <- predict(glm.fits, g, type = "response")
pred = rep(0, nrow(g))
pred[p >.5] = 1

plot(DT$x1, DT$x2, col = as.integer(DT$y)+1L, pch = as.integer(DT$y)+1L)
points(g, col = as.integer(pred)+1L, pch = ".")

# Plot the contour of our function at level 0.5 (our function returns 0 or 1)
z <- matrix(as.integer(pred), nrow = resolution, byrow = TRUE)
  contour(xs, ys, z, add = TRUE, drawlabels = FALSE,
          lwd = 2, levels = .5)
```



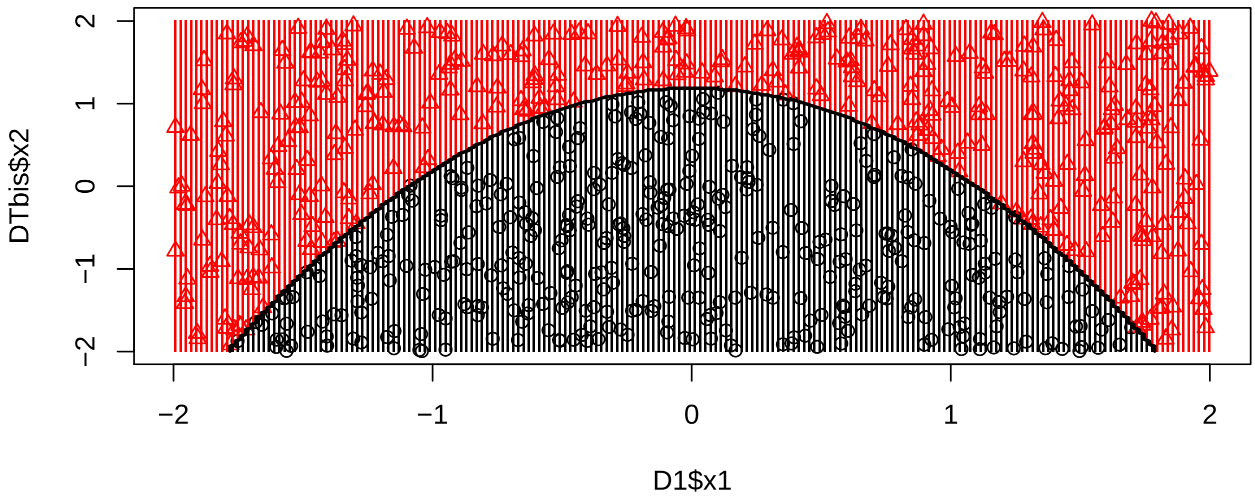
- (3) Run logistic regression on the predictors x_1 and x_2, as well as the predictor x_1^2. This is analogous to adding a quadratic term to a linear regression. What is the training misclassification rate? Why is this better than the model from (1)?

```
DTbis <- data.frame(DT, x3 = DT$x1^2)
glm.fits2 <- glm(y~x1+x2+ x3, data = DTbis , family = binomial)
glm.probs2 <- predict(glm.fits2, DTbis, type = "response")
glm.pred2 = rep(0, nrow(DTbis))
glm.pred2[glm.probs2 >.5] = 1
mean(glm.pred2 == DTbis$y)
# [1] 1


gbis <- data.frame(g, x3=g[, 1]^2)
p <- predict(glm.fits2, gbis, type = "response")
pred2 = rep(0, nrow(gbis))
pred2[p >.5] = 1

plot(D1$x1, DTbis$x2, col = as.integer(DTbis$y)+1L, pch = as.integer(DTbis$y)+1L)
points(g, col = as.integer(pred2)+1L, pch = ".")

# Plot the contour of our function at level 0.5 (our function returns 0 or 1)
z <- matrix(as.integer(pred2), nrow = resolution, byrow = TRUE)
  contour(xs, ys, z, add = TRUE, drawlabels = FALSE,
          lwd = 2, levels = .5)
```



- (4) Do (1), (2) and (3) for dataset D2. What additional predictors can you pass to logistic regression in order for it to do a better job of separating the classes? (Hint: draw a curve between the classes by eye. What shape does this have?)

Replace the following lines in the previous code

```
DTbis <- data.frame(DT, x3 = DT$x1^2, x4 = DT$x1^3)
glm.fits2 <- glm(y~x1+x2+ x3 + x4, data = DTbis , family = binomial)
gbis <- data.frame(g, x3=g[, 1]^2, x4=g[, 1]^3)
```

TURN IN

- Your `.Rmd` file (which should knit without errors and without assuming any packages have been pre-loaded)
- Your Word (or pdf) file that results from knitting the Rmd.
- DUE: April 1, 11:55pm (late submissions not allowed), loaded into moodle