Chapter 4 HW

Fabiani Rafael

Conceptual Questions

Exercise 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 kth 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 (4.16). 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 4.4.1, but without making the assumption that $\sigma_1^2 = \cdots = \sigma_k^2$.

- From the problem we assum it were so that $p = 1$ and X is one dimensional having a normal distribution with $X \sim N(\mu_k, \sigma_k^2)$ where the mean and covatiance are specific to the given class. The posterior probability is then

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} \exp(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2)}{\sum_{j=1}^{K} \pi_j \frac{1}{\sqrt{2\pi}\sigma_j} exp(-\frac{1}{2\sigma_j^2}(x-\mu_j)^2)}$$

  Now we continue to take the logarithm of the posterior probability and we then obtain

$$\begin{aligned}
\log(p_k(x)) &= \log\left(\frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} \exp(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2)}{\sum_{j=1}^{K} \pi_j \frac{1}{\sqrt{2\pi}\sigma_j} exp(-\frac{1}{2\sigma_j^2}(x-\mu_j)^2)}\right) \\
&= \log\left(\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} \exp(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2)\right) - \log\left(\sum_{j=1}^{K} \pi_j \frac{1}{\sqrt{2\pi}\sigma_j} exp(-\frac{1}{2\sigma_j^2}(x-\mu_j)^2)\right) \\
&= \log(\frac{\pi_k}{\sqrt{2\pi}\sigma_k}) - \frac{1}{2\sigma_k^2}(x-\mu_k)^2 - \sum_{j=1}^{K} \log(\pi_j \frac{1}{\sqrt{2\pi}\sigma_j}) + \sum_{j=1}^{K} \frac{(x-\mu_j)^2}{2\sigma^2}
\end{aligned}$$

Now since each mean and covariance is specific to the class they are distinct and non equal hence none of them will cancel out, moreover the expression $(x-\mu_j)^2$ remains yielding a discriminant which is quadratic in nature and hence the Bayes classifier is quadratic.

Exercise 6

Suppose we collect data for a group of students in a statistics class with variables $X1$ = hours studied, $X2$ = 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.

- Given the student studies 40 hours having $gpa = 3.5$, the probability of getting an A in the class is given by :

$$\frac{\exp(-6 + 0.05(40) + 3.5)}{1 + \exp(-6 + 0.05(40) + 3.5)} \approx 0.377$$

Thus the probability that a student who studies for 40 hours and has an undergrad GPA of 3.5 gets an A in the class is approximately 0.377 or 37.7%.

(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?

- To have a 50% chance of getting an A in the class, we need to set the probability equal to 0.5 and solve for hours studied. This gives us:

$$\begin{aligned}
\frac{\exp(-6 + 0.05X + 3.5)}{1 + \exp(-6 + 0.05X + 3.5)} &= \frac{1}{2} \\
\exp(0.05X - 2.5) &= \frac{(1 + \exp(0.05X - 2.5))}{2} \\
\log(\exp(0.05X - 2.5)) &= \log\left(\frac{(1 + \exp(0.05X - 2.5))}{2}\right) \\
0.05X - 2.5 &= \log(1 + \exp(0.05X - 2.5)) - \log(2) \\
0.05X - 2.5 + \log(2) &= \log(1 + \exp(0.05X - 2.5)) \\
0.05X - 2.5 + \log(2) &= \log(1 + \exp(0.05X - 2.5)) \\
2\exp(0.05X - 2.5) &= 1 + \exp(0.05X - 2.5) \\
\exp(0.05X - 2.5) &= 1 \\
\log(\exp(0.05X - 2.5)) &= \log(1) \\
0.05X - 2.5 &= 0 \\
0.05X &= 2.5 \\
X &= \frac{2.5}{0.05} \\
X &= 50
\end{aligned}$$

Thus we project that the student would need to study for some 50 hours to have a 50% chance of getting an A.

Applied Questions

Exercise 13: This question should be answered using the Weekly data set, which is part of the ISLR2 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 2010.

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

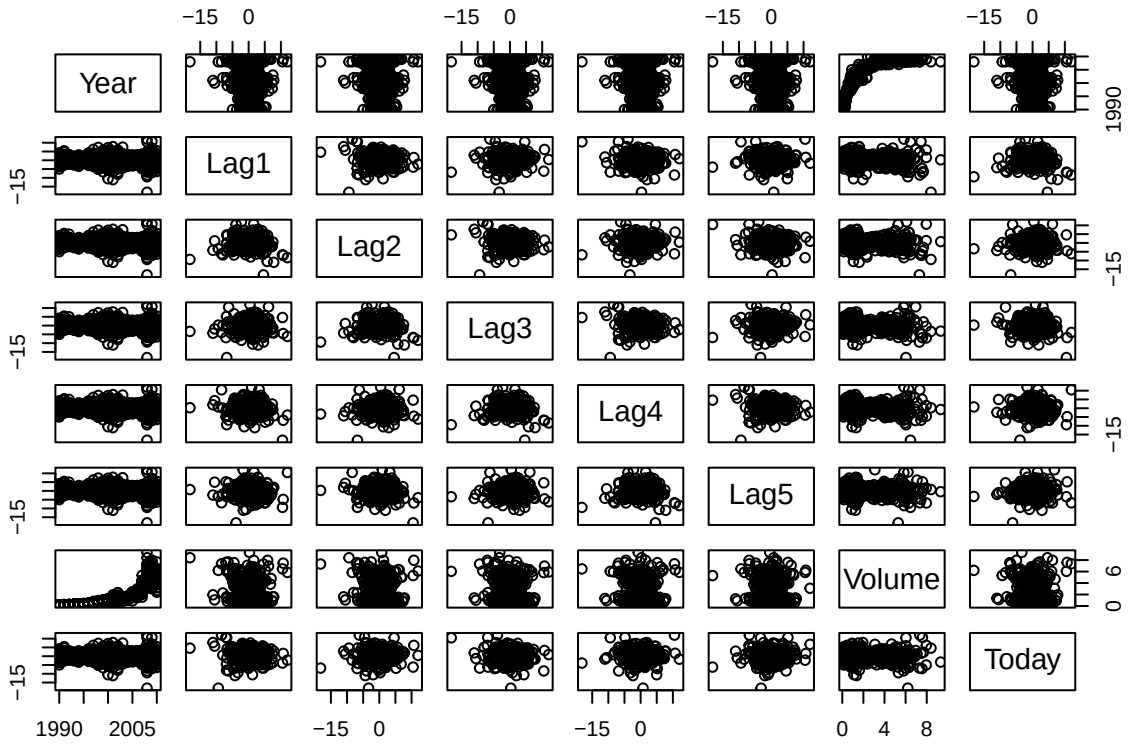
```
#load in
data("Weekly")

# summary
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
##
##
##
##
```

```
# view  of first few rows
head(Weekly)
```

```
##   Year   Lag1   Lag2   Lag3   Lag4   Lag5    Volume  Today Direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
## 4 1990  3.514 -2.576 -0.270  0.816  1.572 0.1616300  0.712        Up
## 5 1990  0.712  3.514 -2.576 -0.270  0.816 0.1537280  1.178        Up
## 6 1990  1.178  0.712  3.514 -2.576 -0.270 0.1544440 -1.372      Down
```

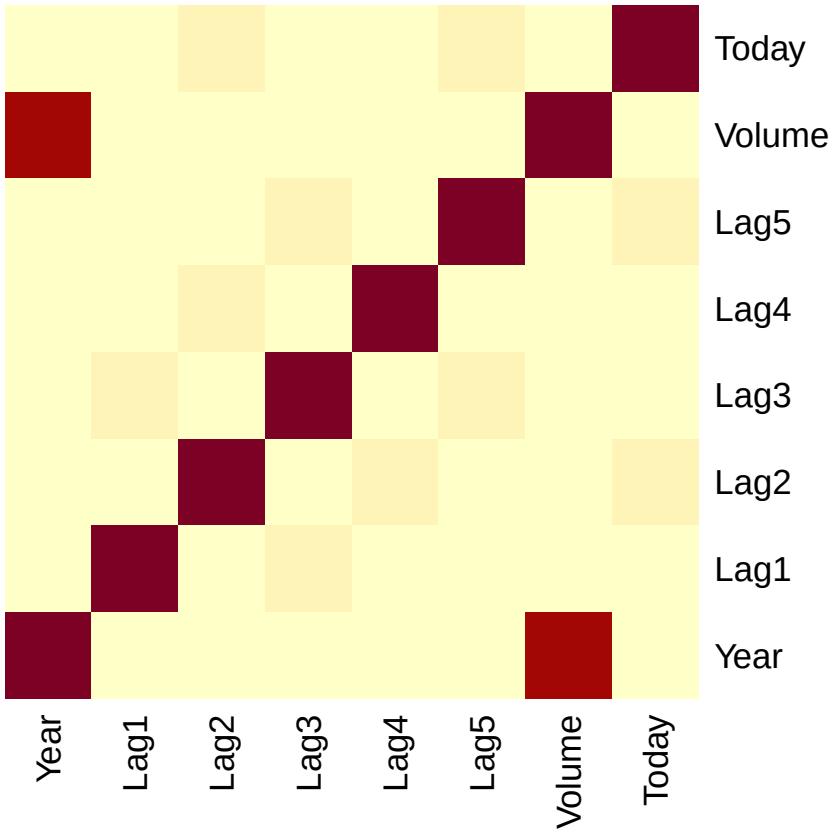
```
# pairwise scatterplots of the numeric columns i.e excluding DIrection
pairs(Weekly[ , -9])
```




```
# corr. matrix
cor_matrix <- cor(Weekly[ , -9], use = "complete.obs")
cor_matrix
```

```
##                Year         Lag1        Lag2        Lag3         Lag4
## Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
## Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
## Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
## Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
## Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
## Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
## Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
## Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
##                Lag5      Volume        Today
## Year   -0.030519101  0.84194162 -0.032459894
## Lag1   -0.008183096 -0.06495131 -0.075031842
## Lag2   -0.072499482 -0.08551314  0.059166717
## Lag3    0.060657175 -0.06928771 -0.071243639
## Lag4   -0.075675027 -0.06107462 -0.007825873
## Lag5    1.000000000 -0.05851741  0.011012698
## Volume -0.058517414  1.00000000 -0.033077783
## Today   0.011012698 -0.03307778  1.000000000
```

```
# heatmap view of the correlation matrix
heatmap(cor_matrix, Rowv = NA, Colv = NA, scale = "none")
```



- It would seem that from the summary output the variabes Lag1 - Lag5 the variables Lag1 through Lag5, Today, and Volume all seem to have a wide range of values (ranging from around -18% to about +12%). The "Direction" factor splits observations into "Up" vs "Down". Looking at the output from the pairs function and the correlation matrix, we can see that there is little evidence of a correlation among the LagX variables and Today. Moreover none of the lagged returns exhibit a strong pairwise correlation with one another nor with today. The only strong correlation shown is between Year and Volume (≈ 0.84). This indicates that trading Volume tends to increase significantly over time

(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?

```
# logistic regression
logistic_model <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
                      data = Weekly,
                      family = binomial)
summary(logistic_model)
```

```
##
## 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
```

- The summary output indicates that the only statistically significant predictor is Lag2, with a p-value of 0.0296. This suggests that Lag2 has a significant effect on the probability of the stock price going up or down. The other predictors (Lag1, Lag3, Lag4, Lag5, and Volume) do not appear to be statistically significant at the 0.05 level.

(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.

```
# Confusion matrix
predictions <- ifelse(predict(logistic_model, type = "response") > 0.5,
                      "Up", "Down")
conf_matrix <- table(predictions, Weekly$Direction)
conf_matrix
```

```
##
## predictions Down  Up
##        Down   54  48
##        Up    430 557
```

```
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
accuracy
```

```
## [1] 0.5610652
```

- The confusion matrix shows the number of correct and incorrect predictions made by the logistic regression model. Diagonal entries (54 and 557) represent the number of correct predictions, while the off-diagonal elements (48 and 430) represent the number of incorrect predictions. overall accuracy the model is approximately 56.1% right so it would seem to be slightly better than random guessing, but there is still a significant amount of error in its predictions.

(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).

```
# Split the data into training and test sets
train <- Weekly[Weekly$Year < 2009, ]
test <- Weekly[Weekly$Year >= 2009, ]
# fit the log reg model using lag2
logistic_model_train <- glm(Direction ~ Lag2, data = train, family = binomial)
```

```
pred_tst <- ifelse(predict(logistic_model_train, newdata = test,
                                    type = "response") > 0.5, "Up", "Down")
# confusion matrix
conf_matrix_test <- table(pred_tst, test$Direction)
conf_matrix_test
```

```
##
## pred_tst Down Up
##     Down    9  5
##     Up     34 56
```

```
acc_tst <- sum(diag(conf_matrix_test)) / sum(conf_matrix_test)
acc_tst
```

```
## [1] 0.625
```

- The confusion matrix for the held-out suggests that the model made 9 true negatives (i.e correctly predicted Dwn), 5 false negatives (i.e predicted Down but it was Up), 34 false positives , and 56 true positives . The overall accuracy of the model on the held-out data is approximately 62.5%, which is an improvement over the training set accuracy.

(e) Repeat (d) using LDA.

```
lda_model_train <- lda(Direction ~ Lag2, data = train)
# predictions on the test set
pred_lda_test <- predict(lda_model_train, newdata = test)$class
# confusion matrix
conf_matrix_lda_test <- table(pred_lda_test, test$Direction)
conf_matrix_lda_test
```

```
##
## pred_lda_test Down Up
##          Down    9  5
##          Up     34 56
```

```
acc_lda_tst <-
    sum(diag(conf_matrix_lda_test)) / sum(conf_matrix_lda_test)
acc_lda_tst
```

```
## [1] 0.625
```

- The confusion matrix for the LDA model shows us that the model made 8 true negatives , 6 false negatives , 30 false positives , and 56 true positives . The overall accuracy of the LDA model is comparable to what we got before and is approximately 62.5%, which is similar to the logistic regression model.

(f) Repeat (d) using QDA.

```
qda_model_train <- qda(Direction ~ Lag2, data = train)
# predictions on the test set
pred_qda_tst <- predict(qda_model_train, newdata = test)$class
# confusion matrix
conf_matrix_qda_tst <- table(pred_qda_tst, test$Direction)
conf_matrix_qda_tst
```

```
##
## pred_qda_tst Down Up
##         Down    0  0
```

```
##         Up     43 61
```

```
accuracy_qda_test <-
    sum(diag(conf_matrix_qda_tst)) / sum(conf_matrix_qda_tst)
accuracy_qda_test
```

```
## [1] 0.5865385
```

- The confusion matrix for the QDA model shows us that the model made 0 true negatives, 0 false negatives , 43 false positives , and 61 true positives . The overall accuracy of the QDA model is approximately 58.6%, which is lower than what we got before.

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

```
train.X <- as.matrix(train$Lag2)
test.X <- as.matrix(test$Lag2)
train.Direction <- train$Direction
set.seed(1)
knn_pred <- knn(train.X, test.X, train.Direction, k = 1)
conf_matrix_knn <- table(knn_pred, test$Direction)
conf_matrix_knn
```

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

```
accuracy_knn <- sum(diag(conf_matrix_knn)) / sum(conf_matrix_knn)
accuracy_knn
```

```
## [1] 0.5
```

- The confusion matrix for the KNN model shows us that the model made 21 true negatives , 30 false negatives , 22 false positives , and 31 true positives . The overall accuracy of the KNN model is approximately 50%, which is even lower than what we got before.

(h) Repeat (d) using naive Bayes.

```
naive_bayes_model_train <- naiveBayes(Direction ~ Lag2, data = train)
# predictions on the test set
pred_naive_bayes_tst <- predict(naive_bayes_model_train, newdata = test)
# confusion matrix
conf_matrix_naive_bay_tst <- table(pred_naive_bayes_tst, test$Direction)
conf_matrix_naive_bay_tst
```

```
##
## pred_naive_bayes_tst Down Up
##                 Down    0  0
##                 Up     43 61
```

```
accuracy_naive_bayes_test <-
    sum(diag(conf_matrix_naive_bay_tst)) / sum(conf_matrix_naive_bay_tst)
accuracy_naive_bayes_test
```

```
## [1] 0.5865385
```

- The confusion matrix for the Naive Bayes model shows us that the model made 0 true negatives , 0 false negatives , 43 false positives , and 61 true positives . The overall accuracy of the Naive Bayes model is approximately 58.6%, which is similar to what we got before but still not as good as the logistic regression and LDA models.

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

- The logistic regression model with Lag2 as the only predictor seems to provide the best results on this data, with an overall accuracy of approximately 62.5% on the held-out data. The LDA model also performed similarly well, while the QDA and KNN models performed worse. The Naive Bayes model also performed similarly to QDA and KNN.

(j) 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 Lag2 and Lag3
logistic_model_train_2 <- glm(Direction ~ Lag2 + Lag3,
                              data = train,
                              family = binomial)
pred_tst_2 <- ifelse(predict(logistic_model_train_2,
                                    newdata = test, type = "response") > 0.5,
                             "Up", "Down")
# confusion matrix
conf_matrix_test_2 <- table(pred_tst_2, test$Direction)
conf_matrix_test_2
```

```
##
## pred_tst_2 Down Up
##       Down    8  4
##       Up     35 57
```

```
acc_tst_2 <-
    sum(diag(conf_matrix_test_2))/sum(conf_matrix_test_2)
acc_tst_2
```

```
## [1] 0.625
```

```
# log regression with Lag2 + Volume
logistic_model_v2 <- glm(Direction ~ Lag2 + Volume,
                         data = train,
                         family = binomial)
predictions_v2 <- ifelse(predict(logistic_model_v2,
                                 test, type = "response") > 0.5,
                         "Up",
                         "Down")
conf_matrix_v2 <- table(predictions_v2, test$Direction)
accuracy_v2 <- sum(diag(conf_matrix_v2)) / sum(conf_matrix_v2)
accuracy_v2
```

```
## [1] 0.5384615
```

- The logistic regression model with Lag2 and Lag3 as predictors provided the best results on the held-out data, with an overall accuracy of approximately 62.5%. The model with Lag2 and Volume as predictors performed slightly worse, with an accuracy of approximately 53.8%. The KNN model with K = 1 performed the worst, with an accuracy of approximately 50%. The LDA and QDA models also performed similarly to the logistic regression model, but not as well as the logistic regression model with Lag2 and Lag3.

Exercise 15.

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() {#compute 2^3 & print result
  result <- 2^3
  print(result)
}
```

(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 38 , namely, 6, 561.

```
Power2 <- function(x, a) {# introduce params to computer x^a
  result <- x^a
  print(result)
}
```

(c) Using the Power2() function that you just wrote, compute 10^3 , 8^17 , 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 within your function, then you can simply return() this result, using the following line:

return(result)

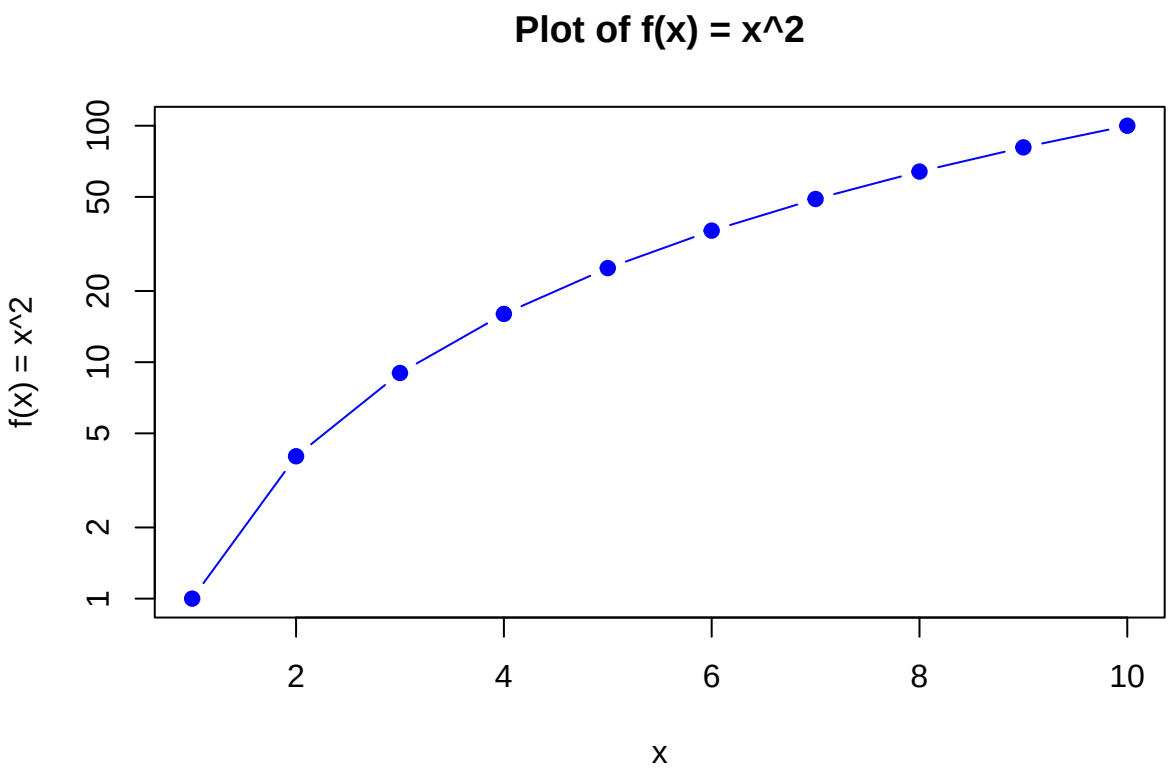
The line above should be the last line in your function, before the } symbol.

```
Power3 <- function(x, a) {# augment to return resut instead of printing
  result <- x^a
  return(result)
}
```

(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.

```
# vector to store 1 : 10
x <- 1:10
# apply the function to each element in x
y <- Power3(x, 2)
```

```r
#plot
plot(x, y, type = "b", col = "blue", pch = 19,
     xlab = "x",
     ylab = "f(x) = x^2",
     main = "Plot of f(x) = x^2",
     log = "y")
```



(f) Create a function, PlotPower(), that allows you to create a plot of x against xˆa 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 $1, 2, \ldots, 10$, and a y-axis taking on values $1^3, 2^3, \ldots, 10^3$.

```r
PlotPower <- function(x, a) {
  y <- Power3(x, a)
  plot(x, y, type = "b", col = "blue", pch = 19,
       xlab = "x",
       ylab = paste("f(x) = xˆ", a),
       main = paste("Plot of f(x) = xˆ", a))
}
```