```
library('ISLR') #library
data('Weekly')    #using weekly data set
#(a)
summary(Weekly) #summary the weekly dataset
dim(Smarket)       #find the dimension of the dataset
names(Weekly)     #find all the names of the dataset
par(mfrow = c(2, 4)) #plot the histogram for every variable with weekly data
hist(Weekly$Lag1)
hist(Weekly$Lag2)
hist(Weekly$Lag3)
hist(Weekly$Lag4)
hist(Weekly$Lag5)
hist(Weekly$Volume)
hist(Weekly$Today)
plot(Weekly$Year, Weekly$Today, type = "l") #plot the relationship between year and today
pairs(Weekly[, c(2:6, 8)])    #plot the pairs scatter points graph

#(b)
# Split the data into training and test sets
train <- Weekly[1:828, ]
test <- Weekly[829:1089, ]
fit.glm <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = train, family
= binomial)
summary(fit.glm)
# Fit LDA model
library(MASS)
fit.lda <- lda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = train)
# Fit QDA model
fit.qda <- qda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = train)
# Fit KNN model
library(class)
fit.knn <- knn(train[, c("Lag1", "Lag2", "Lag3", "Lag4", "Lag5", "Volume")], test[, c("Lag1",
"Lag2", "Lag3", "Lag4", "Lag5", "Volume")], train$Direction, k = 5)

# Make predictions using logistic regression
glm.probs <- predict(fit.glm, newdata = test, type = "response")
glm.pred <- ifelse(glm.probs > 0.5, "Up", "Down") #use if-else statement to convert to class
predictions

# Make predictions using LDA
lda.probs <- predict(fit.lda, newdata = test)$class
# Make predictions using QDA
qda.pred <- predict(fit.qda, newdata = test)$class
```

```
# Make predictions using KNN
knn.pred <- fit.knn #knn prediction is just what I did for fitting knn method

#(c)
# Compute confusion matrix for logistic regression
table(glm.pred, test$Direction)
# Compute confusion matrix for LDA
table(lda.pred, test$Direction)
# Compute confusion matrix for QDA
table(qda.pred, test$Direction)
# Compute confusion matrix for KNN
table(knn.pred, test$Direction)

#(d)
summary(fit.glm)

#(e)
# Calculate FPR and TPR under a given threshold(using another code function provided in class)
LR.fit = glm(Direction~Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,family=binomial, data = train)
LR.pred = predict(LR.fit, newdata = test, type="response")
roc_LR.curve=function(s,print=FALSE){
   Ps=(LR.pred>s)*1
   FP=sum((Ps==1)*(test$Direction=="Down"))/sum(test$Direction=="Down")
   TP=sum((Ps==1)*(test$Direction=="Up"))/sum(test$Direction=="Up")
   if(print==TRUE){
      print(table(Observed=test$Direction,Predicted=Ps))
   }
   vect=c(FP,TP)
   names(vect)=c("FPR","TPR")
   return(vect)
}
threshold=0.4
roc_LR.curve(threshold,print=TRUE)

library(MASS)
LDA.fit = lda(Direction~Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,data=train)
LDA.pred0 = predict(LDA.fit,newdata = test, type="response")
LDA.pred = LDA.pred0$posterior[,2]
roc_LDA.curve=function(s,print=FALSE){
   Ps=(LDA.pred>s)*1
   FP=sum((Ps==1)*(test$Direction=="Down"))/sum(test$Direction=="Down")
   TP=sum((Ps==1)*(test$Direction=="Up"))/sum(test$Direction=="Up")
```

```
  if(print==TRUE){
    print(table(Observed=test$Direction,Predicted=Ps))
  }
  vect=c(FP,TP)
  names(vect)=c("FPR","TPR")
  return(vect)
}
threshold=0.4
roc_LDA.curve(threshold,print=TRUE)

QDA.fit = qda(Direction~Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,data=train)
QDA.pred0 = predict(QDA.fit,newdata = test, type="response")
QDA.pred = QDA.pred0$posterior[,2]
roc_qda.curve=function(s,print=FALSE){
  Ps=(QDA.pred>s)*1
  FP=sum((Ps==1)*(test$Direction=="Down"))/sum(test$Direction=="Down")
  TP=sum((Ps==1)*(test$Direction=="Up"))/sum(test$Direction=="Up")
  if(print==TRUE){
    print(table(Observed=test$Direction,Predicted=Ps))
  }
  vect=c(FP,TP)
  names(vect)=c("FPR","TPR")
  return(vect)
}
threshold=0.4
roc_qda.curve(threshold,print=TRUE)

fit.knn <- knn(train[, c("Lag1", "Lag2", "Lag3", "Lag4", "Lag5", "Volume")], test[, c("Lag1",
"Lag2", "Lag3", "Lag4", "Lag5", "Volume")], train$Direction, k = 5)
knn.probs <- as.numeric(fit.knn == "Up")
roc_knn.curve=function(s,print=FALSE){
  Ps=(knn.probs>s)*1
  FP=sum((Ps==1)*(test$Direction=="Down"))/sum(test$Direction=="Down")
  TP=sum((Ps==1)*(test$Direction=="Up"))/sum(test$Direction=="Up")
  if(print==TRUE){
    print(table(Observed=test$Direction,Predicted=Ps))
  }
  vect=c(FP,TP)
  names(vect)=c("FPR","TPR")
  return(vect)
}
threshold=0.4
roc_knn.curve(threshold,print=TRUE)
```

```
#(f)
## Plot ROC curve
ROC_LR.curve=Vectorize(roc_LR.curve)
M.ROC=ROC.curve(seq(0,1,by=0.01))
plot(M.ROC[1,],M.ROC[2,],col="grey",lwd=2,type="l",xlab="False
positive rate",ylab="True positive rate")

ROC_LDA.curve=Vectorize(roc_LDA.curve)
M.ROC=ROC.curve(seq(0,1,by=0.01))
plot(M.ROC[1,],M.ROC[2,],col="grey",lwd=2,type="l",xlab="False
positive rate",ylab="True positive rate")

ROC_QDA.curve=Vectorize(roc_qda.curve)
M.ROC=ROC.curve(seq(0,1,by=0.01))
plot(M.ROC[1,],M.ROC[2,],col="grey",lwd=2,type="l",xlab="False
positive rate",ylab="True positive rate")

# Create ROC curve for KNN model
library(pROC)
knn.probs <- as.numeric(fit.knn == "Up")
knn.roc <- roc(response = test$Direction, predictor = knn.probs)
plot(knn.roc, main = "ROC Curve for KNN Model", col = "red")

#(g)
# Transform lag variables to categorical based on whether they are positive or negative
Weekly$lag1_dir <- ifelse(Weekly$Lag1 >= 0, "pos", "neg")
Weekly$lag2_dir <- ifelse(Weekly$Lag2 >= 0, "pos", "neg")
Weekly$lag3_dir <- ifelse(Weekly$Lag3 >= 0, "pos", "neg")
Weekly$lag4_dir <- ifelse(Weekly$Lag4 >= 0, "pos", "neg")
Weekly$lag5_dir <- ifelse(Weekly$Lag5 >= 0, "pos", "neg")

# Develop Logistic regression model with the transformed lag variables and numerical
Volume variable
glm.fit <- glm(Direction ~ lag1_dir + lag2_dir + lag3_dir + lag4_dir + lag5_dir + Volume,
data=Weekly, family=binomial, subset=Year<2009)
summary(glm.fit)

# Develop LDA model with the transformed lag variables and numerical Volume variable
library(MASS)
lda.fit <- lda(Direction ~ lag1_dir + lag2_dir + lag3_dir + lag4_dir + lag5_dir + Volume,
data=Weekly, subset=Year<2009)
lda.fit

# Develop QDA model with the transformed lag variables and numerical Volume variable
```

```
qda.fit <- qda(Direction ~ lag1_dir + lag2_dir + lag3_dir + lag4_dir + lag5_dir + Volume, data=Weekly, subset=Year<2009)
qda.fit

# Develop KNN model with the transformed lag variables and numerical Volume variable
  Weekly <- Weekly[complete.cases(Weekly),]
  # Split the data into training and test datasets
  train <- Weekly[Weekly$Year < 2009,]
  test <- Weekly[Weekly$Year >= 2009,]

# Convert lag variables to categorical based on whether they are positive or negative
  train[,c("lag1_dir", "lag2_dir", "lag3_dir", "lag4_dir", "lag5_dir")] <- lapply(train[,c("lag1", "lag2", "lag3", "lag4", "lag5")], function(x) ifelse(x >= 0, "Up", "Down"))

# Fit the KNN model with k = 3
  library(class)
  knn.pred <- knn(train[, c("lag1_dir", "lag2_dir", "lag3_dir", "lag4_dir", "lag5_dir", "Volume")], test[, c("lag1_dir", "lag2_dir", "lag3_dir", "lag4_dir", "lag5_dir", "Volume")], train$Direction, k = 3)

# Compute the confusion matrix
  table(knn.pred, test$Direction)
```