Module 4: Solutions to Recommended Exercises

TMA4268 Statistical Learning V2022

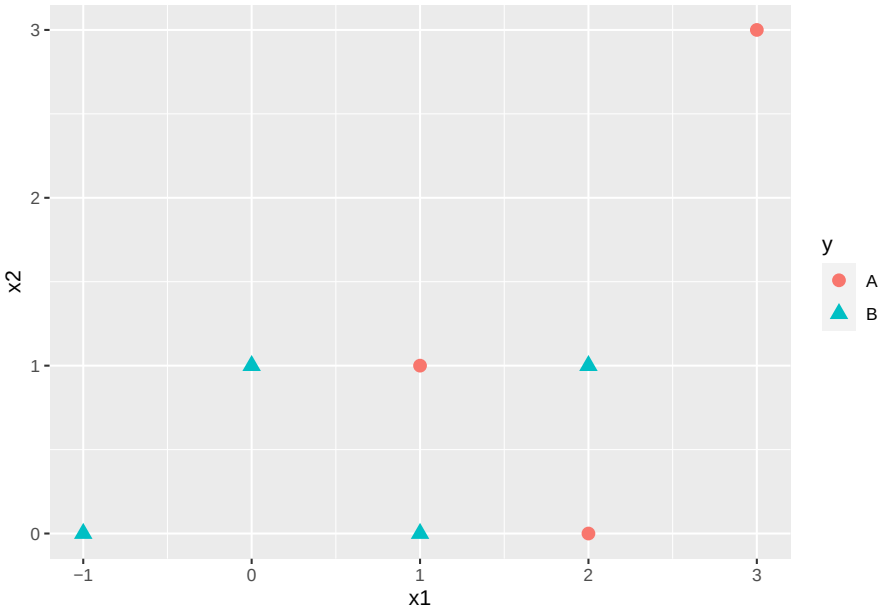
Emma Skarstein, Daesoo Lee, Stefanie Muff
Department of Mathematical Sciences, NTNU

January 31, 2022

Problem 1: KNN

The table and plot below provides a training data set consisting of seven observations, two predictors and one qualitative response variable.

```
##   x1 x2 y
## 1  3  3 A
## 2  2  0 A
## 3  1  1 A
## 4  0  1 B
## 5 -1  0 B
## 6  2  1 B
## 7  1  0 B
```



We wish to use this data set to make a prediction for Y when $X_1 = 1, X_2 = 2$ using the K-nearest neighbors classification method.

a)

Observation 1: $\sqrt{(3-1)^2+(3-2)^2} = \sqrt{5} = 2.24$

Observation 2: $\sqrt{(2-1)^2+(0-2)^2} = \sqrt{5} = 2.24$

Observation 3: $\sqrt{(1-1)^2+(1-2)^2} = 1$

Observation 4: $\sqrt{(0-1)^2+(1-2)^2} = \sqrt{2} = 1.41$

Observation 5: $\sqrt{(-1-1)^2+(0-2)^2} = \sqrt{8} = 2.83$

Observation 6: $\sqrt{(2-1)^2+(1-2)^2} = \sqrt{2} = 1.41$

Observation 7: $\sqrt{(1-1)^2+(0-2)^2} = 2$

b)

When $K = 1$ the nearest neighbor is observation 3 $(1, 1)$ which is class A. The predicted probabilities for class A and B are then,

$$P(Y = A|X = x_0) = \frac{1}{1}\sum_{i\in\mathcal{N}_0} I(y_i = A) = 1$$

$$P(Y = B|X = x_0) = \frac{1}{1}\sum_{i\in\mathcal{N}_0} I(y_i = B) = 0,$$

and we classify the point as A. For $K = 4$ the nearest neighbors are observations 3, 4, 6 and 7 with corresponding classes {A,B,B,B} and the predicted probabilities for class A and B for point $(1, 2)$ are then

$$P(Y = A|X = x_0) = \frac{1}{4}\sum_{i\in\mathcal{N}_0} I(y_i = A) = \frac{1}{4}(1+0+0+0) = 0.25$$

$$P(Y = B|X = x_0) = \frac{1}{4}\sum_{i\in\mathcal{N}_0} I(y_i = B) = \frac{1}{4}(0+1+1+1) = 0.75,$$

and we classify the point as B. For $K = 7$ we use all our observations to predict the probabilties,

$$P(Y = A|X = x_0) = \frac{1}{7}\sum_{i\in\mathcal{N}_0} I(y_i = A) = \frac{1}{7}(1+1+1+0+0+0+0) = \frac{3}{7} = 0.429,$$

$$P(Y = B|X = x_0) = \frac{1}{7}\sum_{i\in\mathcal{N}_0} I(y_i = B) = \frac{1}{7}(0+0+0+1+1+1+1) = \frac{4}{7} = 0.571,$$

and we classify the point as B.

Using all observations for classification with KNN, we will always choose the class that are most represented in the data (B in this case), and the predicted class will be B regardless of the position of the new point.

c)

If the Bayes decision boundary is highly non-linear, we would choose a K that is not to big, as the decision boundary becomes approximately linear when K tends to infinity.

Problem 2: Bank notes and LDA

Here we have measures of the length and the diagonal of an image part of $n_G = 500$ genuine bank notes and $n_F = 500$ false bank notes, with the following mean and covariance matrices,

$$\mu_G = \bar{\mathbf{x}}_G = \begin{bmatrix} 214.97 \\ 141.52 \end{bmatrix} \text{ and } \hat{\mathbf{\Sigma}}_G = \begin{bmatrix} 0.1502 & 0.0055 \\ 0.0055 & 0.1998 \end{bmatrix}$$

$$\mu_F = \bar{\mathbf{x}}_F = \begin{bmatrix} 214.82 \\ 139.45 \end{bmatrix} \text{ and } \hat{\mathbf{\Sigma}}_F = \begin{bmatrix} 0.1240 & 0.0116 \\ 0.0116 & 0.3112 \end{bmatrix}$$

a)

Assuming the observations x_G and x_F are independent observations from normal distribution with the same covariance matrix $\mathbf{\Sigma} = \mathbf{\Sigma}_G = \mathbf{\Sigma}_F$, and all observations are independent of each other, we can find the estimated pooled covariance matrix as

$$\begin{aligned} \hat{\mathbf{\Sigma}} &= \frac{(n_G - 1)\hat{\mathbf{\Sigma}}_G + (n_F - 1)\hat{\mathbf{\Sigma}}_F}{n_G + n_F - 2} \\ &= \begin{bmatrix} 0.13710 & 0.00855 \\ 0.00855 & 0.25550 \end{bmatrix} \end{aligned}$$

b)

For LDA we assume that the class conditional distributions are normal (Gaussian) and that all of the classes have the same covariance matrix. Assuming the same covariance matrix for both classes (G and F), we classify the new observation, x_0 based on which of the discriminant functions that are the largest,

$$\delta_k(\mathbf{x}) = \mathbf{x^T \Sigma^{-1} \mu_k} - \frac{\mathbf{1}}{\mathbf{2}} \boldsymbol{\mu}_\mathbf{k}^\mathbf{T} \mathbf{\Sigma^{-1}} \boldsymbol{\mu}_\mathbf{k} + \log \pi_\mathbf{k}.$$

We have not been given any information of the prior probabilites - but we would believe that the probability of a fake bank note is much smaller than the probability of a genuine bank note.

However, since our training data is 50% fake and genuine, we might use that to estimate prior probabilites, $\hat{\pi}_G = \hat{\pi}_F = \frac{n_F}{n} = 0.5$. Inserting the pooled covariance matrix and the estimated mean values, we have that

$$\delta_G(\mathbf{x_0}) = \mathbf{x_0}^T \hat{\mathbf{\Sigma}}^{-1} \boldsymbol{\mu}_G - \frac{1}{2} \boldsymbol{\mu}_G^T \hat{\mathbf{\Sigma}}^{-1} \boldsymbol{\mu}_G + \log \pi_G.$$

and

$$\delta_F(\mathbf{x_0}) = \mathbf{x_0}^T \hat{\mathbf{\Sigma}}^{-1} \boldsymbol{\mu}_F - \frac{1}{2} \boldsymbol{\mu}_F^T \hat{\mathbf{\Sigma}}^{-1} \boldsymbol{\mu}_F + \log \pi_F.$$

Alternatively: The rule would be to classify to G if $\delta_G(\mathbf{x}) - \delta_F(\mathbf{x}) > 0$, which can be written

$$\mathbf{x_0}^T \hat{\mathbf{\Sigma}}^{-1} (\boldsymbol{\mu}_G - \boldsymbol{\mu}_F) - \frac{1}{2} \boldsymbol{\mu}_G^T \hat{\mathbf{\Sigma}}^{-1} \boldsymbol{\mu}_G + \frac{1}{2} \boldsymbol{\mu}_F^T \hat{\mathbf{\Sigma}}^{-1} \boldsymbol{\mu}_F + (\log \pi_G - \log \pi_F) > 0$$

c)

With $\mathbf{x_0} = [214.0, 140.4]^T$ and using the formula given in the exercise, $\hat{\mathbf{\Sigma}}^{-1} = \begin{bmatrix} 7.31 & -0.24 \\ -0.24 & 3.92 \end{bmatrix}$. Inserting these into the formulas, we have that

$$\delta_G(\mathbf{x_0}) = [214\ 140.4] \begin{bmatrix} 7.31 & -0.24 \\ -0.24 & 3.92 \end{bmatrix} \begin{bmatrix} 214.97 \\ 141.52 \end{bmatrix} - \frac{1}{2}[214.97\ 141.52] \begin{bmatrix} 7.31 & -0.24 \\ -0.24 & 3.92 \end{bmatrix} \begin{bmatrix} 214.97 \\ 141.52 \end{bmatrix} + \log 0.5$$
$$= 198667.1$$

and

$$\delta_F(\mathbf{x_0}) = [214\ 140.4] \begin{bmatrix} 7.31 & -0.24 \\ -0.24 & 3.92 \end{bmatrix} \begin{bmatrix} 214.82 \\ 139.42 \end{bmatrix} - \frac{1}{2}[214.82\ 139.42] \begin{bmatrix} 7.31 & -0.24 \\ -0.24 & 3.92 \end{bmatrix} \begin{bmatrix} 214.82 \\ 139.42 \end{bmatrix} + \log 0.5$$
$$= 198668.3$$

Since $\delta_F(\mathbf{x_0})$ is larger than $\delta_G(\mathbf{x_0})$, we classify the bank note as fake!

Doing all these calulations in R:

```
xg = c(214.97, 141.52)
xf = c(214.82, 139.45)
sigmaG = matrix(c(0.1502, 0.0055, 0.0055, 0.1998), byrow = T, ncol=2)
sigmaF = matrix(c(0.1240, 0.0116, 0.0116, 0.3112), byrow = T, ncol=2)
sigma = (499*sigmaG + 499*sigmaF)/(998)
x = c(214.0, 140.4)
#lda discriminant function
a = t(x)%*%solve(sigma)%*%xg -0.5*t(xg)%*%solve(sigma)%*%xg - log(0.5)
b = t(x)%*%solve(sigma)%*%xf -0.5*t(xf)%*%solve(sigma)%*%xf - log(0.5)
```

d)

The difference between LDA and QDA is the QDA assumes that each class has its own covariance matrix, such that an observation from class k is on the form $X \sim N(\mu_k, \mathbf{\Sigma_k})$ where $\mathbf{\Sigma_k}$ is the covariance matrix. The discriminant function for class k is

$$\delta_k(\mathbf{x}) = -\frac{\mathbf{1}}{\mathbf{2}}\mathbf{x^T}\mathbf{\Sigma_k}^{-\mathbf{1}}\mathbf{x} + \mathbf{x^T}\mathbf{\Sigma_k}^{-\mathbf{1}}\boldsymbol{\mu}_\mathbf{k} - \frac{\mathbf{1}}{\mathbf{2}}\boldsymbol{\mu}_\mathbf{k}^\mathbf{T}\mathbf{\Sigma_k}^{-\mathbf{1}}\boldsymbol{\mu}_\mathbf{k} - \frac{\mathbf{1}}{\mathbf{2}}\log|\mathbf{\Sigma_k}| + \log \pi_\mathbf{k},$$

which unlike the disscriminant function for LDA is a quadratic function.

Using the QDA for classification of the bank note from c., we use that $\hat{\mathbf{\Sigma}}_{\boldsymbol{G}}^{-1} = \begin{bmatrix} 6.66 & -0.18 \\ -0.18 & 5.01 \end{bmatrix}$ and $\hat{\mathbf{\Sigma}}_{\boldsymbol{F}}^{-1} = \begin{bmatrix} 8.09 & -0.30 \\ -0.30 & 3.22 \end{bmatrix}$. Inserting the estimated values into the discriminant function, we have that

$$\begin{aligned}
\delta_G(\mathbf{x_0}) = &-\frac{1}{2}[214\ 140.4] \begin{bmatrix} 6.66 & -0.18 \\ -0.18 & 5.01 \end{bmatrix} \begin{bmatrix} 214 \\ 140.4 \end{bmatrix} \\
&+ [214\ 140.4] \begin{bmatrix} 6.66 & -0.18 \\ -0.18 & 5.01 \end{bmatrix} \begin{bmatrix} 214.97 \\ 141.52 \end{bmatrix} \\
&- \frac{1}{2}[214.97\ 141.52] \begin{bmatrix} 6.66 & -0.18 \\ -0.18 & 5.01 \end{bmatrix} \begin{bmatrix} 214.97 \\ 141.52 \end{bmatrix} \\
&- \frac{1}{2}\log(6.66 \cdot 5.01 - 0.18^2) + \log(0.5) \\
= &-5.018
\end{aligned}$$

and

$$
\begin{aligned}
\delta_F(\mathbf{x_0}) = &-\frac{1}{2}\,[214\ 140.4]\begin{bmatrix} 8.09 & -0.30 \\ -0.30 & 3.22 \end{bmatrix}\begin{bmatrix} 214 \\ 140.4 \end{bmatrix} \\
&+ [214\ 140.4]\begin{bmatrix} 8.09 & -0.30 \\ -0.30 & 3.22 \end{bmatrix}\begin{bmatrix} 214.82 \\ 139.45 \end{bmatrix} \\
&-\frac{1}{2}\,[214.82\ 139.45]\begin{bmatrix} 8.09 & -0.30 \\ -0.30 & 3.22 \end{bmatrix}\begin{bmatrix} 214.82 \\ 139.45 \end{bmatrix} \\
&-\frac{1}{2}\log(8.09 \cdot 3.22 - 0.30^2) + \log(0.5) \\
= &-3.47
\end{aligned}
$$

Since $\delta_F(\mathbf{x_0})$ is larger than $\delta_G(\mathbf{x_0})$ we classify the bank note as fake using QDA, the same result as for LDA.

R code to get to these results:

```
#qda discriminant function
a = -0.5*t(x)%*%solve(sigmaG)%*%x + t(x)%*%solve(sigmaG)%*%xg -
  0.5*t(xg)%*%solve(sigmaG)%*%xg - 0.5*log(det(sigmaG)) -log(2)
b = -0.5*t(x)%*%solve(sigmaF)%*%x + t(x)%*%solve(sigmaF)%*%xf -
  0.5*t(xf)%*%solve(sigmaF)%*%xf - 0.5*log(det(sigmaF)) -log(2)
```

Problem 3: Odds

a)

Recall that the odds ratio is defined as

$$\text{odds} = \frac{p}{1-p},$$

where p is the probability of some event. We know that the odds ratio is equal to 0.37. We thus solve for p:

$$
\begin{aligned}
\frac{p}{1-p} &= 0.37 \\
p &= 0.37(1-p) \\
p &= \frac{0.37}{1.37} = 0.270
\end{aligned}
$$

b)

For an individual we are given that $p = 0.16$. We are asked to find the odds that she will default. We can calculate this by inserting the probability of default into the formula for the odds ratio:

$$\frac{p}{1-p} = \frac{0.16}{1-0.16} = 0.19$$

Problem 4: Logistic regression

a)

We have that X_1 = hours studied = 40 and X_2 = undergrad GPA = 3.5. The formula for the predicted probability is

$$\begin{aligned} P(Y = A \mid X_1 = 40, X_2 = 3.5) &= \frac{\exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2)}{1 + \exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2)} \\ &= \frac{\exp(-6 + 0.05 \cdot 40 + 1 \cdot 3.5)}{1 + \exp(-6 + 0.05 \cdot 40 + 1 \cdot 3.5)} \\ &\approx 0.378 \end{aligned}$$

b)

We know that $x_2 = 3.5$, and need to solve $\hat{P}(Y = A \mid x_1, x_2 = 3.5) = 0.5$ for x_1.

$$\begin{aligned} 0.5 &= \frac{\exp(-6 + 0.05x_1 + 3.5)}{1 + \exp(-6 + 0.05x_1 + 3.5)} \\ 0.5(1 + \exp(-2.5 + 0.05x_1)) &= \exp(-2.5 + 0.05x_1) \\ 0.5 &= (1 - 0.5)\exp(-2.5 + 0.05x_1) \\ \log(1) &= -2.5 + 0.05x_1 \\ x_1 &= 50 \text{ hours} \end{aligned}$$

Problem 5: Sensitivity, specificity, ROC and AUC

a)

We start by denoting the number of true diseased as P and the number of true non-diseased as N, and we have that $n = P + N$. We count the ones with predicted probability of disease $p(x) > 0.5$, and denote the count P^* and the count for the ones with $p(x) \leq 0.5$ as N^*. Then, we can make the confusion table

	Predicted non-diseased -	Predicted diseased +	Total
True non-diseased -	TN	FP	N
True diseased +	FN	TP	P
Total	N^*	P^*	n

where TN (true negative) is the number of predicted non-diseased that are actually non-diseased, FP (false positive) is the number of predicted diseased that are actually non-diseased, FN (false negative) is the number of predicted non-diseased that are actually diseased and TP (true positive) is the number of predicted diseased that are actually diseased. Using the confusion table, we can calculate the sensitivity and specificity where

$$Sens = \frac{\text{True positive}}{\text{Actual positive}} = \frac{TP}{P}$$

and

$$Spes = \frac{\text{True negative}}{\text{Actual negative}} = \frac{TN}{N}$$

b)

In the ROC-curve we plot the sensitivity against 1-specificity for all possible thresholds of the probability. To construct the ROC-curve we would have to calculate the sensitivity and specificity for different values of the cutoff $p(x) > cut$. Using a threshold of 0.5, you say that if a new person has a probability of 0.51 of having the disease, he is classified as diseased. Another person with a probability of 0.49 would then be classified as non-diseased. However, using another cutoff could easily lead to a better balance between sensitivity and specificity (i.e., the point could lie closer to the top-left corner than for a 0.5 cut-off). Moreover, depending on the cost of a false positive or a false negative, the cut-off could be adjusted. It is for example often less severe to generate a false positive (i.e., a positive testresult although a person is healty) than a false negative (i.e., you overlook a disease), in which case the sensitivity is increased at a cost of a lower specificity.

c)

The AUC is the area under the ROC-curve and gives the overall performance of the test for all possible thresholds. An AUC value of 1 means a perfect fit for all possible thresholds, while a AUC of 0.5 corresponds to the classifier that performs no better than chance. Hence, a classification method $p(x)$ giving 0.6 and another $q(x)$ giving 0.7, we would probably prefer $q(x)$.

Data analysis in R

Problem 6

a)

Install the `ISLR` package and load the `ggplot2` and `GGally` libraries.

```
# install.packages("ISLR")
library(ISLR)
library(ggplot2)
library(GGally)
```

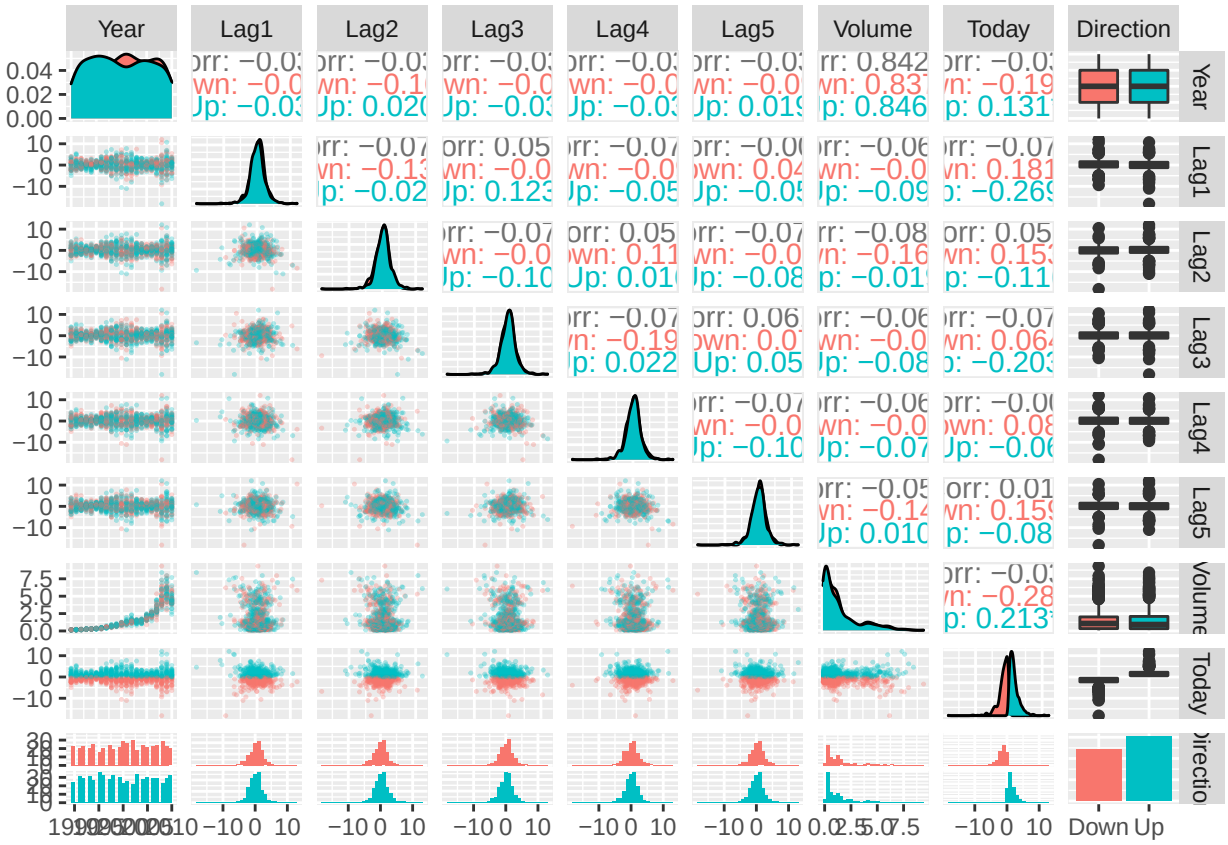
Now make a summary of the `Weekly` data set using the `summary` function and pairwise plots of the variables using the `ggpairs` function:

```
data("Weekly")
?Weekly
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
##
##
##
##
```

```
ggpairs(Weekly, aes(color=Direction), lower = list(continuous = wrap("points", alpha = 0.3, size=0.2)))
```



We can observe that the variables `Year` and `Volume` are highly correlated and it might look like the `Volume` increases quadratically with increasing `Year`. No other clear patterns are observed.

b)

We fit a logistic regression model using the `glm` function and specifying that we want to fit a binomial function by giving `function="binomial"` as an argument.

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

```
##
## 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
```

`Lag2` has a relatively low p-value, so it could be interesting to understand the connection. However, remember that `Lag2` is not necessarily causally related to `Direction`, and it could also well be that the p-value is small just by chance.

c)

We use our fitted model to calculate the probabilities for `Direction="Up"` for the response variable and compare the predictions with the true classes of the response variable. To find the confusion matrix, the function `table` can be used.

```
glm.probs_Weekly = predict(glm.Weekly, type="response")
glm.preds_Weekly = ifelse(glm.probs_Weekly > 0.5, "Up", "Down")
table(glm.preds_Weekly, Weekly$Direction)
```

```
##
## glm.preds_Weekly Down  Up
##             Down   54  48
##             Up    430 557
```

The fraction of correct predictions is

$$\frac{54+557}{1089} = 0.561 \ ,$$

which may seem like a bad performance, but in the context of stock market predictions, it is actually quite good. From the confusion matrix we also see that the classifier does a better job predicting when the market goes `Up`, but a rather poor job predicting the market goes `Down`.

d)

We start by dividing the `Weekly` data set into a train and a test set, where the training set consists of all observations in the period from 1990 to 2008, while the test set consists of the observations from the period 2009 to 2010. We then fit a logistic regression model to the training data set, make predictions for the test set, and calculate the confusion matrix.

```
Weekly_trainID = (Weekly$Year < 2009)
Weekly_train = Weekly[Weekly_trainID,]
Weekly_test = Weekly[!Weekly_trainID,]
glm.Weekly2 = glm(Direction~Lag2, family="binomial", data=Weekly_train)
glm.Weekly2_prob = predict(glm.Weekly2, newdata=Weekly_test, type="response")
glm.Weekly2_pred = ifelse(glm.Weekly2_prob > 0.5, "Up", "Down")
table(glm.Weekly2_pred, Weekly_test$Direction)
```

```
##
## glm.Weekly2_pred Down Up
##             Down    9  5
##             Up     34 56
```

The fraction of correct predictions on the test set is

$$\frac{9+56}{104} = 0.625.$$

e)

The `lda` function is available in the `MASS` library, thus we need to start by loading the library. We proceed by fitting a LDA model to the training set. We then use this model to make predictions for the test set and then compare the predicted and true classes using the confusion matrix.

```
library(MASS)
lda.Weekly = lda(Direction~Lag2, data=Weekly_train)
lda.Weekly_pred = predict(lda.Weekly, newdata=Weekly_test)$class
lda.Weekly_prob = predict(lda.Weekly, newdata=Weekly_test)$posterior
table(lda.Weekly_pred, Weekly_test$Direction)
```

```
##
## lda.Weekly_pred Down Up
##            Down    9  5
##            Up     34 56
```

The fraction of correct classifications is

$$\frac{9+56}{104} = 0.625.$$

f)

We follow the same procedure and test an QDA classifier on the data:

```
qda.Weekly = qda(Direction~Lag2, data=Weekly_train)
qda.Weekly_pred = predict(qda.Weekly, newdata=Weekly_test)$class
qda.Weekly_prob = predict(qda.Weekly, newdata=Weekly_test)$posterior
table(qda.Weekly_pred, Weekly_test$Direction)
```

```
##
## qda.Weekly_pred Down Up
##            Down    0  0
##            Up     43 61
```

The fraction of correct classifications is now

$$\frac{0+61}{104} = 0.587.$$

However, this QDA classifier is very naive and not really informative, because it just predicts an "Up" trend all the time. So it has a 100% error rate for the true "Down" trends, and it is thus a useless classifier.

g)

The KNN classifier is implemented in the `knn` function of the `class` library. This function requires some preparation of the data, as it does not accept a formula as a function argument. When there are ties (same amount of Up and Down for the nearest neighbors), the `knn` function picks a class at random. We use the `set.seed()` function such that we don't get different answers for each time we run the code.

```
library(class)
knn.train = as.matrix(Weekly_train$Lag2)
knn.test = as.matrix(Weekly_test$Lag2)
set.seed(123)
knn1.Weekly = knn(train = knn.train, test = knn.test, cl = Weekly_train$Direction, k=1, prob=T)
table(knn1.Weekly, Weekly_test$Direction)
```

```
##
## knn1.Weekly Down Up
##        Down   21 29
##        Up     22 32
```

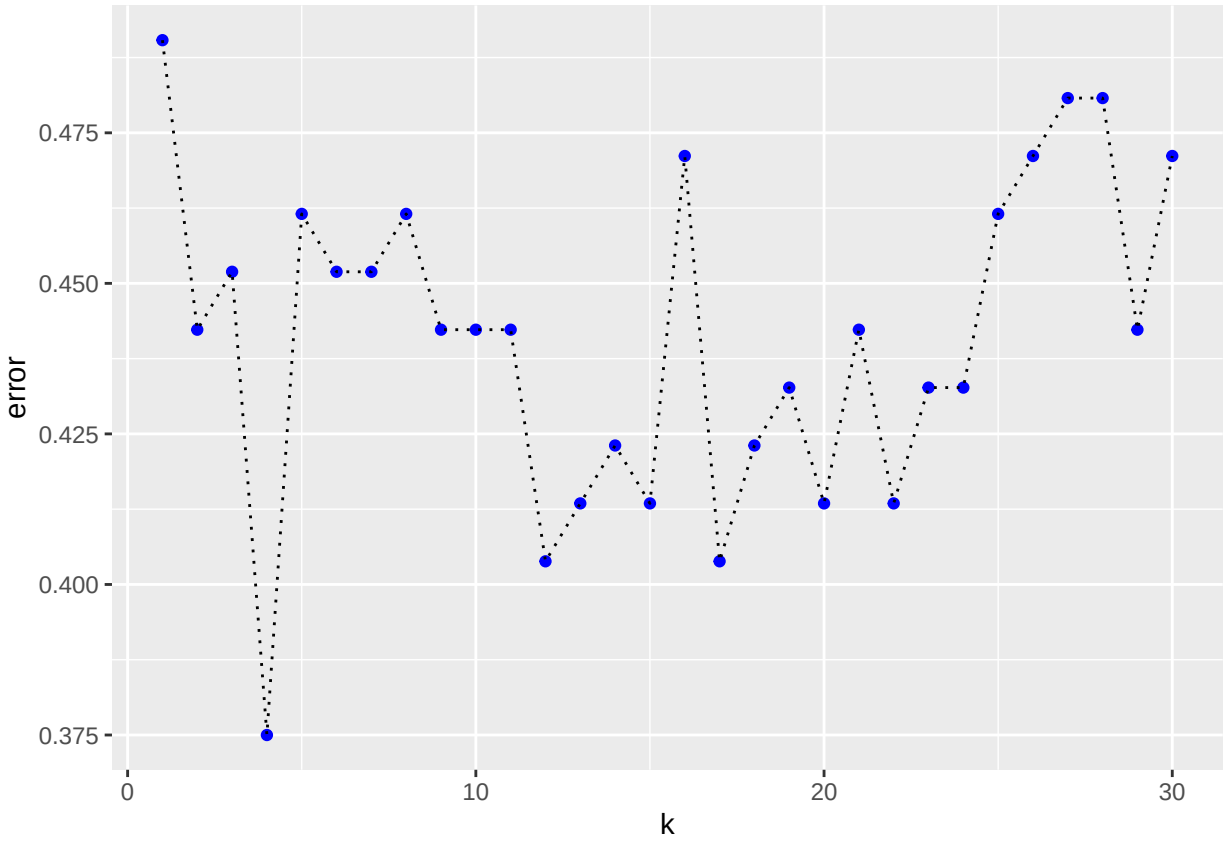
The fraction of correct classifications for the KNN classifier with $K = 1$ is

$$\frac{21+32}{104} = 0.5096$$

h)

To find the value of K giving the smallest error, we make a for-loop where we store the error for each iteration and plot the error against the value of K.

```
#knn error:
K=30
knn.error = rep(NA,K)
set.seed(321)
for(k in 1:K){
  knn.pred = knn(train = knn.train, test = knn.test, cl = Weekly_train$Direction, k=k)
  knn.error[k] = mean(knn.pred != Weekly_test$Direction)
}
knn.error.df = data.frame(k=1:K, error = knn.error)
ggplot(knn.error.df, aes(x=k, y=error))+geom_point(col="blue")+geom_line(linetype="dotted")
```



From the plot, we see that $K = 4$ gives the smallest error. We create a confusion matrix and store the probabilities of `Direction="Up"` for this model.

```
library(class)
set.seed(234)
knn4.Weekly = knn(train = knn.train, test = knn.test, cl = Weekly_train$Direction, k=4, prob=T)
table(knn4.Weekly, Weekly_test$Direction)
```

```
##
## knn4.Weekly Down Up
##        Down   19 24
##        Up     24 37
```

```
#get the probabilities for the classified class
knn4.Weekly_prob = attributes(knn4.Weekly)$prob
# since we want the probability for Up, we need to take 1-p for the elements that
#gives probability for Doum
down= which(knn4.Weekly == "Down")
knn4.Weekly_prob[down] = 1-knn4.Weekly_prob[down]
```

The fraction of correct classifications for the KNN classifier is

$$\frac{19+37}{104} = 0.538$$

i)

The logistic regression model and the LDA classifier provided the highest fractions of correct classifications on this data. Note that QDA does not classify any of the observations as "Up".

j)

To make ROC-curves and calculate AUC we use the packages `pROC` and `plotROC`.

```
#install.packages("plotROC")
#install.packages("pROC")
library(pROC)
library(plotROC)
glmroc = roc(response = Weekly_test$Direction, predictor = glm.Weekly2_prob, direction = "<")
ldaroc = roc(response = Weekly_test$Direction, predictor = lda.Weekly_prob[,2], direction = "<")
qdaroc = roc(response = Weekly_test$Direction, predictor = qda.Weekly_prob[,2], direction = "<")
knnroc = roc(response = Weekly_test$Direction, predictor = knn4.Weekly_prob, direction = "<")
auc(glmroc)
```

```
## Area under the curve: 0.5463
```

```
auc(ldaroc)
```

```
## Area under the curve: 0.5463
```

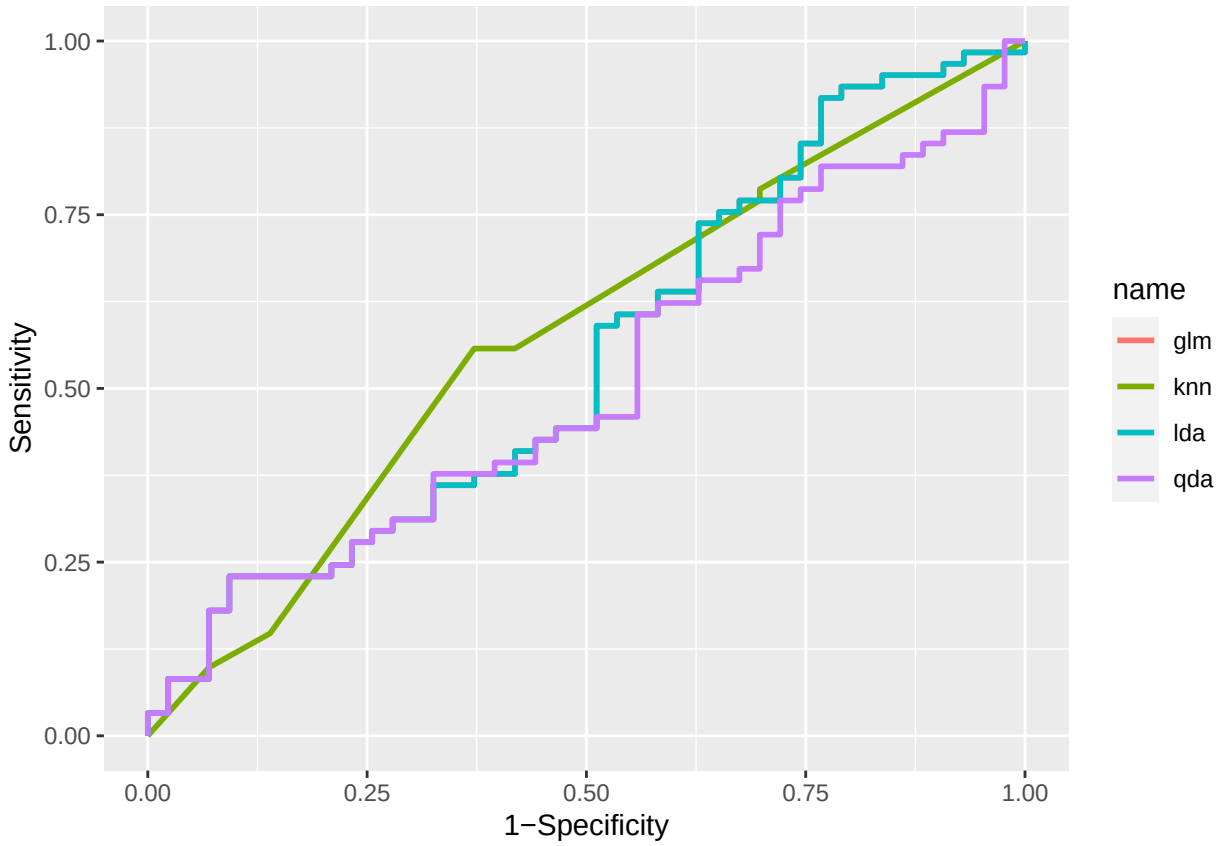
```
auc(qdaroc)
```

```
## Area under the curve: 0.5086
```

```
auc(knnroc)
```

```
## Area under the curve: 0.5753
```

```
dat = data.frame(Direction = Weekly_test$Direction, glm = glm.Weekly2_prob, lda = lda.Weekly_prob[,2],
                 qda = qda.Weekly_prob[,2], knn = knn4.Weekly_prob)
dat_long = melt_roc(dat, "Direction", c("glm", "lda", "qda", "knn"))
ggplot(dat_long, aes(d = D, m = M, color = name)) + geom_roc(n.cuts = F) +
  xlab("1-Specificity") + ylab("Sensitivity")
```




```
#glm is very similar to lda, so the roc-curve for glm is not shown.
```

Note first that the ROC-curve for glm is very similar to lda, so the glm-ROC-curve is not visible behind the lda curve. The ROC-curves with the corresponding AUCs tells us that none of our models perform well when classifying the test set. The ROC-curves are very close to the diagonal line, representing the "no information" classifier, thus the AUC values are very close to 0.5. A good classifier would hug the left upper corner in the ROC-plot and have a AUC value close to 1.