hw_02

lisa liubovich

2024-05-28

1. Ex. 2.4.7. KNN Prediction

The table below provides a training data set containing six observations, three predictors, and one qualitative response variable.

obs.	X_1	X_2	X_3	Y	Distance to X = (1,1,1)	Neighbor for k = 1	Neighbor for k = 3
1	0	3	0	Red	$\sqrt{6}$		
2	2	0	0	Red	$\sqrt{3}$		
3	0	1	3	Red	$\sqrt{5}$		
4	0	1	2	Green	$\sqrt{2}$		
5	-1	0	1	Green	$\sqrt{5}$		
6	1	1	1	Red	0		

Suppose we wish to use this data set to make a prediction for Y when $X_1 = X_2 = X_3 = 1$ using K-nearest neighbors. Do this manually (no code) and show your work.

(a) Compute the Euclidean distance between each observation and the test point, $X_1 = X_2 = X_3 = 1$.

for observation 1: square root of: $(0\text{-}1)^2 + (3\text{-}1)^2 + (0\text{-}1)^2 = \text{-}1^2 + 2^2 + 1^2 = 1 + 4 + 1 = 6 \rightarrow \sqrt{6}$

for observation 2: square root of: $(2\text{-}1)^2 + (0\text{-}1)^2 + (0\text{-}1)^2 = 1^2 + \text{-}1^2 + \text{-}1^2 = 1 + 1 + 1 = 3 \rightarrow \sqrt{3}$

for observation 3: square root of: $(0\text{-}1)^2 + (1\text{-}1)^2 + (3\text{-}1)^2 = \text{-}1^2 + 0^2 + 2^2 = 1 + 0 + 4 = 5 \rightarrow \sqrt{5}$

for observation 4: square root of: $(0\text{-}1)^2 + (1\text{-}1)^2 + (2\text{-}1)^2 = \text{-}1^2 + 0^2 + 1^2 = 1 + 0 + 1 = 2 \rightarrow \sqrt{2}$

for observation 5: square root of: $(\text{-}1\ \text{-}1)^2 + (0\text{-}1)^2 + (1\text{-}1)^2 = \text{-}2^2 + \text{-}1^2 + 0 = 4 + 1 + 1 = 5 \rightarrow \sqrt{5}$

for observation 5: square root of: $(1\text{-}1)^2 + (1\text{-}1)^2 + (1\text{-}1)^2 = 0 + 0 + 0 = 0$

(b) What is our prediction with $k = 1$? Why?

When k = 1, our prediction is Red. This is because the nearest neighbor to X1 = X2 = X3 =1 is observation 6, which has a distance of 0 between each observation and the test point, and the color of this observation is Red.

(c) What is our prediction with $k = 3$? Why?

When k = 3, our prediction is Red. This is because the three nearest neighbors to X1 = X2 = X3 = 1 are observations 2, 4, and 6, which are $\sqrt{3}$, $\sqrt{2}$, and 0 respectively. Their respective classifications are Red, Green, and Red. Red then wins by popular majority.

(d) If the decision boundary in this problem is highly nonlinear, would we expect the best value for k to be large or small? Why?

If the decision boundary is highly nonlinear, this indicates a flexible model which is associated with a small value of k.

In classification problems, decision boundaries separate different response categories in the X space of independent variables.

2 Ex. 4.8.13 (part) Stock Market Prediction, part 1. KNN

We want to predict the behavior of the stock market in the following week. The Weekly data set, (from the {ISLR2} package), contains 1,089 observations with the following 9 variables.

- Year: The year that the observation was recorded
- Lag1: Percentage return for previous week
- Lag2: Percentage return for 2 weeks previous
- Lag3: Percentage return for 3 weeks previous
- Lag4: Percentage return for 4 weeks previous
- Lag5: Percentage return for 5 weeks previous
- Volume: Volume of shares traded (average number of daily shares traded in billions)
- Today: Percentage return for this week
- Direction: A factor with levels Down and Up indicating whether the market had a positive or negative return on a given week

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

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(GGally)
```

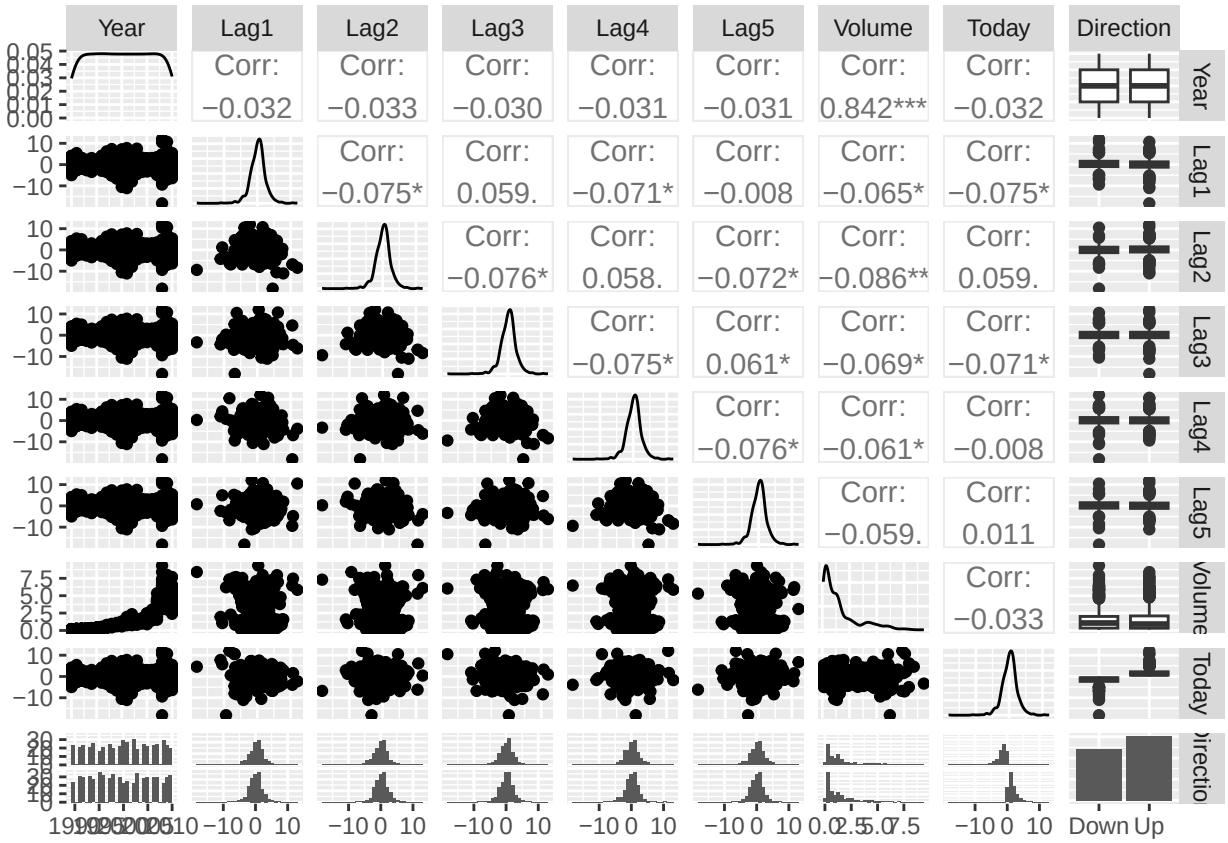
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
library(ggplot2)

weekly.data <- ISLR2::Weekly

summary(weekly.data)
```

```
##       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
##
##
##
##
```

```r
ggpairs(weekly.data)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Some observations of patterns:

- Volume and Year have a moderately high positive correlation (r = 0.842)
- The rest of the variables have very weak correlations with one another
- The earliest year is 1990 and the latest is 2010
- All lag means are around 0.14 or 0.15 ish, as is the mean of Today

(b) Split the data set into training and testing data and use the KNN method with $K = 9$ to predict Direction as the response based on the five lag variables plus Volume as predictors. Use split percentage of 70%. Why or why not is this a good split percentage? Use seed 1235.

```
library(class)
set.seed(1235)
training_pct <- 0.70
Z <- sample(nrow(weekly.data), floor(training_pct*nrow(weekly.data)))
weekly.training <- weekly.data[Z, ]
weekly.testing <- weekly.data[-Z, ]

X.train <- weekly.training[, 2:7]
Y.train <- weekly.training$Direction

X.test <- weekly.testing[, 2:7]
Y.test <- weekly.testing$Direction
```

```
weekly.knn <- knn(train=X.train, test=X.test, cl=Y.train, k = 9)
```

c. Compute the confusion matrix, showing cross-tabulation of the actual and predicted responses.

```
table(Y.test, weekly.knn)
```

```
##       weekly.knn
## Y.test Down  Up
##   Down   51  94
##   Up     67 115
```

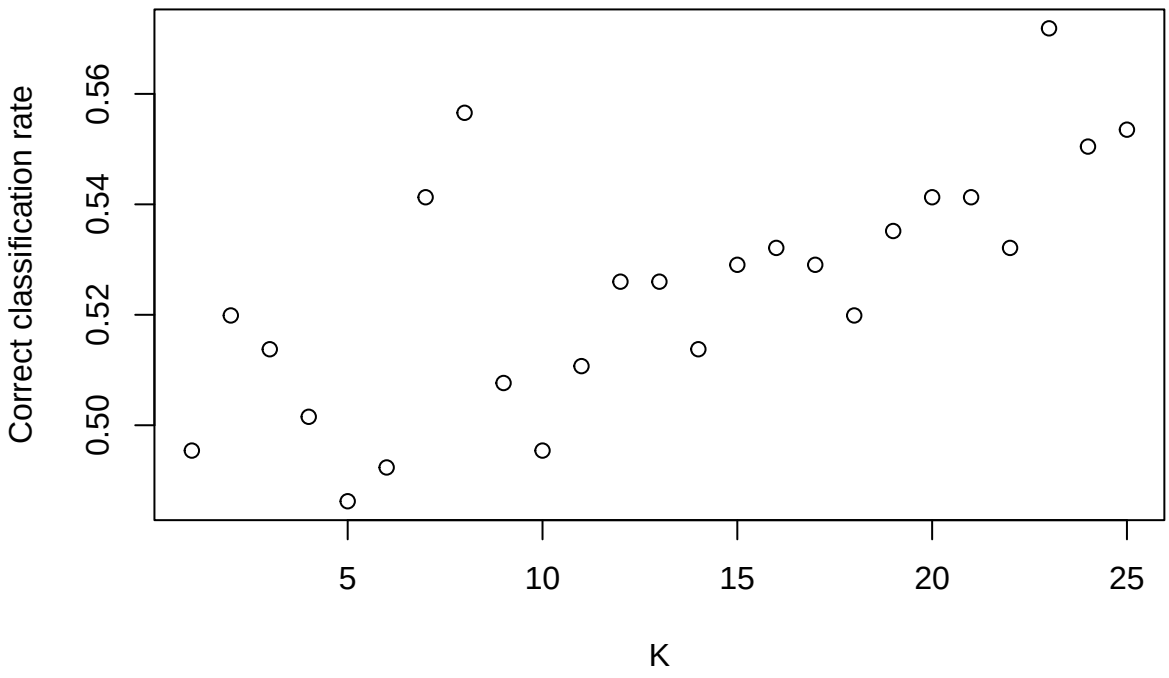
d. Compute the classification rate, which is the overall fraction of **correct** predictions.

```
mean(Y.test == weekly.knn)
```

```
## [1] 0.5076453
```

e. Tuning. Try different values of the tuning parameter K and select the optimal one; the one which minimizes the prediction *error* rate. Report the optimal K and the associated **error rate**.

```
Kmax <- 25
class.rate <- rep(0, Kmax)
for (i in 1:Kmax) {
  knn.out <- knn(train=X.train, test=X.test, cl=Y.train, k = i)
  class.rate[i] <- mean(Y.test == knn.out)
}

plot(c(1:Kmax), class.rate, xlab="K", ylab="Correct classification rate")
```




```
#optimal K
k.opt <- which.max(class.rate)
c(k.opt, class.rate[which.max(class.rate)])
```

```
## [1] 23.0000000  0.5718654
```

```
# error rate
weekly.knnOpt <- knn(train=X.train, test=X.test, cl=Y.train, k = k.opt)
mean(Y.test == weekly.knnOpt) # correct classification rate on the testing data.
```

```
## [1] 0.5718654
```

Optimal K: 23

Error rate: 1-0.5718654 = 0.4281346

3 Ex. 4.8.4. Curse of Dimensionality

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 non-parametric approaches often perform poorly when p is large. We will now investigate this curse.

(a) Suppose 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. 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, 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?

Range: 10% of 1 is 0.10

Fraction of available observations: $0.10/1 = 0.10$

On average, 10% of the available observations will be used to make the prediction.

(b) Now suppose 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] and are independent. We wish to predict a test observation's response using only observations that are within 10% of the range of X_1 **and** within 10% of the range of X_2 closest to that test observation. For instance, 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?

Range for X1: [X1 - 0.05, X1 + 0.05] $\rightarrow$ fraction within range: 0.05+ 0.05 = 0.10

Range for X2: [X2 - 0.05, X2 + 0.05] $\rightarrow$ fraction within range: 0.05 + 0.05 = 0.10

Fraction of available observations: 0.10 X 0.10 = 0.01

On average, 1% of the available observations will be used to make the prediction

(c) Now suppose we have a set of observations on $p = 100$ features. Again the observations are uniformly distributed on each feature, and 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?

Range for each feature: [Xi -0.10, Xi + 0.10] where i = 1, 2, . . . , 100

Fraction within range for each feature: 0.10 + 0.10 = 0.20

Fraction of available observations: $0.2^{100} = 1.268 \text{ x } 10^{-7}$

(d) Using your answers to parts (a) - (c), argue that a drawback of KNN when p is large is there are very few training observations "near" any given test observation.

(d) When p is large, the fraction of available observations used to make predictions becomes exceedingly small. In high-dimensional spaces, due to the curse of dimensionality, the available observations become sparse, meaning there are very few observations "near" any given test observation. KNN relies on finding the nearest neighbors to a test observation to make predictions. However, if there are very few training observations near any given test observation, KNN will have limited data to base its predictions on. This lack of nearby training observations can lead to poor performance of KNN in high-dimensional spaces because it becomes increasingly challenging to find relevant neighbors for making accurate predictions.

(e) 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 our variables that are i.i.d. Uniform (0, 1)). 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.*

Since the hypercube contains on average 10% of the training observations, the length of the side for a hypercube will be 10% of the total range of the variable. Since the range of the variables is 1-0 = 1, the length of the side for a hypercube is 0.10 in one dimension, two dimensions, and all 100 dimensions. While the length of each side of the hypercube remains constant (0.1 in this case), the actual volume of the hypercube grows exponentially with the dimensionality. This is because the total volume of the hypercube is the product of the lengths of its sides in each dimension.

4 Ex. 4.8.9. What are the Odds?

(a) On average, what fraction of people with an odds of 0.33 of defaulting on their credit card payment will in fact default?

   (a) Pr(default) = Odds/1+Odds = 0.33/1.33 = 0.2481203 = approx. 0.248 → the fraction of people with an odds of 0.33 of defaulting on their credit card who will in fact default is approximately 0.248 or 24.8%

(b) Suppose an individual has a 10% chance of defaulting on her credit card payment. What are the odds they will default?

   (a) Odds = probability/1 - probability = 0.10/1-0.10 = 0.10/0.90 = 0.11111 = approx. 0.111 → the odds that an individual with a 10% chance of defaulting will actually default is approximately 0.111

5 Ex. 4.8.6. What does it take to get an A?

Suppose we collect data on a group of students in a undergraduate statistics class using the following variables:

- X_1 : hours studied
- X_2 : undergrad GPA, and
- Y : receive an A (1) or Not an A (0).

We fit a logistic regression and produce estimated coefficients,

- β_0 = -6, β_1 = 0.05, and β_2 = 1.

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

   (a) log(P (Y=1)/(1-P(Y=1))= $\beta_0 + \beta_1 X_1 + \beta_2 X_2$ → logit(P(Y = 1)) = -6 + 0.05(40) + 1(3.5) → logit(P(Y=1)) = -6 +2 + 3.5 → logit(P(Y=1)) = -0.5
   (b) P(Y = 1 | X) = 1/1+ e^-(-0.5) = 1/1+1.648721 = 1/2.648721 = 0.3775407 = approx. 0.3775
   (c) The estimated probability that a student who studies for 40 hours and has an undergrad GPA of 3.5 is about 0.3775, or 37.75%

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

(a) 0.5 = 1/1+eˆ-linear predictor → 0.5 = 1/1+eˆ-2.5+0.0X1 → 0.5 + 0.5eˆ-2.5+0.5x = 1 → eˆ-2.5+0.05X1 =1 → log(eˆ-2.5+0.05X1) = log(1) → -2.5 +0.05X1 = 0 → 0.05X1 = 2.5 → X1 = 2.5/0.05 = 50

(b) The student in part a needs to study for 50 hours in order to have a 50% predicted chance of getting an A in the class

6 Ex. 4.8.13.(part) Stock Market Prediction, Part 2. Logistic Regression

We want to predict the behavior of the stock market in the following week. The Weekly data set, (from the {ISLR2} package), contains 1,089 observations with the following 9 variables.

- Year: The year that the observation was recorded
- Lag1: Percentage return for previous week
- Lag2: Percentage return for 2 weeks previous
- Lag3: Percentage return for 3 weeks previous
- Lag4: Percentage return for 4 weeks previous
- Lag5: Percentage return for 5 weeks previous
- Volume: Volume of shares traded (average number of daily shares traded in billions)
- Today: Percentage return for this week
- Direction: A factor with levels Down and Up indicating whether the market had a positive or negative return on a given week

(a) Perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Do any of the predictors appear to be statistically significant? If so, which ones?

```
weekly.data$direction_binary <- ifelse(weekly.data$Direction == "Up", 1, 0)
stock_glm <- glm(direction_binary ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, family = binomial, data =
summary(stock_glm)
```

```
##
## Call:
## glm(formula = direction_binary ~ Lag1 + Lag2 + Lag3 + Lag4 +
##     Lag5 + Volume, family = binomial, data = weekly.data)
##
## 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 appears to be statistically significant at a 5% level, with a p-value of 0.0296.

b. 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 the logistic regression model.

```
cutoff <- 0.5
predicted.class <- 1*(stock_glm$fitted.values > cutoff)

# Confusion matrix
table(weekly.data$direction_binary, predicted.class)
```

```
##    predicted.class
##       0   1
##   0  54 430
##   1  48 557
```

```
# Correct classification rate
mean(weekly.data$direction_binary == predicted.class)
```

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

the confusion matrix tells us the amount of mis-classifications that the logistic regression model makes compared to the number of observations that were correctly classified.

c. Fit a logistic regression model with Lag2 as the only predictor using as training data the period from 1990 to 2008. Compute the confusion matrix and the overall fraction of correct predictions for the held out test data (that is the data from 2009, and 2010). How does it compare to before? What does that suggest?

```
train_data <- weekly.data %>%
  filter(Year >= 1990 & Year <= 2008)
test_data <- weekly.data %>%
  filter(Year >= 2009 & Year <= 2010)

logit_model <- glm(direction_binary ~ Lag2, data = train_data, family = "binomial")

predicted.probs <- predict(logit_model, newdata = test_data, type = "response")

predicted.class <- ifelse(predicted.probs > cutoff, 1, 0)

# Confusion matrix
table(test_data$direction_binary, predicted.class)
```

```
##    predicted.class
##      0  1
##   0  9 34
##   1  5 56
```

```r
mean(test_data$direction_binary == predicted.class)
```

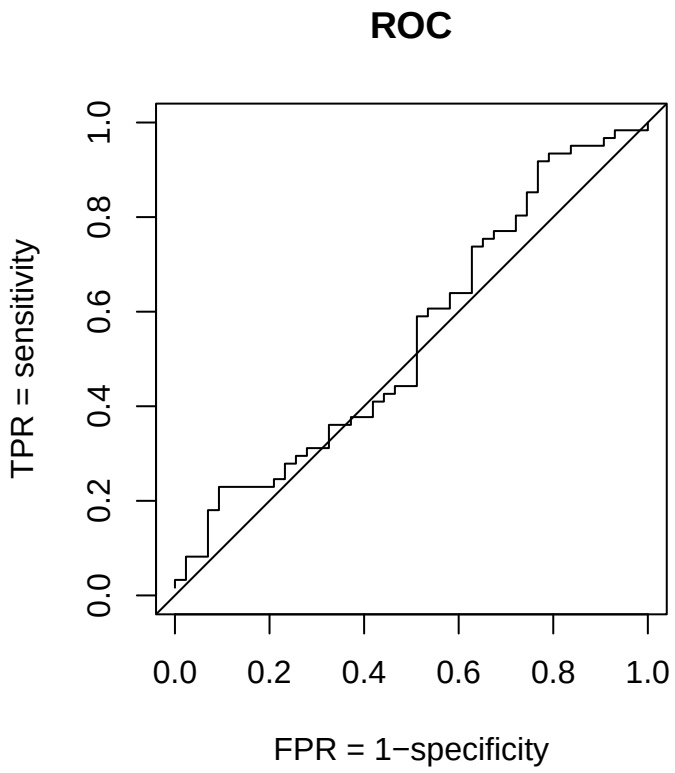
```
## [1] 0.625
```

This particular fraction of correct predictions is higher than before, telling us that Lag2 is a strong predictor of direction.

d. Plot an ROC curve for the logistic regression on the test data from (c), using different probability thresholds and add an diagonal line for FPR = TPR. Interpret the plot in terms of how useful the model might be?

```r
roc.analysis <-function (object, newdata = NULL, newplot=TRUE)
{
  if (is.null(newdata)) {
    pi.tp <- object$fitted[object$y == 1]
    pi.tn <- object$fitted[object$y == 0]
  }
  else {
    pi.tp <- predict(object, newdata, type = "response")[newdata$y == 1]
    pi.tn <- predict(object, newdata, type = "response")[newdata$y == 0]
  }

  pi.all <- sort(c(pi.tp, pi.tn))
  sens <- rep(1, length(pi.all)+1)
  specc <- rep(1, length(pi.all)+1)
  for (i in 1:length(pi.all)) {
    sens[i+1] <- mean(pi.tp >= pi.all[i], na.rm = T)
    specc[i+1] <- mean(pi.tn >= pi.all[i], na.rm = T)
  }

  npoints <- length(sens)
  area <- sum(0.5 * (sens[-1] + sens[-npoints]) * (specc[-npoints] -
        specc[-1]))

  lift <- (sens - specc)[-1]
  cutoff <- pi.all[lift == max(lift)][1]
  sensopt <- sens[-1][lift == max(lift)][1]
  specopt <- 1 - specc[-1][lift == max(lift)][1]

  par(pty="s")
  if (newplot){
  plot(specc, sens, xlim = c(0, 1), ylim = c(0, 1), type = "s",
            xlab = "FPR = 1-specificity", ylab = "TPR = sensitivity", main="ROC")
  abline(0, 1)
  }
  else lines(specc, sens, type="s", lty=2, col=2)

  list(pihat=as.vector(pi.all), sens=as.vector(sens[-1]),
```

```
  spec=as.vector(1-specc[-1]), area = area, cutoff = cutoff,
  sensopt = sensopt, specopt = specopt)
}

test_data$y <- 1*(test_data$direction_binary == 1)
test.ROC <- roc.analysis(logit_model, newdata = test_data)
```



The ROC curve is close to the diagonal line, the classifier's performance is near random. That is, it is not doing a good job between the positive and negative classes. Some of the parts of the curve that are above the diagonal line suggest the classifer is doing better than random guessing, but overall the curve is close to the line.

7 Ex. 4.8.5, LDA v 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?

  (a) For the training set, QDA is likely to perform better because of its flexibility and ability to capture noise better than the LDA. For the testing set, the LDA is likely to perform better because its decision boundary is a better approximation of a Bayes linear decision boundary.

(b) Compare the expected performance of LDA and QDA on the training set and then on the test set if the Bayes decision boundary is *non-linear*.

(a) If the Bayes decision boundary is non-linear, the QDA will perform better on both sets because it better approximates a non-linear Bayes decision boundary than LDA. However, QDA's performance could be impacted by over fitting.

(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?

(a) In general, as the sample size n increases, we expect the test prediction accuracy of QDA relative to LDA to improve because QDA's flexibility allows it to estimate parameters more accurately (due to the lower variability) and capture more complex decision boundaries with the increased data.

(d) **True or False**: 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.**

(a) False: The added flexibility of the QDA can lead to overfitting while the LDA's assumption of a common covariance matrix offers more robust estimation with fewer parameters. Also, the bias-variance tradeoff should be considered; While QDA may have lower bias than LDA, its higher variance could lead to poorer performance, especially if the true Bayes decision boundary is linear.

8 Ex. 4.8.7. Non-uniform Prior. Predicting Issuance of a Stock Dividend

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:

- The mean value of percent profit X for companies that issued a dividend was $\bar{x}_1 = 10$, while the mean for those that didn't was $\bar{x}_2 = 0$.
- The variance of X for these two sets of companies was $\sigma^2 = 36$.
- 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.

```
mean_dividend <- 10
mean_no_dividend <- 0
variance <- 36
x <- 4

pdf_dividend <- dnorm(x, mean_dividend, sqrt(variance))
pdf_no_dividend <- dnorm(x, mean_no_dividend, sqrt(variance))
prob_dividend <- 0.80
pdf_dividend
```

```
## [1] 0.04032845
```

```
prob_x_4 <- prob_dividend * pdf_dividend + (1 - prob_dividend) * pdf_no_dividend
prob_x_4
```

```
## [1] 0.04291103
```

P(Dividend | X = 4) = (P(X = 4 | Dividend) * P(Dividend))/P(X = 4) = (0.04032845 * 0.80)/0.04291103 = 0.7518524

The predicted probability that a company will issue a dividend when its percentage profit that year was 4% is approximately 0.752 or 75.2%

Hint: Recall that the density function for a Normal random variable is f(x) = $1/\ \sigma\sqrt{2}\pi\ e^{-\frac{(x-\mu)^2}{2\sigma^2}}$. You will need to use the Bayes theorem. Also, use R function dnorm(x, mean, sd) to compute the pdf ($f(x)$) of Normal distribution.

We can also use the probability calculator in R.

9 Ex. 4.8.13.(part) Stock Market Prediction, Part 3. LDA, QDA and Summary

We want to continue with trying to predict the behavior of the stock market in the following week. We computed a KNN estimate and a Logistic regression estimate already. We will now look at LDA and QDA using the {MASS} package and compare across these prediction methods.

The Weekly data set, (from the {ISLR2} package), contains 1,089 observations with the following 9 variables.

- Year: The year that the observation was recorded
- Lag1: Percentage return for previous week
- Lag2: Percentage return for 2 weeks previous
- Lag3: Percentage return for 3 weeks previous
- Lag4: Percentage return for 4 weeks previous
- Lag5: Percentage return for 5 weeks previous
- Volume: Volume of shares traded (average number of daily shares traded in billions)
- Today: Percentage return for this week
- Direction: A factor with levels Down and Up indicating whether the market had a positive or negative return on a given week

(a) Use LDA with 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).

```
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
lda_model <- lda(Direction ~ Lag2, data = train_data)

predictions <- predict(lda_model, newdata = test_data)$class

# Compute the confusion matrix
table(predictions, test_data$Direction)
```

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

```
mean(predictions == test_data$Direction)
```

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

(b) Repeat (a) using QDA.

```
library(MASS)
qda_model <- qda(Direction ~ Lag2, data = train_data)

predictions <- predict(qda_model, newdata = test_data)$class

# Compute the confusion matrix
table(predictions, test_data$Direction)
```

```
##
## predictions Down Up
##        Down    0  0
##        Up     43 61
```

```
mean(predictions == test_data$Direction)
```

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

(c) Using the results from the previous questions (Stock Market Prediction. Part 1, 2), compare the correct classification rates on the testing data obtained from the 4 methods: KNN, Logistic regression, LDA and QDA. Recommend one or more methods. Explain your rationale.

KNN correct classification rate: 0.5718654

Logistic regression correct classification rate: 0.625

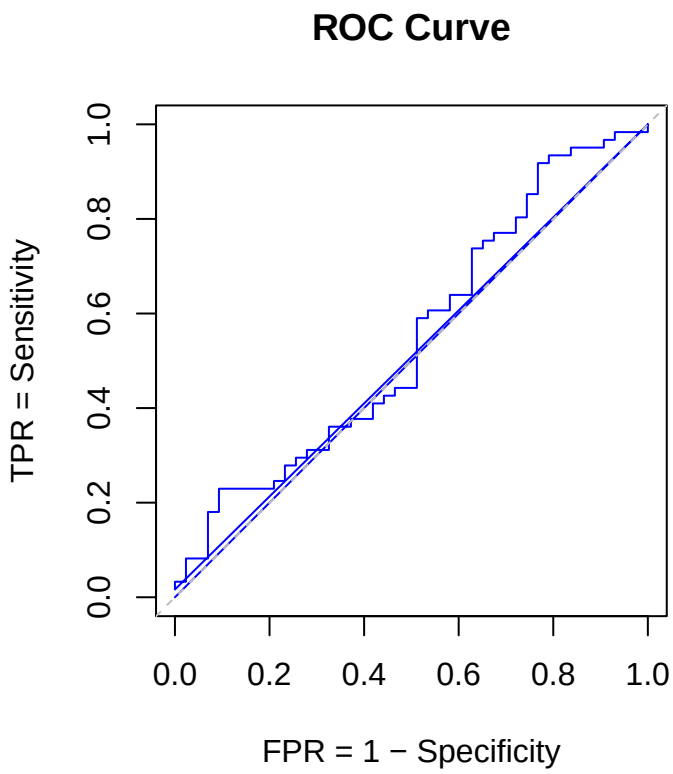
LDA correct classification rate: 0.625

QDA correct classification rate: 0.5865385

While the logistic regression and LDA have the same correct classification rate, the logistic regression didn't perform very well on the ROC curve.

```
roc.analysis <- function(object, newdata = NULL, newplot = TRUE) {
  if (is.null(newdata)) {
    pred <- object$posterior
    pi.tp <- pred[object$Y == 1, 2]
    pi.tn <- pred[object$Y == 0, 2]
  } else {
    pred <- predict(object, newdata)
    pi.tp <- pred$posterior[newdata$Y == 1, 2]
    pi.tn <- pred$posterior[newdata$Y == 0, 2]
  }
```

```
  pi.tp <- unlist(pi.tp)
  pi.tn <- unlist(pi.tn)

  # Combine and sort unique thresholds
  pi.all <- sort(unique(c(pi.tp, pi.tn)))

  tpr <- numeric(length(pi.all) + 1)
  fpr <- numeric(length(pi.all) + 1)

  # Initialize with (0, 0)
  tpr[1] <- 0
  fpr[1] <- 0

  for (i in 1:length(pi.all)) {
    tpr[i + 1] <- mean(pi.tp >= pi.all[i], na.rm = TRUE)
    fpr[i + 1] <- mean(pi.tn >= pi.all[i], na.rm = TRUE)
  }

  # Add (1, 1) to the end
  tpr <- c(tpr, 1)
  fpr <- c(fpr, 1)

  # Calculate the area under the curve (AUC)
  auc <- sum(diff(fpr) * (tpr[-length(tpr)] + tpr[-1])) / 2

  # Plot ROC Curve
  par(pty = "s")
  if (newplot) {
    plot(fpr, tpr, type = "l", col = "blue", xlim = c(0, 1), ylim = c(0, 1),
         xlab = "FPR = 1 - Specificity", ylab = "TPR = Sensitivity", main = "ROC Curve")
    abline(a = 0, b = 1, col = "gray", lty = 2)
  } else {
    lines(fpr, tpr, type = "l", col = "blue")
  }

  # Return the ROC metrics
  list(pihat = as.vector(pi.all), tpr = as.vector(tpr),
       fpr = as.vector(fpr), area = auc)
}

# Example usage (assuming lda_model and test_data are correctly specified):
test_data$Y <- as.integer(test_data$direction_binary == 1)
test.ROC <- roc.analysis(lda_model, newdata = test_data)
```



This ROC curve is not much better than the logistic regression one. So in this scenario I would recommend either logistic regression and LDA. I don't know anymore. I lost my sanity 3 hours ago. This HW was very hard. Please go easy on me.