DSE6111 - Module 3 - Ch 4 HW

Nathan Monges

2024-07-21

```r
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   3.5.0     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 error
```

```r
library(ISLR2)
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:ISLR2':
##
##     Boston
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(e1071)
library(class)
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 1.2.0 --
## v broom        1.0.5      v rsample      1.2.1
## v dials        1.2.1      v tune         1.2.1
## v infer        1.0.7      v workflows    1.1.4
## v modeldata    1.4.0      v workflowsets 1.1.0
## v parsnip      1.2.1      v yardstick    1.3.1
## v recipes      1.0.10
```

```
## Warning: package 'modeldata' was built under R version 4.3.3
```

```
## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x scales::discard()       masks purrr::discard()
## x dplyr::filter()         masks stats::filter()
## x recipes::fixed()        masks stringr::fixed()
## x dplyr::lag()            masks stats::lag()
## x rsample::permutations() masks e1071::permutations()
## x MASS::select()          masks dplyr::select()
## x yardstick::spec()       masks readr::spec()
## x recipes::step()         masks stats::step()
## x tune::tune()            masks parsnip::tune(), e1071::tune()
## * Use tidymodels_prefer() to resolve common conflicts.
```

```r
attach(Weekly)
```

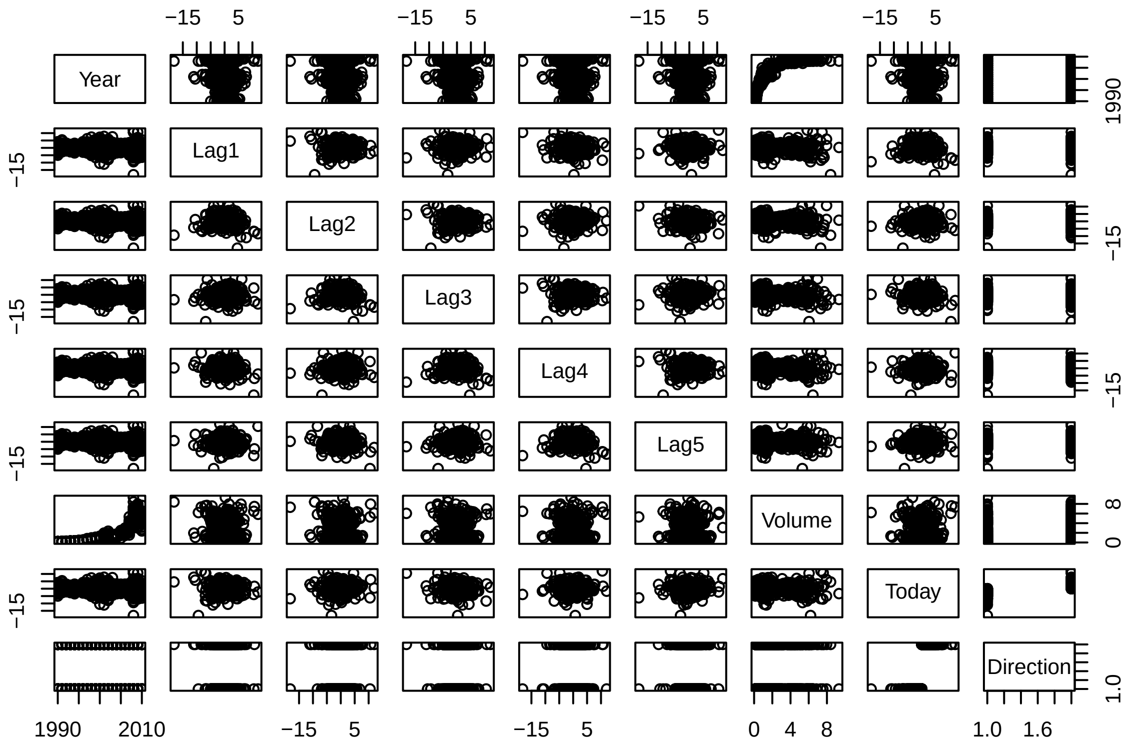
Exercise 13

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

```r
cor(Weekly[,-9])
```

```
##               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
```

```r
plot(Weekly)
```



Based off of the correlations matrix above it seems that volume and year are significantly related with a 0.84 correlation which makes sense as years increase more people are entering the markets. It also appears that there is little correlation between today's returns and previous days returns as all of the lag variables correaltion are negative or close to zero.

(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?

```r
weekly.logistic <- glm(
  Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
  data = Weekly, family = binomial
  )
summary(weekly.logistic)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Weekly)
##
## 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
```

From the logistic regression using Direction as the response it appears that for most the predictor variables, there is no clear evidence of a real association with Directionn, since these varibales have such high p-values. The only predictor variable with a positve intercept and relatively low p-value compared to the other predictors is Lag2. Which signifies the baseline probability of the stock market moving up is greater than 50%.

(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.

```
weekly.probs <- predict(weekly.logistic, type = "response")
#weekly.probs[1:10]
#contrasts(Direction)

weekly.pred <- rep("Down", 1089)
weekly.pred[weekly.probs > .5] = "Up"

table(weekly.pred, Direction)
```

```
##            Direction
## weekly.pred Down  Up
##        Down   54  48
##        Up    430 557
```

```
mean(weekly.pred == Direction) #logistic regression correctly predicted the movement of the market 52.2
```

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

Based on the confusion matrix from the predictions made my the logistic regression model, it seems that the model works fairly well in predicting whether the market will be up or down. The model correctly predicted that the market will go up on 557 days and that it would go down on 54 days, for a total of 611 correct predictions. The mean function helps in computing the overall fraction of correct predictions and shows that the model correctly predicted the movement of the market 56.11% of the time.

(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).

```
train <- (Year < 2009)

weekly.0910 <- Weekly[!train, ]
Direction.0910 <- Direction[!train]



weekly.train <- glm(
  Direction ~ Lag2, data = Weekly, family = binomial, subset = train)

weekly.test <- predict(weekly.train, weekly.0910, type = "response")
weekly.test <- rep("Down", 104)
weekly.test[weekly.test > .5] <- "Up"
table(weekly.test, Direction.0910)
```

```
##            Direction.0910
## weekly.test Down Up
##          Up   43 61
```

```
mean(weekly.test == Direction.0910)
```

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

(e) Repeat (d) using LDA.

```
weekly.lda <- lda(Direction ~ Lag2, data = Weekly,
               subset = train)

lda.pred <- predict(weekly.lda, weekly.0910)
lda.class <- lda.pred$class
table(lda.class, Direction.0910)
```

```
##          Direction.0910
## lda.class Down Up
##      Down    9  5
##      Up     34 56
```

```
mean(lda.class == Direction.0910)
```

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

(f) Repeat (d) using QDA.

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

qda.pred <- predict(weekly.qda, weekly.0910)
qda.class <- qda.pred$class
table(qda.class, Direction.0910)
```

```
##          Direction.0910
## qda.class Down Up
##      Down    0  0
##      Up     43 61
```

```
mean(qda.class == Direction.0910)
```

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

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

```
train.x <- Weekly[Year < 2009, ]
train.x2 <- train.x$Lag2

test.x <- Weekly[Year >= 2009, ]
test.x2 <- test.x$Lag2

train_x_matrix <- as.matrix(train.x2)
test_x_matrix <- as.matrix(test.x2)

train.direction <- Direction[train]


set.seed(1)
weekly.knn <- knn(train_x_matrix , test_x_matrix, train.direction, k =1)
table(weekly.knn, Direction.0910)
```

```
##           Direction.0910
## weekly.knn Down Up
##       Down   21 30
##       Up     22 31
```

```r
mean(weekly.knn == Direction.0910)
```

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

(h) Repeat (d) using naive Bayes.

```r
weekly.nb <- naiveBayes(Direction ~ Lag2, data = Weekly, subset = train)


nb.class <- predict(weekly.nb, weekly.0910)
table(nb.class, Direction.0910)
```

```
##         Direction.0910
## nb.class Down Up
##     Down    0  0
##     Up     43 61
```

```r
mean(nb.class == Direction.0910)
```

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

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

After all of the regression models fitted to the Weekly data, it is evident that Linear Discriminant Analysis was the best at predicting whether the market will go up or down with as it did so correctly at a rate of 62.5%. All of the other models were still able to predict correctly at a significant rate of at least >50%. KNN performed the most poorly with the lowest correct prediction rate of 50%.

(j) Experiment with different combinations of predictors, includ- ing possible transformations and interactions, for each of the methods. Report the variables, method, and associated confu- sion 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.

```r
#LDA
weekly.lda2 <- lda(Direction ~ Lag1 + Lag4, data = Weekly,
               subset = train)

lda.pred2 <- predict(weekly.lda2, weekly.0910)
lda.class2 <- lda.pred2$class
table(lda.class2, Direction.0910)
```

```
##           Direction.0910
## lda.class2 Down Up
##       Down    6  6
##       Up     37 55
```

```r
mean(lda.class2 == Direction.0910)
```

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

Linear Discriminant Analysis givees the best results at predicting the market status based on the 4th last and last days market status with a correct prediciton rate of 58.65%.

```r
train_x <- cbind(Lag1, Lag4)[train, ]
test_x <- cbind(Lag1, Lag4)[!train, ]
train_direction <- Direction[train]


set.seed(1)
weekly_knn <- knn(train_x, test_x, train_direction, k =3)
table(weekly_knn, Direction.0910)
```

```
##           Direction.0910
## weekly_knn Down Up
##       Down   19 31
##       Up     24 30
```

```r
mean(weekly_knn == Direction.0910)
```

```
## [1] 0.4711538
```

```r
set.seed(1)
weekly_knn <- knn(train_x, test_x, train_direction, k =5)
table(weekly_knn, Direction.0910)
```

```
##           Direction.0910
## weekly_knn Down Up
##       Down   20 29
##       Up     23 32
```

```r
mean(weekly_knn == Direction.0910)
```

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

KNN gives disappointing results in predictig the market status based on the histroic last and 4th last day price status (lag1, lag2) with just a 47.15% correct prediction rate at K = 3. But these results to get slightly better at a 50% correct prediction rate at K = 5.

```r
weekly.logistic2 <- glm(
  Direction ~ Lag1 + Lag4, data = Weekly, family = binomial)


weekly.probs2 <- predict(weekly.logistic2, type = "response")
#weekly.probs[1:10]
#contrasts(Direction)

weekly.pred2 <- rep("Down", 1089)
weekly.pred2[weekly.probs2 > .5] = "Up"

table(weekly.pred2, Direction)
```

```
##             Direction
## weekly.pred2 Down  Up
##         Down   13  11
##         Up    471 594
```

```r
mean(weekly.pred2 == Direction) #logistic regression correctly predicted the movement of the market 52.
```

```
## [1] 0.5573921
```

Logistc regression gives realtively good results when compared to the previous 2 methods used with a 55.74%

correct prediction rate.

Exercise 16

Using the Boston data set, fit classification models in order to predict whether a given census tract has a crime rate above or below the median. Explore logistic regression, LDA, naive Bayes, and KNN models using various subsets of the predictors. Describe your findings. Hint: You will have to create the response variable yourself, using the variables that are contained in the Boston data set.

```
attach(Boston)
median_crim <- median(Boston$crim)

Boston$high_crim <- ifelse(Boston$crim > median_crim, 1, 0)

cor(Boston)
```

```
##                  crim          zn       indus         chas         nox
## crim       1.00000000 -0.20046922  0.40658341 -0.055891582  0.42097171
## zn        -0.20046922  1.00000000 -0.53382819 -0.042696719 -0.51660371
## indus      0.40658341 -0.53382819  1.00000000  0.062938027  0.76365145
## chas      -0.05589158 -0.04269672  0.06293803  1.000000000  0.09120281
## nox        0.42097171 -0.51660371  0.76365145  0.091202807  1.00000000
## rm        -0.21924670  0.31199059 -0.39167585  0.091251225 -0.30218819
## age        0.35273425 -0.56953734  0.64477851  0.086517774  0.73147010
## dis       -0.37967009  0.66440822 -0.70802699 -0.099175780 -0.76923011
## rad        0.62550515 -0.31194783  0.59512927 -0.007368241  0.61144056
## tax        0.58276431 -0.31456332  0.72076018 -0.035586518  0.66802320
## ptratio    0.28994558 -0.39167855  0.38324756 -0.121515174  0.18893268
## black     -0.38506394  0.17552032 -0.35697654  0.048788485 -0.38005064
## lstat      0.45562148 -0.41299457  0.60379972 -0.053929298  0.59087892
## medv      -0.38830461  0.36044534 -0.48372516  0.175260177 -0.42732077
## high_crim  0.40939545 -0.43615103  0.60326017  0.070096774  0.72323480
##                    rm         age         dis          rad         tax
## crim      -0.21924670  0.35273425 -0.37967009  0.625505145  0.58276431
## zn         0.31199059 -0.56953734  0.66440822 -0.311947826 -0.31456332
## indus     -0.39167585  0.64477851 -0.70802699  0.595129275  0.72076018
## chas       0.09125123  0.08651777 -0.09917578 -0.007368241 -0.03558652
## nox       -0.30218819  0.73147010 -0.76923011  0.611440563  0.66802320
## rm         1.00000000 -0.24026493  0.20524621 -0.209846668 -0.29204783
## age       -0.24026493  1.00000000 -0.74788054  0.456022452  0.50645559
## dis        0.20524621 -0.74788054  1.00000000 -0.494587930 -0.53443158
## rad       -0.20984667  0.45602245 -0.49458793  1.000000000  0.91022819
## tax       -0.29204783  0.50645559 -0.53443158  0.910228189  1.00000000
## ptratio   -0.35550149  0.26151501 -0.23247054  0.464741179  0.46085304
## black      0.12806864 -0.27353398  0.29151167 -0.444412816 -0.44180801
## lstat     -0.61380827  0.60233853 -0.49699583  0.488676335  0.54399341
## medv       0.69535995 -0.37695457  0.24992873 -0.381626231 -0.46853593
## high_crim -0.15637178  0.61393992 -0.61634164  0.619786249  0.60874128
##              ptratio       black      lstat       medv   high_crim
## crim       0.2899456 -0.38506394  0.4556215 -0.3883046  0.40939545
## zn        -0.3916785  0.17552032 -0.4129946  0.3604453 -0.43615103
## indus      0.3832476 -0.35697654  0.6037997 -0.4837252  0.60326017
## chas      -0.1215152  0.04878848 -0.0539293  0.1752602  0.07009677
## nox        0.1889327 -0.38005064  0.5908789 -0.4273208  0.72323480
## rm        -0.3555015  0.12806864 -0.6138083  0.6953599 -0.15637178
## age        0.2615150 -0.27353398  0.6023385 -0.3769546  0.61393992
## dis       -0.2324705  0.29151167 -0.4969958  0.2499287 -0.61634164
## rad        0.4647412 -0.44441282  0.4886763 -0.3816262  0.61978625
## tax        0.4608530 -0.44180801  0.5439934 -0.4685359  0.60874128
## ptratio    1.0000000 -0.17738330  0.3740443 -0.5077867  0.25356836
## black     -0.1773833  1.00000000 -0.3660869  0.3334608 -0.35121093
## lstat      0.3740443 -0.36608690  1.0000000 -0.7376627  0.45326273
## medv      -0.5077867  0.33346082 -0.7376627  1.0000000 -0.26301673
## high_crim  0.2535684 -0.35121093  0.4532627 -0.2630167  1.00000000
```

After computing the correlations of the variables in the Boston data-set, it is clear that the indus, age, rad, and tax are all significantly related to high crime as these variables all share the trait of having a correlation > .60. These are the variables that will be used in the regression of the new high_crim variable to predict if a specific Boston area has high crime (based on the claim that high crime = a value > median crim rate) or does not.

Logistic Regression

```r
data_split <-initial_split(Boston, strata = "crim", prop = 0.7)
training_set <- training(data_split)
test_set<- testing(data_split)
crim_test <- test_set$high_crim

boston_log <- glm(high_crim ~ indus + age + rad + tax, data = training_set, family = binomial)

crim_prob <- predict(boston_log, test_set, type = "response")
crim_pred <- ifelse(crim_prob > .5, 1, 0)
table(crim_pred, crim_test)
```

```
##          crim_test
## crim_pred  0  1
##         0 65 16
##         1 12 61
```

```r
mean(crim_pred == crim_test)
```

```
## [1] 0.8181818
```

Here it is evident that logistic regression worked very well with the predictors used from the dataset. This model was able to get predict whether an area is an area wiht high crime 81.81% of the time.

Linear Discriminant Analysis

```r
boston_lda <- lda(high_crim ~ indus + age + rad + tax, data = training_set, family = binomial)
crim_pred_lda <- predict(boston_lda, test_set, type = "response")
crim_lda_class <- crim_pred_lda$class
table(crim_lda_class, crim_test)
```

```
##               crim_test
## crim_lda_class  0  1
##              0 68 20
##              1  9 57
```

```r
mean(crim_lda_class == crim_test)
```

```
## [1] 0.8116883
```

Linear discriminant analysis also performed well in its predictions but still underperfomed logistic regression slightly with a 80.52% correct prediction rate.

Naive Bayes

```
boston_nb <- naiveBayes(high_crim ~ indus + age + rad + tax, data = training_set)
boston_nb_class <- predict(boston_nb, test_set)
table(boston_nb_class, crim_test)
```

```
##                crim_test
## boston_nb_class  0  1
##               0 73 25
##               1  4 52
```

```
mean(boston_nb_class == crim_test)
```

```
## [1] 0.8116883
```

Naive Bayes also performs very well on the test set with a correct prediction rate of 81.16%. Although very good, it is still underperforming compared to logistic regression but outperforming when compared to LDA.

K-Nearest Neighbors

```
crim_train <- training_set$high_crim

set.seed(1)
boston.knn <- knn(training_set, test_set, crim_train, k = 1)
table(boston.knn, crim_test)
```

```
##           crim_test
## boston.knn  0  1
##          0 69  7
##          1  8 70
```

```
mean(boston.knn == crim_test)
```

```
## [1] 0.9025974
```

K-nearest neighbors is clearly the best model out of all of the ones used to predict if a Boston area has a high crime rate or not. This model is was able to correctly predict at a rate of 94.15% which greatly out performs our previous best model of logistic regression.