Machine Learning Homework 1

Shikun (Jason) Wang

May 29th, 2017

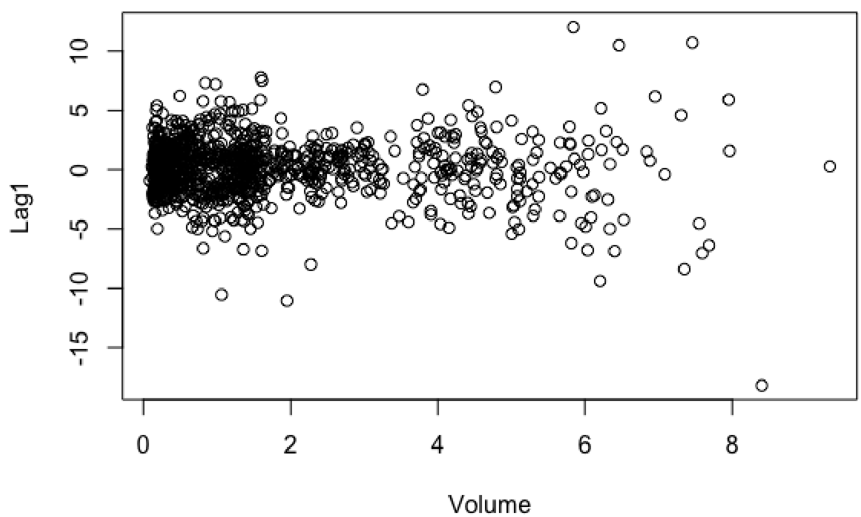
1 Problem1

1.1 a

I used the summary () function to show the numerical analysis of the dataset Weekly

```
> library(ISLR)
> library(MASS)
> library(class)
> summary(Weekly)
      Year           Lag1               Lag2
 Min.   :1990   Min.   :-18.1950   Min.   :-18.1950
 1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540
 Median :2000   Median :  0.2410   Median :  0.2410
 Mean   :2000   Mean   :  0.1506   Mean   :  0.1511
 3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090
 Max.   :2010   Max.   : 12.0260   Max.   : 12.0260
      Lag3               Lag4               Lag5
 Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
 1st Qu.: -1.1580   1st Qu.: -1.1580   1st Qu.: -1.1660
 Median :  0.2410   Median :  0.2380   Median :  0.2340
 Mean   :  0.1472   Mean   :  0.1458   Mean   :  0.1399
 3rd Qu.:  1.4090   3rd Qu.:  1.4090   3rd Qu.:  1.4050
 Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
     Volume            Today            Direction
 Min.   :0.08747   Min.   :-18.1950   Down:484
 1st Qu.:0.33202   1st Qu.: -1.1540   Up  :605
 Median :1.00268   Median :  0.2410
 Mean   :1.57462   Mean   :  0.1499
 3rd Qu.:2.05373   3rd Qu.:  1.4050
 Max.   :9.32821   Max.   : 12.0260
```

As you can see all the Lag1 to Lag5 have the similar numerical values (median, mean), which means that the return percentage has not correlation with time. The scatter plot between Year and Lag also shows this conclusion:



And as I performed the calculation of correlation between variables, we can see that all of them are approximately zero, which is great

```
> cor(Weekly[, -9])
              Year         Lag1        Lag2        Lag3         Lag4         Lag5
Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923 -0.030519101
Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876 -0.008183096
Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535 -0.072499482
Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865  0.060657175
Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000 -0.075675027
Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027  1.000000000
Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617 -0.058517414
Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873  0.011012698
            Volume        Today
Year    0.84194162 -0.032459894
Lag1   -0.06495131 -0.075031842
Lag2   -0.08551314  0.059166717
Lag3   -0.06928771 -0.071243639
Lag4   -0.06107462 -0.007825873
Lag5   -0.05851741  0.011012698
Volume  1.00000000 -0.033077783
Today  -0.03307778  1.000000000
```

1.2 b

As I used the glm() function to perform logistic regression on the whole dataset, this is the result I got this result:

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

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
```

As you can see only the variable Lag2 is statistically significant with 0.05 significant level since its p-value is 0.0296 which is smaller than 0.0296.

1.3 c

Then I use predict() function to test my predictor on the deflaut original training data. And then I create create vector to record my prediction. Lastly, I used table() to create the confusion matrix. The number at the diagonal is the number of correct prediction and the number off the diagonal is the number of incorrect prediction.:

```
> glm.probs = predict(glm.fit, type = "response")
> glm.probs[1:10]
        1         2         3         4         5         6         7         8 
0.6086249 0.6010314 0.5875699 0.4816416 0.6169013 0.5684190 0.5786097 0.5151972 
        9        10 
0.5715200 0.5554287 
> contrasts(Direction)
     Up
Down  0
Up    1
> glm.pred=rep ("Down",1089)
> glm.pred[glm.probs >.5]="Up"
> table(glm.pred, Direction)
        Direction
glm.pred Down  Up
    Down   54  48
    Up    430 557
> mean(glm.pred== Direction)
[1] 0.5610652
```

As you can see, the predictor mostly made mistake when the market is down while it predict UP (430 times). And I used mean() function to calculate the correction rate, which is 0.56, better than randomly guessing.

1.4 d

First, I split the dataset into two part: training data (Year¡2009) and test data (Year¿=2009).

```
> train = (Year < 2009)
> Weekly.heldOut = Weekly[!train, ]
> Direction.heldOut = Direction[!train]
```

Then, I perform the logistic regression with only Lag2 as feature on my training data. And then I tested my predictor on my test data, similarly above. In the end, I calculate the confusion matrix and correction rate, 0.625, which is higher:

```
> glm.fit=glm(Direction ~ Lag2, data=Weekly,family=binomial, subset = train)
> glm.probs = predict(glm.fit, Weekly.heldOut, type = "response")
> contrasts(Direction)
     Up
Down  0
Up    1
> glm.pred=rep ( cbind(..., deparse.level = 1)
> glm.pred[glm.probs >.5]="Up"
> table (glm.pred , Direction.heldOut)
        Direction.heldOut
glm.pred Down Up
    Down    9  5
    Up     34 56
> mean(glm.pred== Direction.heldOut)
[1] 0.625
```

1.5 e

So, this time, I use the lda() function, which implement linear discrimintive analysis on my training data. It models each class as Guassina distribution and use Bayes' classifier to find the class.:

```
> lda.fit=lda(Direction~Lag2,data=Weekly ,subset =train)
> plot(lda.fit)
> lda.pred=predict (lda.fit , Weekly.heldOut)
> lda.class = lda.pred$class
> table(lda.class, Direction.heldOut)
         Direction.heldOut
lda.class Down Up
     Down    9  5
     Up     34 56
> mean(lda.class==Direction.heldOut)
[1] 0.625
> lda.fit
Call:
lda(Direction ~ Lag2, data = Weekly, subset = train)

Prior probabilities of groups:
     Down        Up
0.4477157 0.5522843

Group means:
            Lag2
Down -0.03568254
Up    0.26036581

Coefficients of linear discriminants:
           LD1
Lag2 0.4414162
```

As I calculate the confusion matrix similarily, we can see that it comes out of the same answer as logistic regression, with 0.625 correction rate.

1.6 f

Lastly, I used the KNN algorithm with parameter k =1. First, I create the training data and test data matrix with each row as value of Lag2. Then I use the knn()function to directly make a prediction for each input test data. And the result is 0.5 correction rate, which is the same as randomly guessing:

```
> train.X = cbind(cbind(Lag2)[train, ])
> test.X = cbind(cbind(Lag2)[!train, ])
> train.Direction = Direction[train]
> set.seed(1)
> knn.pred = knn(train.X, test.X, train.Direction, k=1)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   21 30
    Up     22 31
> mean(knn.pred==Direction.heldOut)
[1] 0.5
```

1.7 g

(g) It appears that logistic regression and LDA are better then KNN with k =1 since the error rate is smaller.

1.8 h

First, I used different values for k in the KNN algorithm, and the following is my result:

```
> knn.pred = knn(train.X, test.X, train.Direction, k=7)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   16 19
    Up     27 42
> mean(knn.pred==Direction.heldOut)
[1] 0.5576923
> knn.pred = knn(train.X, test.X, train.Direction, k=9)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   17 20
    Up     26 41
> mean(knn.pred==Direction.heldOut)
[1] 0.5576923
> knn.pred = knn(train.X, test.X, train.Direction, k=11)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   18 22
    Up     25 39
> mean(knn.pred==Direction.heldOut)
[1] 0.5480769
> knn.pred = knn(train.X, test.X, train.Direction, k=13)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   20 20
    Up     23 41
> mean(knn.pred==Direction.heldOut)
[1] 0.5865385
> knn.pred = knn(train.X, test.X, train.Direction, k=15)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   20 20
    Up     23 41
> mean(knn.pred==Direction.heldOut)
[1] 0.5865385
```

As you can see, when I increase the value of k, the correction rate is increasing but as the k reaches certain value, it remains the same. Then as I kept increasing k value, the correction rate starts decreasing:

```
> set.seed(1)
> knn.pred = knn(train.X, test.X, train.Direction, k=17)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   20 20
    Up     23 41
> mean(knn.pred==Direction.heldOut)
[1] 0.5865385
> set.seed(1)
> knn.pred = knn(train.X, test.X, train.Direction, k=19)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   19 21
    Up     24 40
> mean(knn.pred==Direction.heldOut)
[1] 0.5673077
> set.seed(1)
> knn.pred = knn(train.X, test.X, train.Direction, k=21)
> table(knn.pred, Direction.heldOut)
        Direction.heldOut
knn.pred Down Up
    Down   18 21
    Up     25 40
> mean(knn.pred==Direction.heldOut)
[1] 0.5576923
```

Then, I tried to transform the feature with x^2 for the logistic regression and LDA:

```
> train = (Year < 2009)
> Weekly.heldOut = Weekly[!train, ]
> Direction.heldOut = Direction[!train]
> glm.fit=glm(Direction ~ Lag2 + I(Lag2^2), data=Weekly,family=binomial, subset = train)
> glm.probs = predict(glm.fit, Weekly.heldOut, type = "response")
> contrasts(Direction)
     Up
Down  0
Up    1
> glm.pred=rep ("Down",104)
> glm.pred[glm.probs >.5]="Up"
> table (glm.pred , Direction.heldOut)
        Direction.heldOut
glm.pred Down Up
    Down    8  4
    Up     35 57
> mean(glm.pred== Direction.heldOut)
[1] 0.625
```

The result of modified logistic regression has not changed, but the result of LDA has changed: the correction rate decreased.

```
> lda.fit=lda(Direction~Lag2 + I(Lag2^2),data=Weekly ,subset =train)
> lda.fit
Call:
lda(Direction ~ Lag2 + I(Lag2^2), data = Weekly, subset = train)

Prior probabilities of groups:
     Down        Up
0.4477157 0.5522843

Group means:
            Lag2 I(Lag2^2)
Down -0.03568254  4.828121
Up    0.26036581  5.428657

Coefficients of linear discriminants:
                 LD1
Lag2      0.43203575
I(Lag2^2) 0.02957998
> plot(lda.fit)
> lda.pred=predict (lda.fit , Weekly.heldOut)
> lda.class = lda.pred$class
> table(lda.class, Direction.heldOut)
         Direction.heldOut
lda.class Down Up
     Down    7  4
     Up     36 57
> mean(lda.class==Direction.heldOut)
[1] 0.6153846
> train = (Year < 2009)
```