Assignment - 3

Problem 1: Show that the ridge regression estimate is the mean (and mode) of the posterior distribution, under a gaussian prior $\beta \sim N(0, \tau^2 I)$, and Gaussian sampling model $y \sim N(X\beta, \sigma^2 I)$. Find the relationship between the regularization parameter λ in the ridge formula, and the variance τ^2 and σ^2.

Solution: The likelihood : $y \sim N(X\beta, \sigma^2 I)$. Prior: $\beta \sim N(0, \tau^2 I)$
Assume that our input data x is already centered so that we can ignore the intercept term β_0.
Posterior distribution is :

$$p(\beta|y, X) = \frac{1}{Z} p(y|\beta, X)\, p(\beta)$$

where $Z = Z(y, X)$

Log poseterior distribution :

$$p(\beta|y, X) = \frac{1}{Z} p(y|\beta, x)\, p(\beta)$$

$$= \frac{1}{Z} * \frac{1}{(2\pi)^{p/2} \sigma^p} e^{\left\{-\frac{1}{2\sigma^2}(y - X\beta)^t (y - X\beta)\right\}} *$$

$$\frac{1}{(2\pi)^{p/2} \tau} e^{\left\{-\frac{\beta^T \beta}{2\tau^2}\right\}}$$

$$\log p(\beta|y, X) = -\log Z - k - \frac{1}{2\sigma^2}(y - X\beta)^t (y - X\beta) - \frac{\beta^t \beta}{2\tau^2}$$

Taking derivative & setting to 0, will give

$$\hat{\beta} = \left(X^t X + \frac{\sigma^2}{\tau^2} I\right)^{-1} X^t y$$

Letting $\lambda = \frac{\sigma^2}{\tau^2}$, we see equivalence, & its shows that $P(\beta | y, X)$ is Gaussian and its mean and mode coincide.

Now, $$\Sigma^{-1} = \frac{1}{\sigma^2}\left(X^T X + \frac{\sigma^2}{\tau^2} I\right)$$

which gives $\hat{\beta} = \frac{1}{\sigma^2} \Sigma X^T y$ which is the mean.

Problem 4: (a) Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in class.

Sol:- $$p(x) = \frac{\exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2)}{1 + \exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2)}$$

where, x_1 = hours studied x_2 = undergrad gpA

$$p(x) = \frac{e^{-6+0.05x_1+x_2}}{1+e^{-6+0.05x_1+x_2}} = \frac{e^{-6+0.05*40+3.5}}{1+e^{-6+0.05*40+3.5}} = \boxed{37.75}$$

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

Solution: $p(x) = \frac{e^{-6+0.05x_1+3.5}}{1+e^{-6+0.05x_1+3.5}}$ substituting $P(A) = 0.5$

$x_2 = 3.5$ $x_1 = ?$

$$0.5 = \frac{e^{-6+0.05x_1+3.5}}{1+e^{-6+0.05x_1+3.5}} \Rightarrow \boxed{x_1 = 50 \text{ hours}}$$

Assignment 3

Pawanjeet Kaur

10/25/2019

Problem 2: Ex. 14[a-f] (Chapter 3, page 125) [2pt]

This problem focuses on the collinearity problem.

(a) Perform the following commands in R: The last line corresponds to creating a linear model in which y is a function of x1 and x2. Write out the form of the linear model. What are the regression coefficients?

```
set.seed(1)
x1=runif(100)
x2=0.5*x1+rnorm(100)/10
y=2+2*x1+0.3*x2+rnorm(100)
```

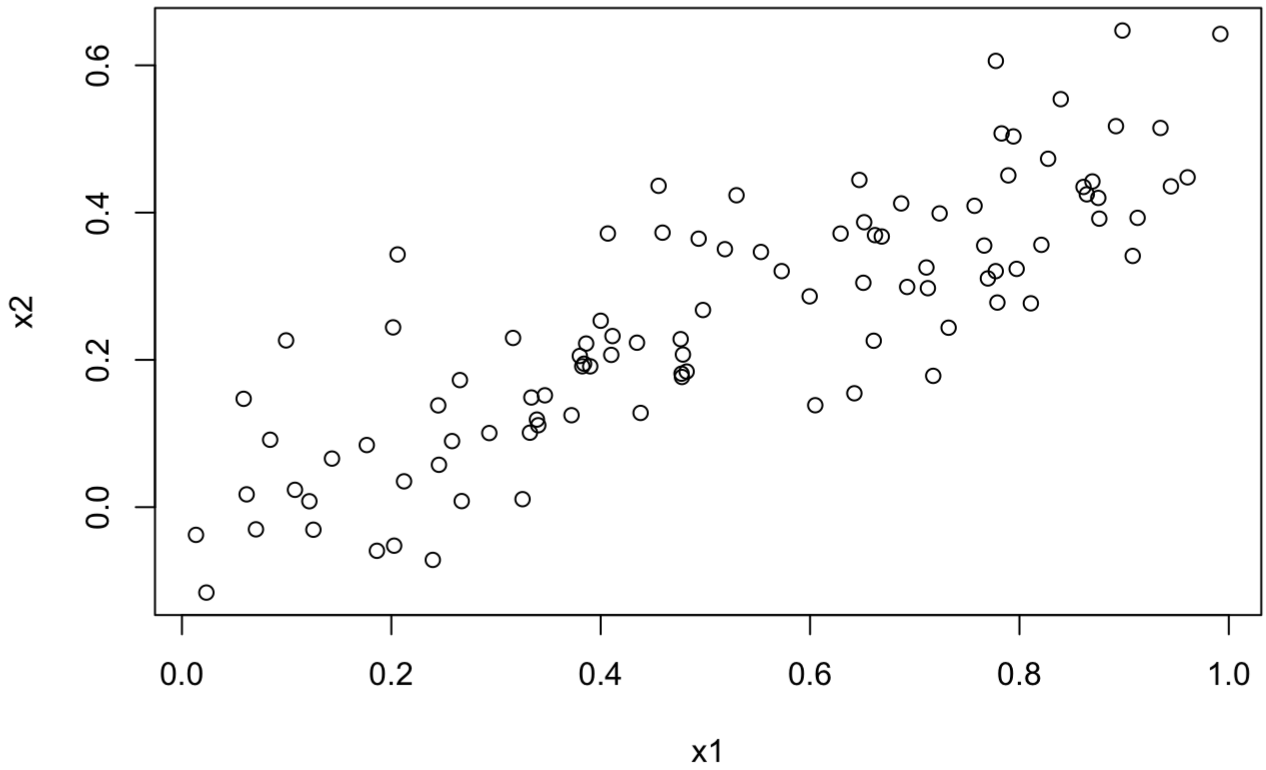
The regression coefficients are β0 = 2+rnorm(100), β1 = 2 and β2 = 0.3

(b) What is the correlation between x1 and x2? Create a scatterplot displaying the relationship between the variables.

```
cor(x1, x2)
```

```
## [1] 0.8351212
```

```
plot(x1, x2)
```



(c) Using this data, fit a least squares regression to predict y using x1 and x2. Describe the results obtained. What are β̂0, β̂1, and β̂2? How do these relate to the true β0, β1, and β2? Can you reject the null hypothesis H0 : β1 = 0? How about the null hypothesis H0 : β2 = 0?

```
lm_fit <- lm(y ~ x1 + x2)

summary(lm_fit)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8311 -0.7273 -0.0537  0.6338  2.3359
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1305     0.2319   9.188 7.61e-15 ***
## x1            1.4396     0.7212   1.996   0.0487 *
## x2            1.0097     1.1337   0.891   0.3754
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.056 on 97 degrees of freedom
## Multiple R-squared:  0.2088, Adjusted R-squared:  0.1925
## F-statistic:  12.8 on 2 and 97 DF,  p-value: 1.164e-05
```

β̂1 is 1.4396 ,β̂2 is 1.0097 , β̂0 is 2.130. For β̂1 we reject ho for ha. For β2 we cannot reject null hypothesis.

(d) Now fit a least squares regression to predict y using only x1. Comment on your results. Can you reject the null hypothesis H0 :β1 =0?

```
lm_fit_1 <- lm(y ~ x1)

summary(lm_fit_1)
```

```
##
## Call:
## lm(formula = y ~ x1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.89495 -0.66874 -0.07785  0.59221  2.45560
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1124     0.2307   9.155 8.27e-15 ***
## x1            1.9759     0.3963   4.986 2.66e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.055 on 98 degrees of freedom
## Multiple R-squared:  0.2024, Adjusted R-squared:  0.1942
## F-statistic: 24.86 on 1 and 98 DF,  p-value: 2.661e-06
```

As the p-value is less than alpha we can reject null hypothesis.

(e) Now fit a least squares regression to predict y using only x2. Comment on your results. Can you reject the null hypothesis H0 :β1 =0?

```
lm_fit_2 <- lm(y ~ x2)

summary(lm_fit_2)
```

```
##
## Call:
## lm(formula = y ~ x2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.62687 -0.75156 -0.03598  0.72383  2.44890
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3899     0.1949   12.26  < 2e-16 ***
## x2            2.8996     0.6330    4.58 1.37e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.072 on 98 degrees of freedom
## Multiple R-squared:  0.1763, Adjusted R-squared:  0.1679
## F-statistic: 20.98 on 1 and 98 DF,  p-value: 1.366e-05
```

As the p-value is less than alpha we can reject ho

(f) Do the results obtained in (c)–(e) contradict each other? Explain your answer.

Yes the results in (c) to (e) contradict as we can see in part c we for β1 we rejected null hypothesis and for β2 we could not reject null hypothesis. Whereas in part d and e we rejected null hypothesis.

Problem 3: Ex. 5 (Chapter 4, page 169) [1pt]

We now examine the differences between LDA and 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?

In QDA on test data the bias will decrease hence, QDA will perform better than LDA.Whereas if the Bayes decision boundary is linear, we know LDA has linear decision boundary hence LDA will perform better than QDA.

(b) If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

If bayes decision boundary is non linear, we can say QDA will perform better on both training and test sets

(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?

With increase in the sample size the accuracy of QDA will improve.As we know with increase in sample size for non linear methods the bias will decrease along with the variance of models.

(d) True or False: Even 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.

False, for small value QDA is more prone to errors and add noise to model. Hence test errors in QDA will be high. So, LDA performs better in such scenario and given statement is false.

Problem 4: Ex. 6 (Chapter 4, page 170) [1pt]

```
library(ISLR)
dim(Weekly)
```

```
## [1] 1089    9
```

```
str(Weekly)
```

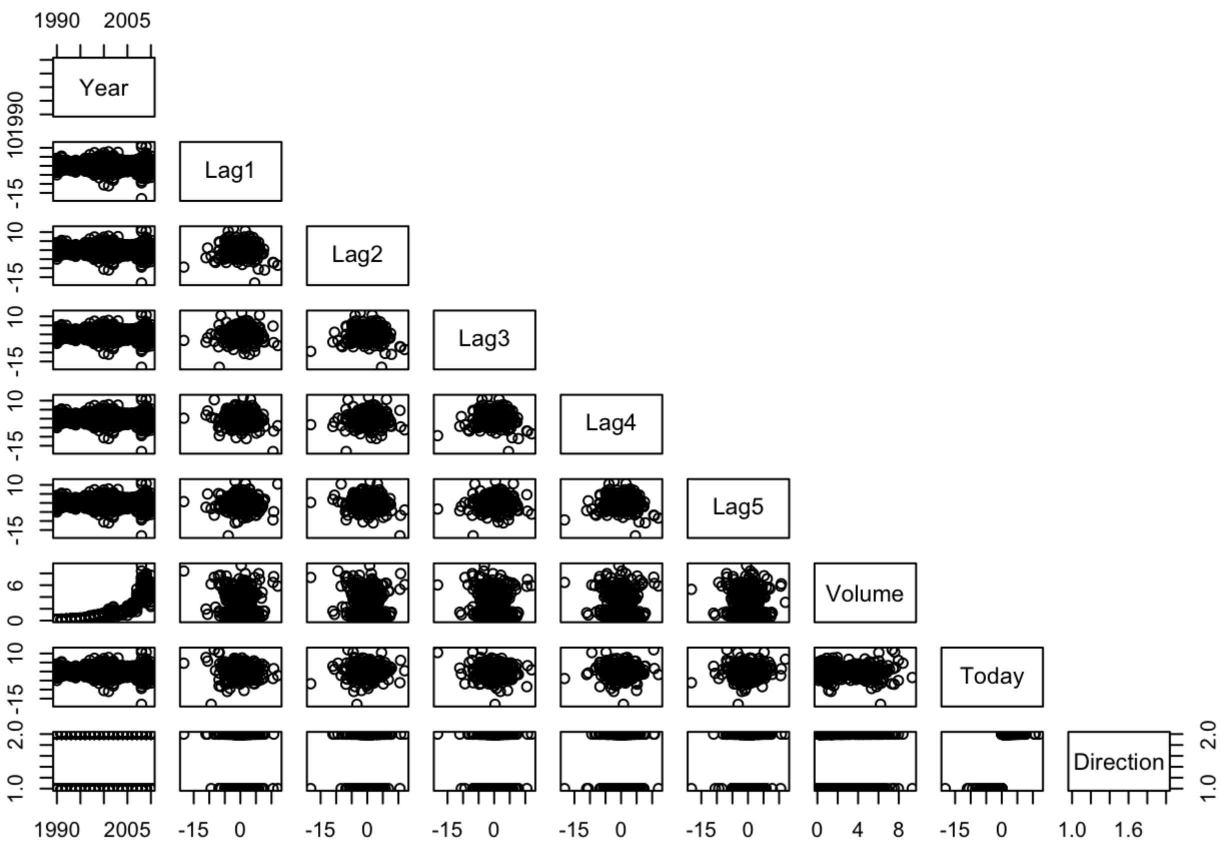
```
## 'data.frame':    1089 obs. of  9 variables:
##  $ Year     : num  1990 1990 1990 1990 1990 1990 1990 1990 1990 1990 ...
##  $ Lag1     : num  0.816 -0.27 -2.576 3.514 0.712 ...
##  $ Lag2     : num  1.572 0.816 -0.27 -2.576 3.514 ...
##  $ Lag3     : num  -3.936 1.572 0.816 -0.27 -2.576 ...
##  $ Lag4     : num  -0.229 -3.936 1.572 0.816 -0.27 ...
##  $ Lag5     : num  -3.484 -0.229 -3.936 1.572 0.816 ...
##  $ Volume   : num  0.155 0.149 0.16 0.162 0.154 ...
##  $ Today    : num  -0.27 -2.576 3.514 0.712 1.178 ...
##  $ Direction: Factor w/ 2 levels "Down","Up": 1 1 2 2 2 1 2 2 2 1 ...
```

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

```
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
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
##  Median :  0.2380   Median :  0.2340   Median :1.00268
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
##      Today          Direction
##  Min.   :-18.1950   Down:484
##  1st Qu.: -1.1540   Up  :605
##  Median :  0.2410
##  Mean   :  0.1499
##  3rd Qu.:  1.4050
##  Max.   : 12.0260
```

```
pairs(Weekly, upper.panel = NULL)
```




```
# Corelation between different variables
cor(Weekly[,-9])[1,]
```

```
##        Year        Lag1        Lag2        Lag3        Lag4        Lag5
##  1.00000000 -0.03228927 -0.03339001 -0.03000649 -0.03112792 -0.03051910
##      Volume       Today
##  0.84194162 -0.03245989
```

Volume is highly correlated with year

(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?

```
log_fit <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly,f
amily = "binomial")

summary(log_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
```

Lag2 is statisitically significant

(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.

```
prediction <- predict(log_fit, type= "response")
prediction <- ifelse(prediction >= 0.5, 'Up', 'Down')

conf_matrix <- table(prediction , Weekly$Direction)

accuracy_logistic_1 <- sum(diag(conf_matrix))/sum(conf_matrix)
accuracy_logistic_1
```

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

(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).

```
summary(Weekly$Year)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1990    1995    2000    2000    2005    2010
```

```
train_data <- Weekly[Weekly$Year <= 2008,]
test_data <- Weekly[Weekly$Year > 2008,]

log_fit_2 <- glm(Direction ~ Lag2 , data = train_data , family = 'binomial')

summary(log_fit_2)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2, family = "binomial", data = train_data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.536  -1.264   1.021   1.091   1.368
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.20326    0.06428   3.162  0.00157 **
## Lag2         0.05810    0.02870   2.024  0.04298 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1350.5  on 983  degrees of freedom
## AIC: 1354.5
##
## Number of Fisher Scoring iterations: 4
```

```
pred_log_fit_2 <- predict(log_fit_2, newdata = test_data , type = 'response')
pred_log_fit_2 <- ifelse(pred_log_fit_2 >= 0.5 , 'Up', 'Down')
pred_log_fit_2
```

```
##    986    987    988    989    990    991    992    993    994    995
##   "Up"   "Up" "Down" "Down"   "Up"   "Up"   "Up" "Down" "Down" "Down"
##    996    997    998    999   1000   1001   1002   1003   1004   1005
## "Down"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"
##   1006   1007   1008   1009   1010   1011   1012   1013   1014   1015
## "Down"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"
##   1016   1017   1018   1019   1020   1021   1022   1023   1024   1025
##   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"
##   1026   1027   1028   1029   1030   1031   1032   1033   1034   1035
##   "Up"   "Up"   "Up"   "Up" "Down"   "Up"   "Up"   "Up"   "Up"   "Up"
##   1036   1037   1038   1039   1040   1041   1042   1043   1044   1045
##   "Up"   "Up"   "Up"   "Up"   "Up"   "Up" "Down"   "Up"   "Up"   "Up"
##   1046   1047   1048   1049   1050   1051   1052   1053   1054   1055
##   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"
##   1056   1057   1058   1059   1060   1061   1062   1063   1064   1065
##   "Up" "Down"   "Up" "Down"   "Up"   "Up"   "Up"   "Up" "Down" "Down"
##   1066   1067   1068   1069   1070   1071   1072   1073   1074   1075
##   "Up"   "Up"   "Up"   "Up"   "Up" "Down"   "Up"   "Up"   "Up"   "Up"
##   1076   1077   1078   1079   1080   1081   1082   1083   1084   1085
##   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"   "Up"
##   1086   1087   1088   1089
##   "Up"   "Up"   "Up"   "Up"
```

```
conf_matrix_2 <- table(pred_log_fit_2 , test_data$Direction)
accuracy_logisitic_2 <- sum(diag(conf_matrix_2))/sum(conf_matrix_2)

accuracy_logisitic_2
```

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

(e) Repeat (d) using LDA.

```
require('MASS')
```

```
## Loading required package: MASS
```

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

summary(lda_model)
```

```
##         Length Class  Mode
## prior   2      -none- numeric
## counts  2      -none- numeric
## means   2      -none- numeric
## scaling 1      -none- numeric
## lev     2      -none- character
## svd     1      -none- numeric
## N       1      -none- numeric
## call    3      -none- call
## terms   3      terms  call
## xlevels 0      -none- list
```

```
pred_lda <- predict(lda_model, newdata = test_data)
conf_matrix_3 <- table(pred_lda$class , test_data$Direction)

accuracy_lda <- sum(diag(conf_matrix_3))/sum(conf_matrix_3)
accuracy_lda
```

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

(f) Repeat (d) using QDA.

```
qda_model_10 <- qda(Direction ~ Lag2 , data = train_data)
summary(qda_model_10)
```

```
##         Length Class  Mode
## prior   2      -none- numeric
## counts  2      -none- numeric
## means   2      -none- numeric
## scaling 2      -none- numeric
## ldet    2      -none- numeric
## lev     2      -none- character
## N       1      -none- numeric
## call    3      -none- call
## terms   3      terms  call
## xlevels 0      -none- list
```

```
pred_qda <- predict(qda_model_10 , newdata = test_data)
conf_matrix_4 <- table(pred_qda$class , test_data$Direction)
accuracy_qda <- sum(diag(conf_matrix_4))/sum(conf_matrix_4)
accuracy_qda
```

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

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

```
require(class)
```

```
## Loading required package: class
```

```
knn_1 <- knn(train = data.frame(train_data$Lag2), test = data.frame(test_data$Lag2),
cl = train_data$Direction , k=1)

conf_matrix_5 <- table(knn_1 , test_data$Direction)
conf_matrix_5
```

```
##
## knn_1  Down Up
##   Down   21 30
##   Up     22 31
```

```
accuracy_knn = sum(diag(conf_matrix_5))/sum(conf_matrix_5)
accuracy_knn
```

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

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

```
# Accuracy of logisitic model with all variables
accuracy_logistic_1
```

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

```
# Accuracy of logisitic model with statisitically significant variables
accuracy_logisitic_2
```

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

```
# Accuracy of lda
accuracy_lda
```

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

```
# Accuracy of qda
accuracy_qda
```

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

```
# Accuracy of knn
accuracy_knn
```

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

The LDA model with Lag2 as its only predictor did the best.

i) Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables, method, and associated confusion 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.

```
glm_fit_transform <- glm( Direction ~ Lag2+I(Lag2^2)+I(Lag2^3),data = train_data, fam
ily = "binomial")
summary(glm_fit_transform)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2 + I(Lag2^2) + I(Lag2^3), family = "binomial",
##     data = train_data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.194  -1.245   1.008   1.108   1.142
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 0.1608285  0.0714552   2.251   0.0244 *
## Lag2        0.0491970  0.0340597   1.444   0.1486
## I(Lag2^2)   0.0095243  0.0072076   1.321   0.1864
## I(Lag2^3)   0.0005092  0.0005445   0.935   0.3497
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1348.5  on 981  degrees of freedom
## AIC: 1356.5
##
## Number of Fisher Scoring iterations: 4
```

```
pred_glm_fit_transf <- predict(glm_fit_transform, newdata = test_data , type = 'respo
nse')
pred_glm_fit_transf <- ifelse(pred_glm_fit_transf >= 0.5 , 'Up', 'Down')

conf_matrix_glm_transf <- table(pred_glm_fit_transf , test_data$Direction)
accuracy_logisitic_glm_transf <- sum(diag(conf_matrix_glm_transf))/sum(conf_matrix_gl
m_transf)

accuracy_logisitic_glm_transf
```

```
## [1] 0.4134615
```

```
# Square Root
glm_fit_sqrt <- glm(Direction~sqrt(abs(Lag2)),data = train_data, family = "binomial")
summary(glm_fit_sqrt)
```

```
##
## Call:
## glm(formula = Direction ~ sqrt(abs(Lag2)), family = "binomial",
##     data = train_data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.405  -1.263   1.058   1.093   1.136
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.09488    0.15028   0.631    0.528
## sqrt(abs(Lag2))  0.09961    0.11788   0.845    0.398
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1354.0  on 983  degrees of freedom
## AIC: 1358
##
## Number of Fisher Scoring iterations: 3
```

```
pred_glm_fit_sqrt <- predict(glm_fit_sqrt, newdata = test_data , type = 'response')
pred_glm_fit_sqrt <- ifelse(pred_glm_fit_sqrt >= 0.5 , 'Up', 'Down')

conf_matrix_glm_sqrt <- table(pred_glm_fit_sqrt , test_data$Direction)
accuracy_logisitic_glm_sqrt <- sum(diag(conf_matrix_glm_sqrt))/sum(conf_matrix_glm_sq
rt)

accuracy_logisitic_glm_sqrt
```

```
## [1] 0.4134615
```

```
# Interaction Effect
glm_fit_int <- glm(Direction~ Lag2*Lag1,data = train_data, family = "binomial")
summary(glm_fit_int)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2 * Lag1, family = "binomial", data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.573  -1.259   1.003   1.086   1.596
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.211419   0.064589   3.273  0.00106 **
## Lag2         0.053471   0.029193   1.832  0.06700 .
## Lag1        -0.051505   0.030727  -1.676  0.09370 .
## Lag2:Lag1    0.001921   0.007460   0.257  0.79680
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1346.9  on 981  degrees of freedom
## AIC: 1354.9
##
## Number of Fisher Scoring iterations: 4
```

```
pred_glm_fit_int <- predict(glm_fit_int, newdata = test_data , type = 'response')
pred_glm_fit_int <- ifelse(pred_glm_fit_int >= 0.5 , 'Up', 'Down')

conf_matrix_glm_int <- table(pred_glm_fit_int , test_data$Direction)
accuracy_logisitic_glm_int <- sum(diag(conf_matrix_glm_int))/sum(conf_matrix_glm_int)

accuracy_logisitic_glm_int
```

```
## [1] 0.5769231
```

```
str(train_data)
```

```
## 'data.frame':    985 obs. of  9 variables:
##  $ Year     : num  1990 1990 1990 1990 1990 1990 1990 1990 1990 1990 ...
##  $ Lag1     : num  0.816 -0.27 -2.576 3.514 0.712 ...
##  $ Lag2     : num  1.572 0.816 -0.27 -2.576 3.514 ...
##  $ Lag3     : num  -3.936 1.572 0.816 -0.27 -2.576 ...
##  $ Lag4     : num  -0.229 -3.936 1.572 0.816 -0.27 ...
##  $ Lag5     : num  -3.484 -0.229 -3.936 1.572 0.816 ...
##  $ Volume   : num  0.155 0.149 0.16 0.162 0.154 ...
##  $ Today    : num  -0.27 -2.576 3.514 0.712 1.178 ...
##  $ Direction: Factor w/ 2 levels "Down","Up": 1 1 2 2 2 1 2 2 2 1 ...
```

```
train_knn_X <- data.frame(train_data[,3])
train_knn_Y <- data.frame(train_data[,9])

dim(train_knn_X)
```

```
## [1] 985   1
```

```
dim(train_knn_Y)
```

```
## [1] 985   1
```

```
test_knn_X <- data.frame(test_data[,3])
test_knn_Y <- data.frame(test_data[,9])

dim(test_knn_X)
```

```
## [1] 104   1
```

```
dim(test_knn_Y)
```

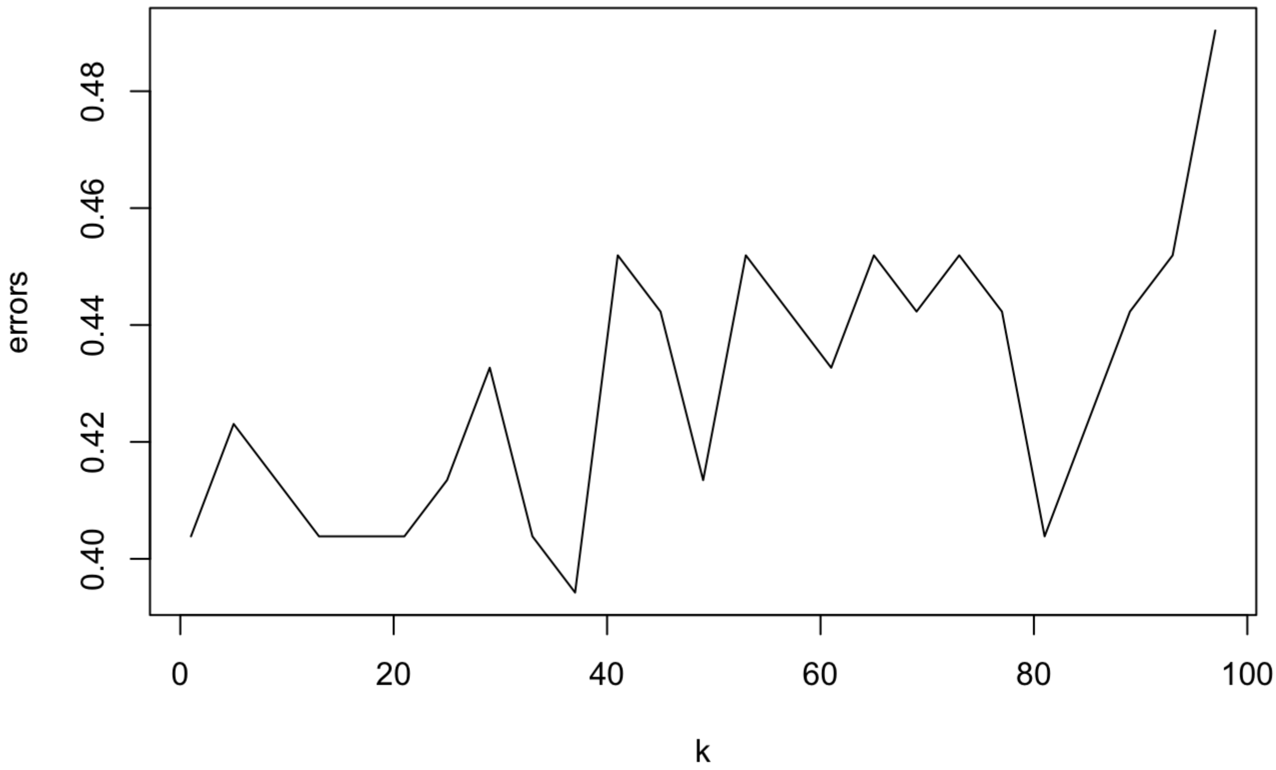
```
## [1] 104   1
```

```
errors <- c()
maxK <- 100
step_k <- 4

for(j in seq(1,maxK,step_k)){
  knn_run <- knn(train = data.frame(train_data$Lag2), test = data.frame(test_data$Lag
2), cl = train_data$Direction,k = j)
  pred <- table(knn_run,test_data$Direction)
  acc <- sum(diag(pred))/sum(pred)
  errors <- c(1-acc , errors)
}

data <- cbind(seq(1,maxK,step_k),errors)

plot(data,type="l",xlab="k")
```


errors
0.40
0.42
0.44
0.46
0.48
0
20
40
60
80
100
k