HW2

Anish Mohan

February 10, 2016

1. Q1

- 1a.

  => $P(Y = A|X) = \frac{e^{\beta_0 + X1*\beta_1 + X2*\beta_2}}{1+e^{\beta_0 + X1*\beta_1 + X2*\beta_2}}$

  => $P(Y = A|X) = \frac{e^{-6+40*0.05+3.5*1}}{1+e^{-6+40*0.05+3.5*1}}$

  =>The probability of getting an A is 0.3375

- 1b.

  => $P(Y = A|X) = \frac{e^{\beta_0 + X1*\beta_1 + X2*\beta_2}}{1+e^{\beta_0 + X1*\beta_1 + X2*\beta_2}}$

  => $\frac{P(Y=A|X)}{1-P(Y=A|X)} = e^{\beta_0 + X1*\beta_1 + X2*\beta_2}$

  => $\mathrm{Log}\left(\frac{P(Y=A|X)}{1-P(Y=A|X)}\right) = \beta_0 + X1 * \beta_1 + X2 * \beta_2$

  => $\mathrm{Log}\left(\frac{0.5}{1-0.5}\right) = -6 + X1 * 0.05 + 3.5 * 1$

  => $0 = -2.5 + X1 * 0.05$

  => $X1 = 2.5/0.05$

  => $X1 = 50$

  => Student must study atleast 50 hours to have a 50% probability of getting an A in the exam.

2. Q2

- We are making a prediction for the response Y for a particular value of the predictor X using a particular statistical learning model. Also given is a dataset.
- We use Bootstrap on the given dataset to get a subset of dataset and use the statistical learning method on it for estimating the parameters of the model for making the prediction of Y from X.
- Per the Boostrap, re-run the learning method with various subsets obtained by Bootstrapping the original dataset.
- This process will give us a distribution for the values of the parameters of the model used for predicting Y from X. By calculating in the standar error in the parameters of the model, we can also calculate the standard error in the estimates of Y from the model.

3. Q3

- 3a.
  - Obtain the dataset for running the statistical model. Let n be number of datapoints
  - Divide the dataset into k-groups; if n is perfectly divisible by k, then we will have n/k groups else some groups will have n/k+1 elements. Note that these are non overlapping sets
  - The groups can be named as n_1, n_2...n_k
  - In the first iteration, fit the model on n_2, n_3, n_4...n_k groups. This is the training set. Use the model to predict the response variable for n_1 group. This is the validation set Calculate the MSE of this group=MSE_1
  - In, the next iteration, fit the model on n_1,n_3, n_4...n_k and use it to predict the response variable for n_2 group. This will be MSE_2.
  - In similar ways we can calculate MSE_3, MSE_4..MSE_k. The CV error estimate is given by $\frac{1}{k} * \sum_{i=1}^{k} MSE_k$. This will be the average Test set error for the chosen statistical model
- 3b.
  - 3b. i.
    - In validation set approach, the statistical model is fit on the validation set which is a subset of the original dataset. The statistical model does not see the datapoints in the test set. In general, a statistical learning method works better when it is fit on most of the data available from the data set. Hence, the validation set error rate may tend to overestimate the test error rate. K-fold validation iterates the statistical methods over K subsets of the the dataset thus refining the validation set error rate and bringing in line with the test error rate.
    - Another drawback is that the validation estimate of test error rate can be highly variable depending on which observations are included in the training set and the test set. K-fold validation considers each group for training and test set thus reducing the variability in the validation estimate of the test error rate.
    - K-Fold validation requires that each of the K subsets are a test set once hence the fitting model has to be run K times. Hence it is bit more computationally expensive than the validation set approach.
  - 3b. ii.

- LOOCV is special case of K- fold validation with n=K i.e each subset has only 1 element. LOOCV is computationally more expensive than K-fold validation because the process has to be run n times.
- In LOOCV, only one element is held for test and rest are used for training hence the training sets are very similar. Since majority of the data is used for training, it has lower bias, but the variance is higher thank K-fold validation i.e there is a bias variance tradeoff while choosing LOOCV and K-fold validation.

4. Q4

- 4a. Training RSS steadily increases. The best fit for the training error is with $\lambda=0$, when the best linear model is fit for training data. As λ starts increasing, we penalize larger values of β thereby increasing the training RSS compared to the ordinary least squares
- 4b. Test RSS: Decrease initially and then evtually start increasing in a U Shape. As λ increases the flexibility of ridge regression fit decreases, leading to decreased variance but increased bias. The decreased variance is at the expense of a slight increase in bias thus reducing the test RSS. However beyond a point, the increase in bias is much more significant than decrease in variance and thus the test RSS increases
- 4c. Variance decreases steadily as λ increases; When λ increases, the flexibility of the model decreases and we are penalizing higher values of β; As the flexibility of the model decreases the variance of the model decreases as well.
- 4d. Squared bias increases steadily as λ; As λ increases the flexibility of the method decreases and hence squared bias increases. As λ increases higher values of β are being penalized and it is being pushed towards 0;
- 4e. Irreducibe error remains constant as it is not dependent on the value of λ

5. Q5

- 5a.

```
library(ISLR)

## Warning: package 'ISLR' was built under R version 3.2.2

attach(Weekly)
glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume, data=Weekly, fam
ily=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
```

Lag2 seems to be statistical significant result as P value is <0.05

- 5b.

```
glm.probs = predict(glm.fit, type="response")
glm.pred=rep("Down", length(glm.probs))
glm.pred[glm.probs>0.5]="Up"
table(glm.pred,Weekly$Direction)
```

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

- ### of correct predicitions= 557+54= 611 (56.1%)

- ### of incorrect predictions = 430+48= 478 (43.9%)

- There is significant error in prediction in the weeks the market goes down. When the market goes down, the model is only correct for 54/(54+430)=11.2%
- For the weeks market goes up, the model has a good prediction capability and is correct 557/(557+48)=92.1%

- 5c.

```
train=(Year<2009)
Weekly.2008=Weekly[train,]
Weekly.2010=Weekly[!train,]
Direction.2008=Direction[train]
Direction.2010=Direction[!train]
glm.fit2=glm(Direction~Lag1+Lag2+Lag3,data=Weekly.2008,family=binomial)
glm.probs2 = predict(glm.fit2, Weekly.2010,type="response")
glm.pred2=rep("Down", length(glm.probs2))
glm.pred2[glm.probs2>0.5]="Up"
table(glm.pred2,Weekly.2010$Direction)
```

```
##
## glm.pred2 Down Up
##      Down    8  9
##      Up     35 52
```

- % of Correct predictions= (52+8)/(52+8+9+35)= 57.69%

- 5d.

```
library(MASS)
lda.fit=lda(Direction~Lag1+Lag2+Lag3, data=Weekly.2008)
lda.pred=predict(lda.fit,Weekly.2010)
lda.class=lda.pred$class
table(lda.class, Direction.2010)
```

```
##          Direction.2010
## lda.class Down Up
##      Down    8  9
##      Up     35 52
```

- Correct predictions= (52+8)/(52+8+9+35)= 57.69%

- 5e

```
library(class)
train.X=cbind(Lag1,Lag2,Lag3)[train,]
test.X=cbind(Lag1,Lag2, Lag3)[!train,]
train.Direction=Direction[train]
set.seed(2016)
knn.pred=knn(train.X, test.X,train.Direction,k=1)
table(knn.pred,Direction.2010)
```

```
##          Direction.2010
## knn.pred Down Up
##     Down   19 29
##     Up     24 32
```

  - Correct predictions= (19+32)/(24+29+19+32)= 49.03%
- 5f.
  - Best results are provided by LDA and Logistic Regression with about 57.7% accuracy
  - KNN's results are bit worse at 49.03% accuracy.
- 5g.
  - LDA assumes that observations are drawn from a gaussian distribution with different classes having common covariance matrix. For the datasets where these assumptions are valid, LDA tends to outperform the logistic regression model.
- 5h.
  - KNN is completely non parametric method and does not make any assumption about the distribution, covariance or the shape of the decision boundary. When the decisions boundaries are highly non-linear, KNN often will outpeform LDA and Logistic regression.

6. Q6

- 6a.

```
  games=read.csv("http://statweb.stanford.edu/~jgorham/games.csv", as.is=
TRUE)
  teams=read.csv("http://statweb.stanford.edu/~jgorham/teams.csv", as.is=
TRUE)
  all.teams=sort(unique(c(teams$team,games$home,games$away)))

  #ii = names(games) %in% c('home','homeScore')
  #head(games)[,ii]

  ##Function to compute teams total margin of victory
  total.margin = function(team){
  with(games,
     sum(homeScore[home==team])+
     sum(awayScore[away==team])-
     sum(homeScore[away==team])-
     sum(awayScore[home==team]))
  }

#Function to compute the humber of games a team played
number.games=function(team){
  with(games,
   sum(home==team)+sum(away==team))
}


y= with(games, homeScore-awayScore)
X0 = as.data.frame(matrix(0,nrow(games),length(all.teams)))
names(X0)=all.teams

for(tm in all.teams){
  X0[[tm]]=1*(games$home==tm)-1*(games$away==tm)

}

X=X0[,names(X0) !="stanford-cardinal"]
reg.season.games=which(games$gameType=="REG")
lm.fit=lm(y~0+.,data=X,subset=reg.season.games)

homeAdv=1-games$neutralLocation
Xh=cbind(homeAdv=homeAdv,X)
lm.fit.homeAdv=lm(y~0+.,data=Xh, subset=reg.season.games)
#head(coef(summary(lm.fit.homeAdv)),1)
#lmrank=coef(summary(lm.fit.homeAdv))[,1]
#rank.table.lm=cbind("Linear Reg Estimate" = lmrank,
#                    "Linear Reg Rank" = rank(-lmrank,ties="min"))
```

```
#lm.top25=order(lmrank, decreasing="TRUE")[1:25]
#rank.table.lm[lm.top25,]


y.win=with(games, homeScore-awayScore>0)
y.win=y.win+0;
glm.fit.ncaa=glm(y.win~0+.,data=Xh, subset=reg.season.games, family=binom
ial)
head(coef(summary(glm.fit.ncaa)))

##                               Estimate Std. Error      z value     Pr(>|
z|)
## homeAdv                     0.679812227 0.04031881 16.860919164 8.722200e
-64
## `air-force-falcons`         0.117271169 0.70171078  0.167121800 8.672742e
-01
## `akron-zips`                0.228890502 0.73602968  0.310979989 7.558158e
-01
## `alabama-a&m-bulldogs`     -4.576626969 0.83025138 -5.512338897 3.540963e
-08
## `alabama-crimson-tide`     -0.004102928 0.66055210 -0.006211361 9.950441e
-01
## `alabama-state-hornets`    -4.590445856 0.77544844 -5.919730632 3.224693e
-09

#coef(summary(glm.fit.ncaa))
```

- saint mary-saint-mary has high coeff of 14.13 with p value 0.9. Saint-Mary'won a lot of games but the margin of most of the victories was fairly narrow. Hence, with the logisitic regression model where we give importance to W/L record, Saint Mary's stats look very good.

- saint-thomas has 13.27 pvalue .9. They have a high score, because they played only 1 away game and won that game.

• 6b.

```
X0play = as.data.frame(matrix(NA,1,length(all.teams)))
names(X0play)=all.teams

i=1
for(tm in all.teams){
 X0play[i]=sum(games$home==tm)+sum(games$away==tm)
 i=i+1
}

X0play.5=X0play[which(X0play[]>5)]
X05 = as.data.frame(matrix(0,nrow(games),ncol(X0play.5)))
names(X05)=names(X0play.5)
```

```
  for(tm in names(X0play.5)){
    X05[[tm]]=1*(games$home==tm)-1*(games$away==tm)

  }

  X5=X05[,names(X05) !="stanford-cardinal"]
  reg.season.games=which(games$gameType=="REG")
  homeAdv=1-games$neutralLocation
  Xh5=cbind(homeAdv=homeAdv,X5)

  lm.fit.ncaa5=glm(y~0+.,data=Xh5, subset=reg.season.games)

  lmrank=coef(summary(lm.fit.ncaa5))[,1]
  rank.table.lm=cbind("Linear Reg Estimate" = lmrank,
                      "Linear Reg Rank" = rank(-lmrank,ties="min"),
                      "AP Rank" = teams$apRank,
                      "USAT Rank" =teams$usaTodayRank)

  lm.top25=order(lmrank, decreasing="TRUE")[1:25]
  rank.table.lm[lm.top25,]
```

```
##                               Linear Reg Estimate Linear Reg Rank AP Ra
nk
## `indiana-hoosiers`                       39.52368               1
NA
## `florida-gators`                         38.94581               2
NA
## `louisville-cardinals`                   38.66837               3
NA
## `gonzaga-bulldogs`                       36.18089               4
NA
## `duke-blue-devils`                       35.75218               5
NA
## `kansas-jayhawks`                        34.69556               6
NA
## `ohio-state-buckeyes`                    34.03453               7
NA
## `pittsburgh-panthers`                    33.96048               8
NA
## `michigan-wolverines`                    33.72313               9
NA
## `syracuse-orange`                        33.51734              10
NA
## `wisconsin-badgers`                      32.97778              11
NA
## `michigan-state-spartans`                32.34475              12
NA
## `creighton-bluejays`                     31.72288              13
23
## `virginia-commonwealth-rams`             31.57597              14
```

```
NA
## `miami-(fl)-hurricanes`                    31.55919              15
NA
## `georgetown-hoyas`                         30.70581              16
9
## `oklahoma-state-cowboys`                   30.00111              17
NA
## `minnesota-golden-gophers`                 29.76057              18
NA
## `saint-mary's-gaels`                       29.56983              19
NA
## `missouri-tigers`                          29.53314              20
NA
## `colorado-state-rams`                      29.40677              21
2
## `saint-louis-billikens`                    29.15598              22
NA
## `north-carolina-tar-heels`                 29.10414              23
NA
## `new-mexico-lobos`                         29.07901              24
NA
## `ole-miss-rebels`                          29.06631              25
NA
##                               USAT Rank
## `indiana-hoosiers`                   NA
## `florida-gators`                     NA
## `louisville-cardinals`               NA
## `gonzaga-bulldogs`                   NA
## `duke-blue-devils`                   NA
## `kansas-jayhawks`                    NA
## `ohio-state-buckeyes`                NA
## `pittsburgh-panthers`                NA
## `michigan-wolverines`                NA
## `syracuse-orange`                    NA
## `wisconsin-badgers`                  NA
## `michigan-state-spartans`            NA
## `creighton-bluejays`                 NA
## `virginia-commonwealth-rams`         NA
## `miami-(fl)-hurricanes`              NA
## `georgetown-hoyas`                    9
## `oklahoma-state-cowboys`             NA
## `minnesota-golden-gophers`           NA
## `saint-mary's-gaels`                 NA
## `missouri-tigers`                    NA
## `colorado-state-rams`                 2
## `saint-louis-billikens`              NA
## `north-carolina-tar-heels`           NA
## `new-mexico-lobos`                   NA
## `ole-miss-rebels`                    NA
```

```
  glm.fit.ncaa5=glm(y.win~0+.,data=Xh5, subset=reg.season.games, family=b
inomial)
  #head(coef(summary(glm.fit.ncaa5)))
  glmrank=coef(summary(glm.fit.ncaa5))[,1]
  rank.table.glm=cbind("Log Reg Estimate" = glmrank,
                       "Log Reg Rank" = rank(-glmrank,ties="min"),
                       "AP Rank" = teams$apRank,
                       "USAT Rank" =teams$usaTodayRank)

  glm.top25=order(glmrank, decreasing="TRUE")[1:25]
  rank.table.glm[glm.top25,]
```

```
##                              Log Reg Estimate Log Reg Rank AP Rank
## `gonzaga-bulldogs`                   5.942567            1      NA
## `louisville-cardinals`               5.591358            2      NA
## `kansas-jayhawks`                    5.403303            3      NA
## `indiana-hoosiers`                   5.373546            4      NA
## `new-mexico-lobos`                   5.353893            5      NA
## `duke-blue-devils`                   5.273410            6      NA
## `ohio-state-buckeyes`                5.246121            7      NA
## `georgetown-hoyas`                   5.185154            8       9
## `michigan-state-spartans`            5.092454            9      NA
## `michigan-wolverines`                5.079822           10      NA
## `miami-(fl)-hurricanes`              4.916801           11      NA
## `kansas-state-wildcats`              4.902514           12      NA
## `syracuse-orange`                    4.777640           13      NA
## `memphis-tigers`                     4.721238           14      NA
## `saint-louis-billikens`              4.689980           15      NA
## `marquette-golden-eagles`            4.673286           16      NA
## `butler-bulldogs`                    4.640346           17      NA
## `wisconsin-badgers`                  4.554807           18      NA
## `oklahoma-state-cowboys`             4.459630           19      NA
## `florida-gators`                     4.453179           20      NA
## `pittsburgh-panthers`                4.445350           21      NA
## `notre-dame-fighting-irish`          4.425768           22      NA
## `unlv-rebels`                        4.362772           23      NA
## `colorado-state-rams`                4.304805           24       2
## `north-carolina-tar-heels`           4.224917           25      NA
##                              USAT Rank
## `gonzaga-bulldogs`                  NA
## `louisville-cardinals`              NA
## `kansas-jayhawks`                   NA
## `indiana-hoosiers`                  NA
## `new-mexico-lobos`                  NA
## `duke-blue-devils`                  NA
## `ohio-state-buckeyes`               NA
## `georgetown-hoyas`                   9
## `michigan-state-spartans`           NA
## `michigan-wolverines`               NA
## `miami-(fl)-hurricanes`             NA
```

```
## `kansas-state-wildcats`                NA
## `syracuse-orange`                      NA
## `memphis-tigers`                       NA
## `saint-louis-billikens`                NA
## `marquette-golden-eagles`              NA
## `butler-bulldogs`                      NA
## `wisconsin-badgers`                    NA
## `oklahoma-state-cowboys`               NA
## `florida-gators`                       NA
## `pittsburgh-panthers`                  NA
## `notre-dame-fighting-irish`            NA
## `unlv-rebels`                          NA
## `colorado-state-rams`                   2
## `north-carolina-tar-heels`             NA
```

- Both linear regression and logistic regression does not have matching ranking with AP and USA rankings. Linear regression does slightly better than logistic regression with 1 additional prediction in the top-25 that also has a top 25 ranking in AP and USAT ranking.

- 6c.

```
u=which(coef(summary(lm.fit.ncaa5))[,4]<0.05)
#coef(summary(lm.fit.ncaa5))[u,]
nrow(coef(summary(glm.fit.ncaa))[u,])

## [1] 318

k=which(coef(summary(glm.fit.ncaa5))[,4]<0.05)
#coef(summary(glm.fit.ncaa5))[k,]
nrow(coef(summary(glm.fit.ncaa))[k,])

## [1] 216
```

- With linear regerssion 318/406= 78% of entries have p value <0.05
- With logistic regression 216/406= 53% of entries have p value <0.05

- 6d.
- 6e