Lab: Logistic Regression, LDA, QDA, and KNN

Jonathan Bryan

April 19, 2018

4.6.1 The Stock Market Data

```
library(ISLR)
names(Smarket)
```

```
## [1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
## [7] "Volume"    "Today"     "Direction"
```

```
dim(Smarket)
```

```
## [1] 1250    9
```

```
summary(Smarket)
```

```
##       Year           Lag1                Lag2
##  Min.   :2001   Min.   :-4.922000   Min.   :-4.922000
##  1st Qu.:2002   1st Qu.:-0.639500   1st Qu.:-0.639500
##  Median :2003   Median : 0.039000   Median : 0.039000
##  Mean   :2003   Mean   : 0.003834   Mean   : 0.003919
##  3rd Qu.:2004   3rd Qu.: 0.596750   3rd Qu.: 0.596750
##  Max.   :2005   Max.   : 5.733000   Max.   : 5.733000
##       Lag3                Lag4                Lag5
##  Min.   :-4.922000   Min.   :-4.922000   Min.   :-4.92200
##  1st Qu.:-0.640000   1st Qu.:-0.640000   1st Qu.:-0.64000
##  Median : 0.038500   Median : 0.038500   Median : 0.03850
##  Mean   : 0.001716   Mean   : 0.001636   Mean   : 0.00561
##  3rd Qu.: 0.596750   3rd Qu.: 0.596750   3rd Qu.: 0.59700
##  Max.   : 5.733000   Max.   : 5.733000   Max.   : 5.73300
##      Volume           Today              Direction
##  Min.   :0.3561   Min.   :-4.922000   Down:602
##  1st Qu.:1.2574   1st Qu.:-0.639500   Up  :648
##  Median :1.4229   Median : 0.038500
##  Mean   :1.4783   Mean   : 0.003138
##  3rd Qu.:1.6417   3rd Qu.: 0.596750
##  Max.   :3.1525   Max.   : 5.733000
```

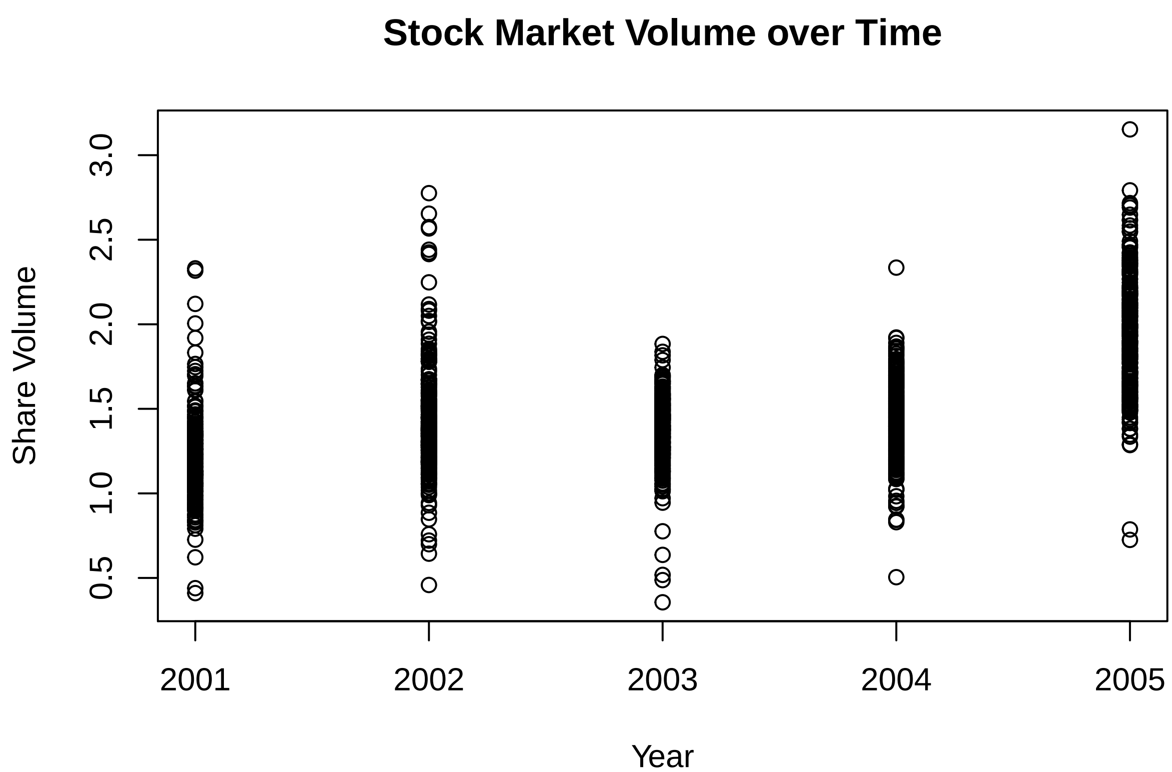
Correlation matrix

```
round(cor(Smarket[,-9]),3)
```

```
##         Year   Lag1   Lag2   Lag3   Lag4   Lag5 Volume  Today
## Year   1.000  0.030  0.031  0.033  0.036  0.030  0.539  0.030
## Lag1   0.030  1.000 -0.026 -0.011 -0.003 -0.006  0.041 -0.026
## Lag2   0.031 -0.026  1.000 -0.026 -0.011 -0.004 -0.043 -0.010
## Lag3   0.033 -0.011 -0.026  1.000 -0.024 -0.019 -0.042 -0.002
## Lag4   0.036 -0.003 -0.011 -0.024  1.000 -0.027 -0.048 -0.007
## Lag5   0.030 -0.006 -0.004 -0.019 -0.027  1.000 -0.022 -0.035
## Volume 0.539  0.041 -0.043 -0.042 -0.048 -0.022  1.000  0.015
## Today  0.030 -0.026 -0.010 -0.002 -0.007 -0.035  0.015  1.000
```

Plot Volume and Year

```r
plot(x=Smarket$Year, y=Smarket$Volume,
     main = "Stock Market Volume over Time",
     xlab = "Year",
     ylab = "Share Volume")
```



Logistic Regression

```r
glm.fit = glm(Direction ~ . -Year -Today, data = Smarket, family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ . - Year - Today, family = binomial,
##     data = Smarket)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.446  -1.203   1.065   1.145   1.326
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.126000   0.240736  -0.523    0.601
## Lag1        -0.073074   0.050167  -1.457    0.145
## Lag2        -0.042301   0.050086  -0.845    0.398
## Lag3         0.011085   0.049939   0.222    0.824
## Lag4         0.009359   0.049974   0.187    0.851
```

```
## Lag5         0.010313   0.049511   0.208    0.835
## Volume       0.135441   0.158360   0.855    0.392
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1731.2  on 1249  degrees of freedom
## Residual deviance: 1727.6  on 1243  degrees of freedom
## AIC: 1741.6
##
## Number of Fisher Scoring iterations: 3
```

Accessing Coefficients

```r
coef(glm.fit)
```

```
##  (Intercept)         Lag1         Lag2         Lag3         Lag4
## -0.126000257 -0.073073746 -0.042301344  0.011085108  0.009358938
##         Lag5       Volume
##  0.010313068  0.135440659
```

```r
summary(glm.fit)$coef
```

```
##                 Estimate Std. Error    z value  Pr(>|z|)
## (Intercept) -0.126000257 0.24073574 -0.5233966 0.6006983
## Lag1        -0.073073746 0.05016739 -1.4565986 0.1452272
## Lag2        -0.042301344 0.05008605 -0.8445733 0.3983491
## Lag3         0.011085108 0.04993854  0.2219750 0.8243333
## Lag4         0.009358938 0.04997413  0.1872757 0.8514445
## Lag5         0.010313068 0.04951146  0.2082966 0.8349974
## Volume       0.135440659 0.15835970  0.8552723 0.3924004
```

```r
summary(glm.fit)$coef[,4] #p-values
```

```
## (Intercept)        Lag1        Lag2        Lag3        Lag4        Lag5
##   0.6006983   0.1452272   0.3983491   0.8243333   0.8514445   0.8349974
##      Volume
##   0.3924004
```

Logistic Regression Prediction

```r
glm.probs=predict(glm.fit, type = "response") #Use training data for predictions
round(glm.probs[1:10],3)
```

```
##     1     2     3     4     5     6     7     8     9    10
## 0.507 0.481 0.481 0.515 0.511 0.507 0.493 0.509 0.518 0.489
```

```r
contrasts(Smarket$Direction)
```

```
##      Up
## Down  0
## Up    1
```

Converting Probabilites into Direction

```r
glm.pred = rep("Down", 1250)
glm.pred[glm.probs > .5] = "Up"
```

Confusion Matrix

```
table(glm.pred, Smarket$Direction)
```

```
##
## glm.pred Down  Up
##     Down  145 141
##     Up    457 507
```

```
(145 +507)/1250
```

```
## [1] 0.5216
```

```
mean(glm.pred == Smarket$Direction) #Accuracy
```

```
## [1] 0.5216
```

Cross Validation

```
train = Smarket$Year < 2005
Smarket.2005 = Smarket[!train,]
dim(Smarket.2005)
```

```
## [1] 252   9
```

```
Direction.2005 = Smarket$Direction[!train]

#Train new model
glm.fit = glm(Direction ~ . -Year -Today, family = binomial, data = Smarket, subset = train)
glm.probs = predict(glm.fit, newdata = Smarket.2005, type = "response")

#New predictions
glm.pred = rep("Down", nrow(Smarket.2005))
glm.pred[glm.probs > 0.5] = "Up"
table(glm.pred, Direction.2005)
```

```
##         Direction.2005
## glm.pred Down Up
##     Down   77 97
##     Up     34 44
```

```
mean(glm.pred == Direction.2005) #Test set accuracy
```

```
## [1] 0.4801587
```

Refit the Logistical Regression with only Lag1 and Lag2

```
glm.fit = glm(Direction ~ Lag1 + Lag2, family = binomial, data = Smarket, subset = train)
glm.probs = predict(glm.fit, newdata = Smarket.2005, type = "response")
glm.pred = rep("Down" , nrow(Smarket.2005))
glm.pred[glm.probs > 0.5] = "Up"
table(glm.pred, Direction.2005)
```

```
##         Direction.2005
## glm.pred Down  Up
##     Down   35  35
##     Up     76 106
```

```
mean(glm.pred == Direction.2005) #Test set accuracy
```

```
## [1] 0.5595238
```

Prediction for Specific Values

```r
predict(glm.fit, newdata = data.frame(Lag1 = c(1.2,1.5), Lag2 = c(1.1,-0.8)), type = "response")
```

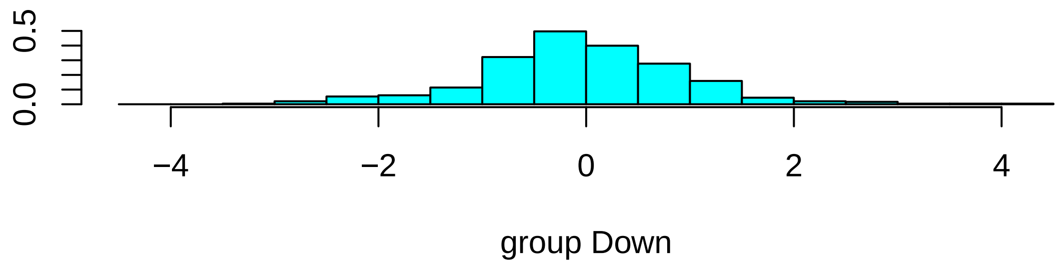
```
##         1         2
## 0.4791462 0.4960939
```

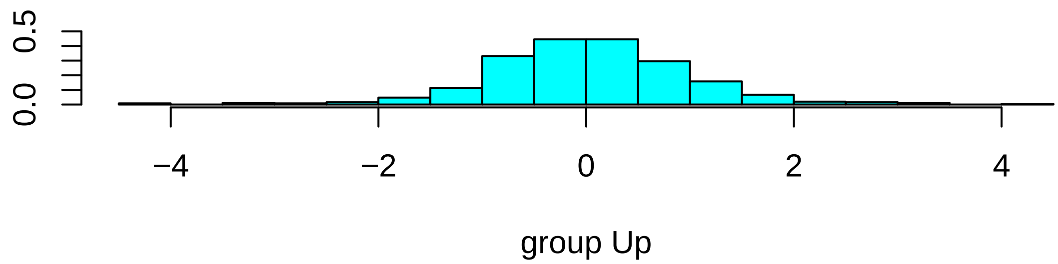
4.6.3 Linear Discriminant Analysis

```r
library(MASS)
lda.fit = lda(Direction ~ Lag1 + Lag2, data= Smarket, subset = train)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag1 + Lag2, data = Smarket, subset = train)
##
## Prior probabilities of groups:
##     Down       Up
## 0.491984 0.508016
##
## Group means:
##             Lag1        Lag2
## Down  0.04279022  0.03389409
## Up   -0.03954635 -0.03132544
##
## Coefficients of linear discriminants:
##             LD1
## Lag1 -0.6420190
## Lag2 -0.5135293
```

```r
plot(lda.fit, main = "LDA Estimation")
```





LDA Prediction

```
lda.pred = predict(lda.fit, newdata = Smarket.2005)
names(lda.pred)
```

```
## [1] "class"     "posterior" "x"
```

```
lda.class = lda.pred$class
table(lda.class, Direction.2005)
```

```
##          Direction.2005
## lda.class Down  Up
##      Down   35  35
##      Up     76 106
```

```
mean(lda.class == Direction.2005) #Accuracy
```

```
## [1] 0.5595238
```

Changing LDA Thresholds

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.4.4
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
```

```
##
##     between, first, last
## The following object is masked from 'package:MASS':
##
##     select
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#Notice that the postive posterior probability corresponds to the null or "Down" days
sum(lda.pred$posterior[,1] >=0.5)
```

```
## [1] 70
```

```
sum(lda.pred$posterior[,1] < 0.5)
```

```
## [1] 182
```

```
df = data.frame(PostProb = round(lda.pred$posterior[1:20,1],3), Class = lda.class[1:20])
df_t = transpose(df)
colnames(df_t) = rownames(df)
rownames(df_t) = colnames(df)
knitr::kable(df_t[,1:10])
```

	999	1000	1001	1002	1003	1004	1005	1006	1007	1008
PostProb	0.49	0.479	0.467	0.474	0.493	0.494	0.495	0.487	0.491	0.484
Class	Up	Up	Up	Up	Up	Up	Up	Up	Up	Up

```
knitr::kable(df_t[,11:20])
```

	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018
PostProb	0.491	0.512	0.49	0.471	0.474	0.48	0.494	0.503	0.498	0.489
Class	Up	Down	Up	Up	Up	Up	Up	Down	Up	Up

Highest Predicted Probability is Low

```
sum(lda.pred$posterior[,1] > .90)
```

```
## [1] 0
```

```
max(lda.pred$posterior)
```

```
## [1] 0.5422133
```

4.6.4 Quadratic Discriminant Analysis

```
qda.fit = qda(Direction ~ Lag1 + Lag2, data= Smarket, subset = train)
qda.fit
```

```
## Call:
## qda(Direction ~ Lag1 + Lag2, data = Smarket, subset = train)
##
## Prior probabilities of groups:
##     Down       Up
## 0.491984 0.508016
##
## Group means:
##             Lag1        Lag2
## Down  0.04279022  0.03389409
## Up   -0.03954635 -0.03132544
```

QDA Prediction

```r
qda.class = predict(qda.fit, newdata=Smarket.2005, type = "response")$class
table(qda.class, Direction.2005)
```

```
##          Direction.2005
## qda.class Down  Up
##      Down   30  20
##      Up     81 121
```

```r
mean(qda.class == Direction.2005)
```

```
## [1] 0.5992063
```

4.6.5 K-Nearest Neighbors

```r
#Set up KNN data
library(class)
train.X = Smarket[train,c("Lag1","Lag2")]
test.X = Smarket.2005[,c("Lag1","Lag2")]
train.Direction = Smarket[train,]$Direction
```

```r
#KNN model where K=1 (very flexible)
set.seed(1)
knn.pred=knn(train.X,test.X,train.Direction,k=1)
table(knn.pred, Direction.2005)
```

```
##         Direction.2005
## knn.pred Down Up
##     Down   43 58
##     Up     68 83
```

```r
mean(knn.pred == Direction.2005) #Accuracy
```

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

```r
#KNN model where K=3 (less flexible)
knn.pred=knn(train.X,test.X,train.Direction,k=3)
table(knn.pred, Direction.2005)
```

```
##         Direction.2005
## knn.pred Down Up
##     Down   48 54
##     Up     63 87
```

```r
mean(knn.pred == Direction.2005) #Accuracy
```

```
## [1] 0.5357143
```

4.6.6 An Application to Caravan Insurance Data

```r
dim(Caravan)
```

```
## [1] 5822   86
```

```r
summary(Caravan$Purchase)
```

```
##   No  Yes
## 5474  348
```

```r
348/nrow(Caravan)
```

```
## [1] 0.05977327
```

```r
set.seed(1)
#KNN is sensitive to scale so we standardize the data
standardized.X = scale((Caravan[,-86]))
test = 1:1000
train.X = standardized.X[-test,]
test.X = standardized.X[test,]
train.Y = Caravan$Purchase[-test]
test.Y = Caravan$Purchase[test]

#Fit the new model
knn.pred = knn(train.X, test.X, train.Y, k=1)
mean(test.Y != knn.pred) #Error rate
```

```
## [1] 0.118
```

```r
mean(test.Y != "No") #Empirical rate for purchased insurance
```

```
## [1] 0.059
```

```r
table(knn.pred, test.Y)
```

```
##         test.Y
## knn.pred  No Yes
##      No  873  50
##      Yes  68   9
```

```r
9/(68+9) #Sensitivity
```

```
## [1] 0.1168831
```

```r
#Decreasing KNN flexibility gives us higher sensitivity
knn.pred = knn(train.X,test.X,train.Y,k=3)
table(knn.pred,test.Y)
```

```
##         test.Y
## knn.pred  No Yes
##      No  920  54
##      Yes  21   5
```

```r
table(knn.pred,test.Y)[2,2]/(table(knn.pred,test.Y)[2,1]+table(knn.pred,test.Y)[2,2])
```

```
## [1] 0.1923077
```

```r
knn.pred = knn(train.X,test.X,train.Y,k=5)
table(knn.pred,test.Y)
```

```
##          test.Y
## knn.pred  No Yes
##      No  930  55
##      Yes  11   4
```

```r
table(knn.pred,test.Y)[2,2]/(table(knn.pred,test.Y)[2,1]+table(knn.pred,test.Y)[2,2])
```

```
## [1] 0.2666667
```

```r
#Logistic regression comparison (0.5 cutoff)
glm.fit = glm(Purchase ~ ., family = binomial, data = Caravan[-test,])
glm.probs = predict(glm.fit, type = "response", newdata = Caravan[test,])
glm.pred = rep("No",1000)
glm.pred[glm.probs > 0.5] = "Yes"
table(glm.pred, test.Y)
```

```
##          test.Y
## glm.pred  No Yes
##      No  934  59
##      Yes   7   0
```

```r
table(glm.pred, test.Y)[2,2] / (table(glm.pred, test.Y)[2,1] + table(glm.pred, test.Y)[2,2])
```

```
## [1] 0
```

```r
#Logistic regression comparison (0.25) cutoff)
glm.pred[glm.probs > 0.25] = "Yes"
table(glm.pred, test.Y)
```

```
##          test.Y
## glm.pred  No Yes
##      No  919  48
##      Yes  22  11
```

```r
table(glm.pred, test.Y)[2,2] / (table(glm.pred, test.Y)[2,1] + table(glm.pred, test.Y)[2,2])
```

```
## [1] 0.3333333
```