4 Logistic Regression

Notes

$\log(\frac{p(X)}{(1-p(X))}) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$

- left-hand side is log odds
- uses maximum likelihood to estimate coefficients
- one unit increase in an independent variable is associated with an increase in the log odds of the variable by its coefficient

Linear Discriminant Analysis (LDA)

- popular used when more than 2 response classes
- if n is small and X is approximately normal
- uses Bayes Theorem to estimate $Pr(Y = k|X = x)$
- $Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$
  - where there are K classes
  - where π_k is prior probability that a randomly chosen observation comes from the kth class
  - where $f_k(x) = Pr(X = x|Y = y)$ is the density function of X from the kth class
- assumes predictor variables come from normal (or multivariate normal) distribution
- class-specific mean vector and covariance matrix that is common to all K classes
- can modify threshold of boundary decisions
- Confusion Matrix used to count number of correctly/incorrectly predicted outcomes

Quadratic Discriminant Analysis (QDA)

- also assumes observations from each class are normally distributed
- assumes each class has its own covariance matrix
- more flexible classifier than LDA
- recommended if training set is very large

****Logistic and LDA both produce linear decision boundaries, while QDA and KNN classifiers have higher flexibility and lower bias**

Applied

10)

```
library(ISLR)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------
```

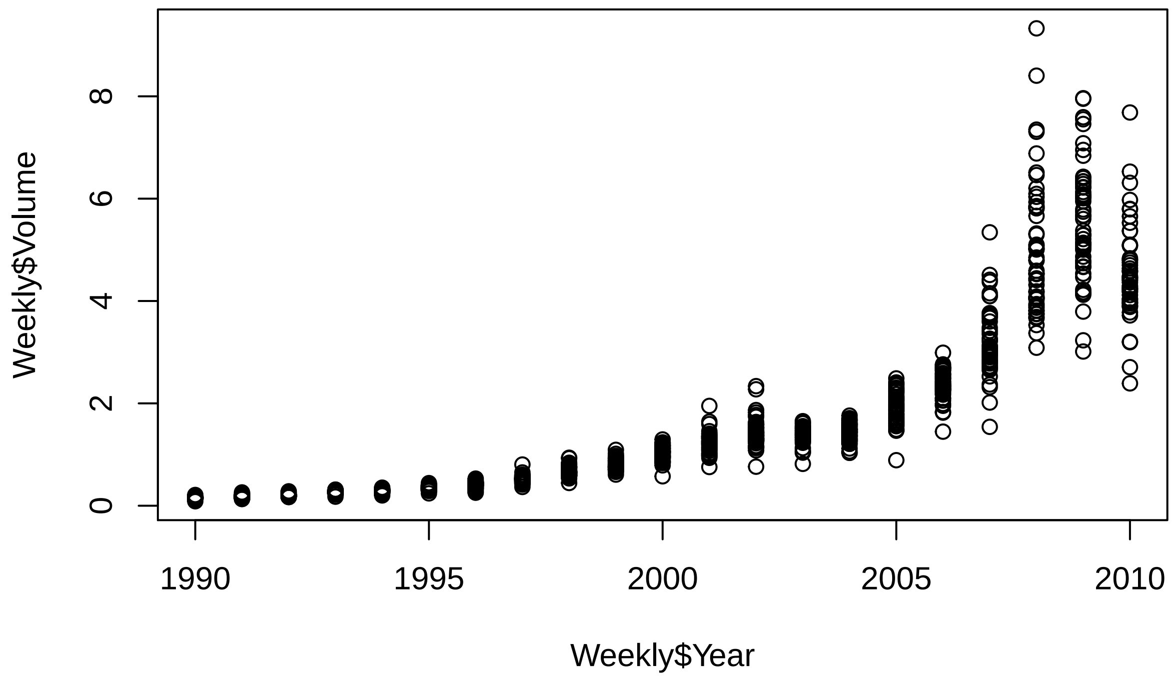
```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts -----------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
#plot(Weekly) #there seems to be a noticeable relationship between Volume and Year
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
```

```
plot(Weekly$Year, Weekly$Volume)
```




```r
#Logistic Regression
week.glm <- glm(Direction~. - Year - Today, data = Weekly, family = "binomial"); summary(week.glm)
```

```
##
## Call:
## glm(formula = Direction ~ . - Year - Today, 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
```

```r
week.probs <- predict(week.glm, type = "response")
week.pred <- rep("Down", times = nrow(Weekly)) #create vector of # of down elements = Weekly rows
week.pred[week.probs > .5] = "Up" #transform elements to up for which the corresponding prob. is >.5,
week.cm <- table(week.pred, Weekly$Direction) #confusion matrix
```

It seems that only Lag2 seems significant

```r
week.cm
```

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

```r
#correct classifications
week.correct <- sum(diag(week.cm))/sum(week.cm); week.correct
```

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

```r
#incorrect classifications
week.incorrect <- sum(diag(week.cm[nrow(week.cm):1,]))/sum(week.cm); week.incorrect
```

```
## [1] 0.4389348
```

The overall fraction of correct predictions is about .561

```r
#train/test using Lag2 as only predictor
train <- (Weekly$Year < 2009) #years before 2008 are set to TRUE, while after set to FALSE
test <- Weekly[!train,]
direction <- Weekly$Direction[!train] #true response values used to compare to test data

week.fit <- glm(Direction~Lag2, data = Weekly, family = "binomial", subset = train); summary(week.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2, family = "binomial", data = Weekly,
##     subset = train)
##
## 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
```

```
week.probs2 <- predict(week.fit, test, type = "response")

week.pred2 <- rep("Down", nrow(test))
week.pred2[week.probs2 > .5] <- "Up"
week.cm2 <- table(week.pred2, direction)
week.correct2 <- sum(diag(week.cm2))/sum(week.cm2); week.correct2
```

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

The correct rate of this model is .625, which is slightly better than the model with all variables

```
#LDA
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

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

```r
lda.pred <- predict(lda.fit, test)

lda.class <- lda.pred$class

table(lda.class, direction)
```

```
##          direction
## lda.class Down Up
##      Down    9  5
##      Up     34 56
```

```r
mean(lda.class == direction)
```

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

```r
#QDA

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

```
## Call:
## qda(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
```

```r
qda.class <- predict(qda.fit,test)$class
table(qda.class,direction)
```

```
##          direction
## qda.class Down Up
##      Down    0  0
##      Up     43 61
```

```r
mean(qda.class == direction)
```

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

```r
#KNN
library(class)

train.k <- Weekly[train, c("Lag2","Direction")]
test.k <- Weekly[!train, c("Lag2","Direction")]

set.seed(1)
```

```r
knn.pred <- knn(train = data.frame(train.k$Lag2),test = data.frame(test.k$Lag2), train.k$Direction, k =

table(knn.pred, direction)
```

```
##          direction
## knn.pred Down Up
##     Down   21 30
##     Up     22 31
```

```r
mean(knn.pred == direction)
```

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

It seems that LDA and Logistic performed the best