Sam Lin
Stats 202
Hw. 2

1. (Ex. 4 Pg. 168)

a) To predict 10% of x, x must be an element of [0.05, 0.95], which can be rewritten as [x-0.05, x+0.05]. However if x < 0.05, then we use observations in interval [0, x+0.05] which represents (100x+5)% and if x > 0.95, then we use observations in interval [x-0.05, 100], which represents (105-100x)%.

$$\int_0^{0.05} (100x+5)\,dx + \int_{0.05}^{0.95} 10\,dx + \int_{0.95}^{1} (105-100x)dx = 9.75$$

b) 0.975 * 0.975 = 0.00950625% assuming the two predictors are independent

c) Assuming the predictors are independent, the fraction of available observations is 0.975^{100}%.

d) As p predictors increase, the fraction of available observations we use for prediction approaches 0, which means nothing to use to train our data.
e) For p = 1, 2, 100, we have length = 0.1, $0.1^{1/2}$, $0.1^{1/100}$

2. (Ex. 6 Pg. 170)

a) Using the logistic model, $p(X) = \frac{e^{-6+0.05X1+X2}}{1+e^{-6+0.05X1+X2}} = 0.3775$
Plug in X1 = 40 hrs. and X2 = 3.75 GPA and you should get the above answer
We use this model, because p(X) = Pr(Y = 1 | X), where Y is getting an A and X = (X1,X2)

b) Given a fixed GPA of 3.5, $\frac{e^{-6+0.05X1+3.5}}{1+e^{-6+0.05X1+3.5}} = 0.5$
Now we solve for X1 by rewriting the equation as

$$e^{-6+0.05X1+3.5} = 0.5 * 1 + 0.5 * e^{-6+0.05X1+3.5}$$
$$0.5 * e^{-6+0.05X1+3.5} = 0.5$$
$$e^{-6+0.05X1+3.5} = 1$$
$$\log(e^{-6+0.05X1+3.5}) = \log(1) \rightarrow -6 + 0.05 * X1 + 3.5 = 0$$
$$0.05 * X1 = 2.5 \rightarrow \mathrm{X1} = 50$$

3. (Ex. 8 Pg. 170)

For K-nearest neighbors with K = 1, we have a training error rate of 0%, because $P(Y = j \mid X = x_i) = I(y_i = j)$. Recall when $y_i = j$, then I = 0, otherwise I = 1. Since K = 1, the training error rate is 0%, because of flexible classification methods. This implies our test error rate is 36% given the average of the test and training was 18%. The test error was greater than the logistic regression 30% error rate, therefore it is better to choose the logistic regression.

4. (Ex. 10 Pg. 171 a, b, c, d)

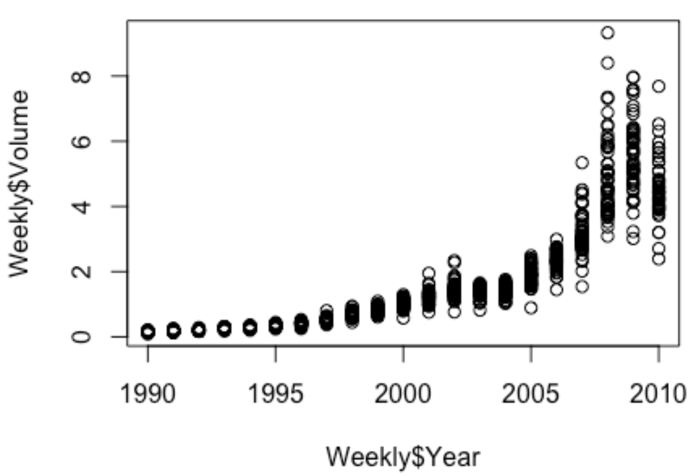
a)

Sam Lin
Stats 202
Hw. 2

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

From this correlation function, we can tell year and volume have the biggest correlation, while correlations between Lags are near zero.


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

b) Must pass argument family = binomial to run a logistic regression

Sam Lin
Stats 202
Hw. 2

```
> 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
```

c) Matrix gives (54+557)/1089 = 0.5651. 1-0.5651 = 0.4389 or 43.89% training error rate. When the market goes up, the model is right 557/(557+48) = 0.921 or 92.1% of the time. When the market goes down, the model is right 54/(54+430) = 0.11157 or 11.16% of the time.

```
> prob<-predict(glm.fit, type = "response") #predict function probabilities
> pred.glm <- rep("Down", length(prob)) #replicates elements with length
> pred.glm[prob > 0.5] <- "Up"
> table(pred.glm, Direction)
        Direction
pred.glm Down  Up
    Down   54  48
    Up    430 557
```

Sam Lin
Stats 202
Hw. 2
d)

```
> train <- (Year < 2009) #from 1990 - 2008
> glm.fit <- glm(Direction ~ Lag2, data = Weekly, family = binomial, subset = train)
> summary(glm.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
```

```
> Weekly.20092010 <- Weekly[!train, ] # for 2009 - 2010
> Direction.20092010 <- Direction[!train]
> probs2 <- predict(glm.fit, Weekly.20092010, type = "response")
> glm.fit<- rep("Down", length(probs2))
> glm.fit[probs2 > 0.5] <- "Up"
> table(glm.fit, Direction.20092010)
       Direction.20092010
glm.fit Down Up
   Down    9  5
   Up     34 56
```

(9+56)/104 = 62.5%, then 1 – 0.625 = 37.5% training error rate. When the market goes up, the model is right 56/(56+5) = 91.8% of the time. When the market goes down, the model is right 9/(34+9) = 20.93% of the time.

5. (Ex. 5 Pg. 198)

a)

Sam Lin
Stats 202
Hw. 2

```
> attach(Default)
> set.seed(1)
> glm.fit <- glm(default ~ income + balance, family = binomial)
> summary(glm.fit)

Call:
glm(formula = default ~ income + balance, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

b) i. & ii. Split data and fit a model with only training data

```
> set.seed(1)
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> glm.fit <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> summary(glm.fit)

Call:
glm(formula = default ~ income + balance, family = "binomial",
    data = Default, subset = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3583  -0.1268  -0.0475  -0.0165   3.8116

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.208e+01  6.658e-01 -18.148   <2e-16 ***
income       1.858e-05  7.573e-06   2.454   0.0141 *
balance      6.053e-03  3.467e-04  17.457   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1457.0  on 4999  degrees of freedom
Residual deviance:  734.4  on 4997  degrees of freedom
AIC: 740.4

Number of Fisher Scoring iterations: 8
```

Sam Lin
Stats 202
Hw. 2
iii. & iv.
```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> #summary(fit.glm)
>
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0268
```
2.68% test error rate with validation set
c)
```
> probs <- predict(glm.fit, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0252
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> glm.fit <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> #summary(fit.glm)
>
> probs <- predict(glm.fit, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0246
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> glm.fit <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> #summary(fit.glm)
>
> probs <- predict(glm.fit, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0266
```
Validation set's test error rate varies depending on what observations are in the training set and what observations in the validation set.

d) Dummy variable student does not seem to affect the reduction of test error rate on the validation set.
```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> glm.fit <- glm(default ~ income + balance + student, data = Default, family = "binomial", subset = train)
> #summary(fit.glm)
>
> probs <- predict(glm.fit, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0254
```

6. (Ex. 6 Pg. 199)
a)

Sam Lin
Stats 202
Hw. 2

```
> glm.fit <- glm(default ~ income + balance, data = Default, family = "binomial")
> summary(glm.fit)

Call:
glm(formula = default ~ income + balance, family = "binomial",
    data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8

> summary(glm.fit)$coefficients[,2]
 (Intercept)       income      balance
4.347564e-01 4.985167e-06 2.273731e-04
```

The coefficients for B_0, B_1, B_2 are 0.4347564, $4.985167*10^{-6}$, and $2.2733731*10^{-4}$

b)

```
> boot.fn <- function(data, index) {
+   fit <- glm(default ~ income + balance, data = data, family = "binomial", subset = index)
+   return(coef(fit))
+ }
```

c)

```
> library(boot)
> boot(Default, boot.fn, 500)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Default, statistic = boot.fn, R = 500)


Bootstrap Statistics :
         original        bias     std. error
t1* -1.154047e+01 -5.655021e-02 4.372395e-01
t2*  2.080898e-05 -5.864198e-08 4.763111e-06
t3*  5.647103e-03  3.287761e-05 2.402848e-04
```

Sam Lin
Stats 202
Hw. 2

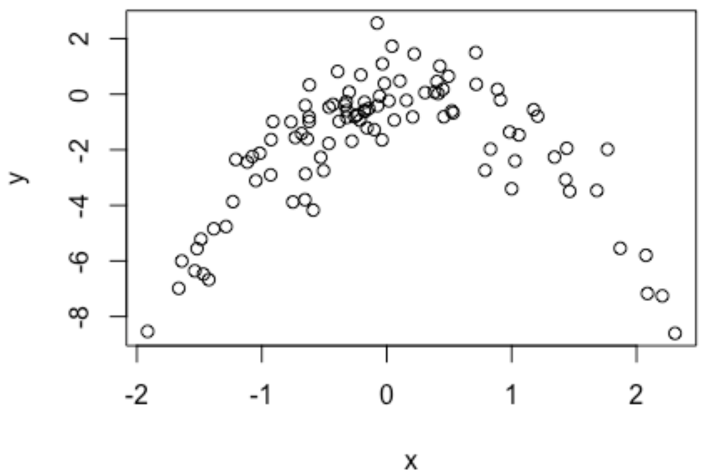
d) Notice from c) that the standard error of for B_0, B_1, B_2 are 0.437239, $4.763111*10^{-6}$ and $2.40284*10^{-4}$ respectively, which is close to the standard error from b) where the coefficients 0.4347564, $4.985167*10^{-6}$, and $2.2733731*10^{-4}$. Therefore we can conclude both estimation methods are pretty close.

7. (Ex. 8 Pg. 200)

a) In the data set, n = 100 observations, and p = 2 predictors

$$y = x - 2x^2 + error$$

b)



There is a curved relationship between x and y.

c)

i.

```
> library(boot)
> set.seed(1)
> Data <- data.frame(x,y)
> fit.glm <- glm(y~x)
> cv.glm(Data, fit.glm)$delta[1]
[1] 5.890979
```

ii.

```
> fit.glm.2 <- glm(y ~ poly(x, 2))
> cv.glm(Data, fit.glm.2)$delta[1]
[1] 1.086596
```

iii.

```
> fit.glm <- glm(y ~ poly(x, 3))
> cv.glm(Data, fit.glm)$delta[1]
[1] 1.102585
```

iv.

Sam Lin
Stats 202
Hw. 2

```
> fit.glm <- glm(y ~ poly(x, 4))
> cv.glm(Data, fit.glm)$delta[1]
[1] 1.114772
```

d) Yes, the results are identical, because leave one out cross validation evaluates n folds of a single observation

```
> set.seed(5)
> Data <- data.frame(x,y)
> fit.glm <- glm(y~x)
> cv.glm(Data, fit.glm)$delta[1]
[1] 5.890979
>
> fit.glm.2 <- glm(y ~ poly(x, 2))
> cv.glm(Data, fit.glm.2)$delta[1]
[1] 1.086596
>
> fit.glm <- glm(y ~ poly(x, 3))
> cv.glm(Data, fit.glm)$delta[1]
[1] 1.102585
>
> fit.glm <- glm(y ~ poly(x, 4))
> cv.glm(Data, fit.glm)$delta[1]
[1] 1.114772
```

e) The second fit had the smallest error, because the relationship between x and y is quadratic as we plotted in b.

f) P-values of linear and quadratic were more significant compared to cubic and quartic, which matches our cross-validation results

```
> summary(fit.glm)

Call:
glm(formula = y ~ poly(x, 4))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8914  -0.5244   0.0749   0.5932   2.7796

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.8277     0.1041 -17.549   <2e-16 ***
poly(x, 4)1   2.3164     1.0415   2.224   0.0285 *
poly(x, 4)2 -21.0586     1.0415 -20.220   <2e-16 ***
poly(x, 4)3  -0.3048     1.0415  -0.293   0.7704
poly(x, 4)4  -0.4926     1.0415  -0.473   0.6373
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 1.084654)

    Null deviance: 552.21  on 99  degrees of freedom
Residual deviance: 103.04  on 95  degrees of freedom
AIC: 298.78

Number of Fisher Scoring iterations: 2
```

Sam Lin
Stats 202
Hw. 2

8. (Ex. 9 Pg. 200)

a)

```
> muest <- mean(medv)
> muest
[1] 22.53281
```

b)

```
> seest <- sd(medv) / sqrt(dim(Boston)[1])
> seest
[1] 0.4088611
```

c)

```
> boot.fn <- function(data, index) {
+   mu <- mean(data[index])
+   return (mu)
+ }
> boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
    original      bias    std. error
t1* 22.53281 -0.01224071   0.4122534
```

Bootstrap estimated standard error is 0.4122534, which is very close to b

d)

```
> t.test(medv)

        One Sample t-test

data:  medv
t = 55.111, df = 505, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 21.72953 23.33608
sample estimates:
mean of x
 22.53281
```

T-test 95% confidence interval: [21.73, 23.33] and bootstrap interval: [21.71, 23.35] are very close.

```
> #22.53 = mean of x and 0.4119 is the standard error
> confI <- c(22.53 - 2 *0.4119, 22.53 + 2*0.4119)
> confI
[1] 21.7062 23.3538
```

Sam Lin
Stats 202
Hw. 2

e)

```
> medest <- median(medv)
> medest
[1] 21.2
```

f)

```
> boot.fn <- function(data, index) {
+   mu <- median(data[index])
+   return (mu)
+ }
> boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*     21.2 -0.0025    0.374358
```

Here we see an estimated 21.2 median value, which is equivalent to the value from e. Also with standard error 0.374 which is small relative to our value.

g)

```
> percentest <- quantile(medv, c(0.1))
> percentest
  10%
12.75
```

h) Similar to h, here we see an estimated tenth percentile value of 12.75, which matches our result from g with standard error approximately 0.5, but small enough to not affect our value.

```
> boot.fn <- function(data, index) {
+   mu <- quantile(data[index], c(0.1))
+   return (mu)
+ }
> boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*    12.75  0.0261   0.4912231
```