Assignment 2 Solution

Q1

(a) Let $Y = 1$ when a student receives an A and $Y = 0$ when doesn't receive an A. We have that

$$\mathbb{P}(Y=1 \mid X) = p(X) = \frac{e^{\hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_0}}{1 + e^{\hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_0}} = \frac{e^{0.05X_1 + X_2 - 6}}{1 + e^{0.05X_1 + X_2 - 6}}$$

Hence, the probability that a student who studies for 40h ($X_1 = 40$) and has an undergrad GPA of 3.5 ($X_2 = 3.5$) gets an A in the class is given by

$$\mathbb{P}(Y=1) = \frac{e^{-0.5}}{1+e^{-0.5}} = 0.378$$

(b) Since $\mathbb{P}(Y=1) = 0.5$, we have that

$$\begin{aligned} \frac{e^{0.05X_1+3.5-6}}{1+e^{0.05X_1+3.5-6}} &= 0.5 \\ e^{0.05X_1-2.5} &= 1 \\ 0.05X_1 - 2.5 &= \ln(1) \\ X_1 &= 50 \end{aligned}$$

Thus, the student in part (a) need to study 70 hours to have a 50% chance of getting an A in the class.

Q2:

(a) If the Bayes decision boundary is linear, we expect QDA to perform better on the training set because it's higher flexiblity will yield a closer fit. On the test set, we expect LDA to perform better than QDA because QDA could overfit the linearity of the Bayes decision boundary.

(b) If the Bayes decision bounary is non-linear, we expect QDA to perform better both on the training and test sets.

(c) We expect the test prediction accuracy of QDA relative to LDA to improve, in general, as the the sample size nn increases because a more flexibile method will yield a better fit as more samples can be fit and variance is offset by the larger sample sizes.

(d) False. With fewer sample points, the variance from using a more flexible method, such as QDA, would lead to overfit, yielding a higher test rate than LDA.

Q4:

(a) $1 - \frac{1}{n}$. The probability of drawing each observation from the original sample are the same.
(b) $1 - \frac{1}{n}$. Since we are sampling with replacement, the probability of drawing each observation from the original sample has no difference between two bootstrap observations.
(c) As we are using sampling with replacement to generate the bootstrap sample, the selection probabilities are independent; so Probability(Observation j is not the first bootstrap observation, Observation j is not the second bootstrap observation, …, Observation j is not the nth bootstrap observation) = Probability(Observation j is not the first bootstrap observation) x Probability(Observation j is not the second bootstrap observation) x … x Probability(Observation j is not the nth bootstrap observation).
(d) *For n* = 5 , $1-(1-1/n) = 1-(1-1/5)^5 = 67.23\%$

Q5:

(a) k-fold cross-validation is implemented by taking the set of n observations and randomly splitting into k non-overlapping groups. Each of these groups acts as a validation set and the remainder as a training set. The test error is estimated by averaging the k resulting MSE estimates.

(b)

i. The validation set approach is conceptually simple and easily implemented as you are simply partitioning the existing training data into two sets. However, there are two drawbacks: (1.) the estimate of the test error rate can be highly variable depending on which observations are included in the training and validation sets. (2.) the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set.

ii. LOOCV is a special case of k-fold cross-validation with k = n. LOOCV has higher variance, but far less bias, than k-fold CV. Additionally, unlike the validation approach, which yields different results when repeated due to randomness in the training/validation set splits, performing LOOCV multiple times always yields the same results because there is no randomness in the training/validation set splits. However, LOOCV is the most computationally intense cross-validation method since the model must be fit n times, which has a big potential to be expensive to implement.

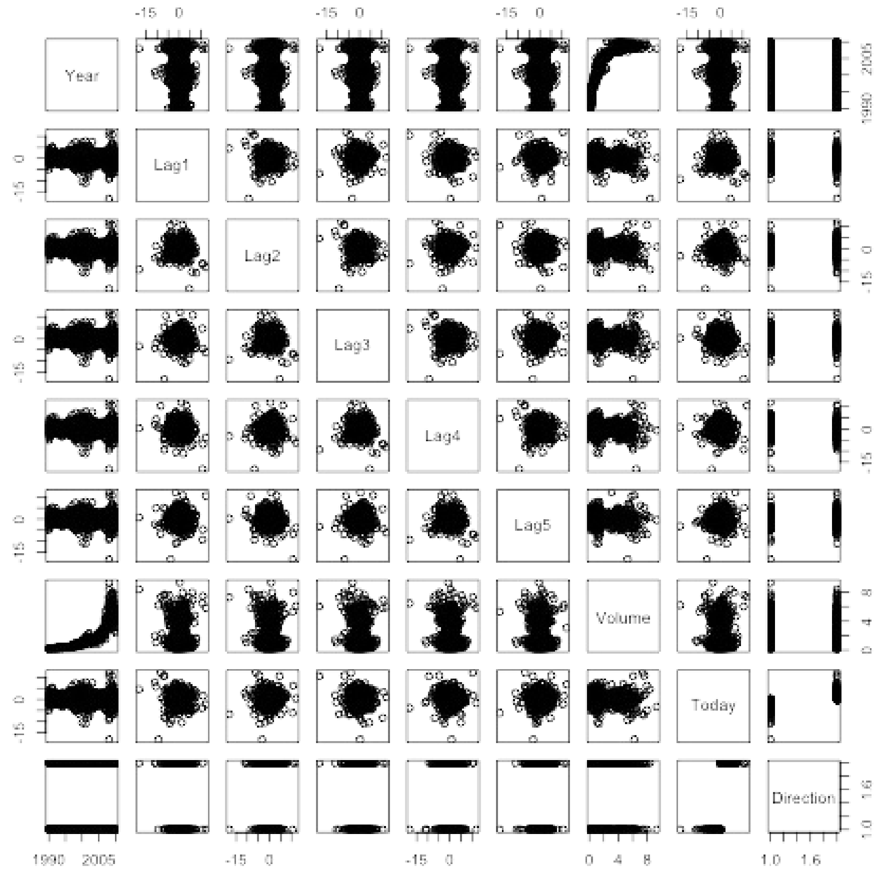
Q3:

(a)

```
library(ISLR)
summary(Weekly)

##       Year           Lag1              Lag2              Lag3
##  Min.   :1990   Min.   :-18.195   Min.   :-18.195   Min.   :-18.195
##  1st Qu.:1995   1st Qu.: -1.154   1st Qu.: -1.154   1st Qu.: -1.158
##  Median :2000   Median :  0.241   Median :  0.241   Median :  0.241
##  Mean   :2000   Mean   :  0.151   Mean   :  0.151   Mean   :  0.147
##  3rd Qu.:2005   3rd Qu.:  1.405   3rd Qu.:  1.409   3rd Qu.:  1.409
##  Max.   :2010   Max.   : 12.026   Max.   : 12.026   Max.   : 12.026
##       Lag4              Lag5              Volume            Today
##  Min.   :-18.195   Min.   :-18.195   Min.   :0.087   Min.:   -18.195
##  1st Qu.: -1.158   1st Qu.: -1.166   1st Qu.:0.332   1st Qu.: -1.154
##  Median :  0.238   Median :  0.234   Median :1.003   Median :  0.241
##  Mean   :  0.146   Mean   :  0.140   Mean   :1.575   Mean   :  0.150
##  3rd Qu.:  1.409   3rd Qu.:  1.405   3rd Qu.:2.054   3rd Qu.:  1.405
##  Max.   : 12.026   Max.   : 12.026   Max.   :9.328   Max.   : 12.026
##  Direction
##  Down:484
##  Up  :605
##

Pairs(Weekly)
```




```
cor(Weekly[,-9])
```

```
##            Year       Lag1      Lag2      Lag3       Lag4       Lag5.     Volume
## Year    1.00000 -0.032289 -0.03339 -0.03001 -0.031128 -0.030519.   0.84194
## Lag1   -0.03229  1.000000 -0.07485  0.05864 -0.071274 -0.008183.  -0.06495
## Lag2   -0.03339 -0.074853  1.00000 -0.07572  0.058382 -0.072499.  -0.08551
## Lag3   -0.03001  0.058636 -0.07572  1.00000 -0.075396  0.060657.  -0.06929
## Lag4   -0.03113 -0.071274  0.05838 -0.07540  1.000000 -0.075675.  -0.06107
## Lag5   -0.03052 -0.008183 -0.07250  0.06066 -0.075675  1.000000.  -0.05852
## Volume  0.84194 -0.064951 -0.08551 -0.06929 -0.061075 -0.058517.   1.00000
## Today  -0.03246 -0.075032  0.05917 -0.07124 -0.007826  0.011013.  -0.03308
##            Today
## Year   -0.032460
## Lag1   -0.075032
## Lag2   -0.059167
## Lag3   -0.071244
## Lag4   -0.007826
## Lag5   -0.011013
## Volume -0.033078
## Today   1.000000
```

Year and Volume appear to have a relationship. No other patterns are discernible.

(b)

```
##
## 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
```

Only predictor Lag2 appears to be statistically significant with its small `p-value = 0.0296`.

(c)

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

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

Percentage of currect predictions: (54+557)/(54+557+48+430) = 56.1%. Weeks the market goes up the logistic regression is right most of the time, 557/(557+48) = 92.1%. Weeks the market goes up the logistic regression is wrong most of the time 54/(430+54) =11.2%.

(d)

```
train = (Year < 2009)
Weekly.0910 = Weekly[!train, ]
glm.fit = glm(Direction ~ Lag2, data = Weekly, family = binomial, subset =
train)
glm.probs = predict(glm.fit, Weekly.0910, type = "response")
glm.pred = rep("Down", length(glm.probs))
glm.pred[glm.probs > 0.5] = "Up"
Direction.0910 = Direction[!train]
table(glm.pred, Direction.0910)

##         Direction.0910
##  glm.pred Down Up
##      Down    9  5
```

```
##        Up   34 56
```

```
mean(glm.pred == Direction.0910)
```

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

(e)

```
library(MASS)
lda.fit = lda(Direction ~ Lag2, data = Weekly, subset = train)
lda.pred = predict(lda.fit, Weekly.0910)
table(lda.pred$class, Direction.0910)
```

```
##       Direction.0910
##        Down Up
##   Down    9  5
##   Up     34 56
```

```
mean(lda.pred$class == Direction.0910)
```

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

(f) By using function `qda()`, we can perform Quadratic Discriminant Analysis to our data set

```
qda.fit = qda(Direction ~ Lag2, data = Weekly, subset = train)
qda.class = predict(qda.fit, Weekly.0910)$class
table(qda.class, Direction.0910)
```

```
##          Direction.2008
## qda.class Down Up
##      Down    0  0
##      Up     43 61
```

```
mean(qda.class == Direction.2008)
```

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

The QDA predictions are only 58.65%, which suggests that the quadratic form assumed by QDA may not capture the true relationship more accurately than the linear forms assumed by LDA and logistic regression.

(g) We perform KNN using the `knn()` function, which is part of `class` library

```
library(class)
train.X = as.matrix(Lag2[train])
test.X = as.matrix(Lag2[!train])
train.Direction = Direction[train]
set.seed(1)
knn.pred = knn(train.X, test.X, train.Direction, k = 1)
table(knn.pred, Direction.0910)
```

```
##         Direction.2008
## knn.pred Down Up
##     Down   21 30
##     Up     22 31
```

```
mean(knn.pred == Direction.2008)
```

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

The results using `K = 1` are not very good, since only 50 % of the observations are correctly predicted. It may be that `K = 1` results in an overly flexible fit to the data.

Question 6

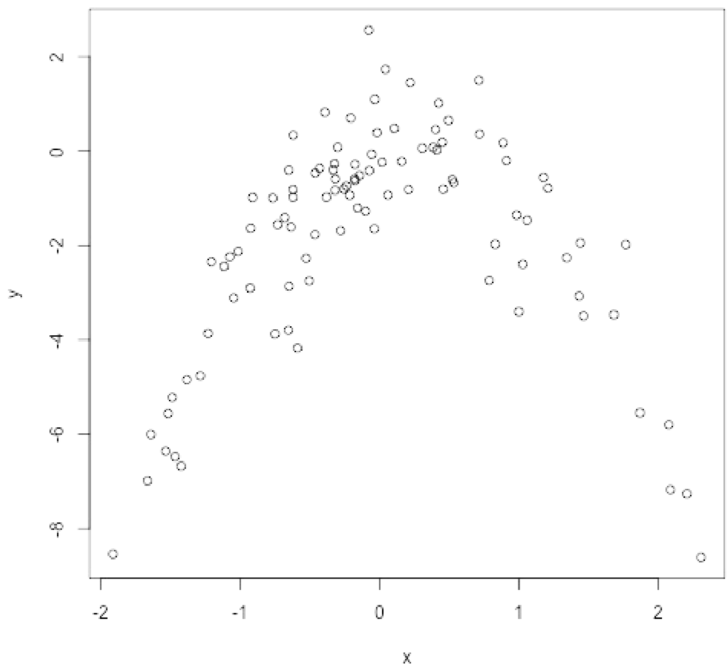
(a)

```
Set.seed(1)
y = rnorm(100)
x = rnorm(100)
y = x - 2 * x^2 + rnorm(100)
```

2 7 . „ 3 7 - 021 * 7 / 6 -/0 5 4

(b)

```
plot(x,y)
```



Quadratic plot. from about -2 to 2. from about -8 to 2.

(c)

```
library(boot)
Data = data.frame(x, y)
set.seed(1)
# i.
glm.fit = glm(y ~ x)
cv.glm(Data, glm.fit)$delta

## [1] 5.891 5.889

# ii.
glm.fit = glm(y ~ poly(x, 2))
cv.glm(Data, glm.fit)$delta

## [1] 1.087 1.086

# iii.
glm.fit = glm(y ~ poly(x, 3))
cv.glm(Data, glm.fit)$delta

## [1] 1.103 1.102

# iv.
glm.fit = glm(y ~ poly(x, 4))
cv.glm(Data, glm.fit)$delta

## [1] 1.115 1.114
```

(d)

```
set.seed(10)
# i.
glm.fit = glm(y ~ x)
cv.glm(Data, glm.fit)$delta

## [1] 5.891 5.889

# ii.
glm.fit = glm(y ~ poly(x, 2))
cv.glm(Data, glm.fit)$delta

## [1] 1.087 1.086

# iii.
glm.fit = glm(y ~ poly(x, 3))
cv.glm(Data, glm.fit)$delta

## [1] 1.103 1.102

# iv.
glm.fit = glm(y ~ poly(x, 4))
cv.glm(Data, glm.fit)$delta

## [1] 1.115 1.114
```

Exact same, because LOOCV will be the same since it evaluates n folds of a single observation.

(e) The quadratic polynomial had the lowest LOOCV test error rate. This was expected because it matches the true form of Y.

(f)

```
summary(glm.fit)

##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8913  -0.5244   0.0749   0.5932   2.7796
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.828      0.104  -17.55   <2e-16 ***
## poly(x, 4)1    2.316      1.041    2.22    0.029 *
## poly(x, 4)2  -21.059      1.041  -20.22   <2e-16 ***
## poly(x, 4)3   -0.305      1.041   -0.29    0.770
## poly(x, 4)4   -0.493.     1.041   -0.47    0.637
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
1
##
## (Dispersion parameter for gaussian family taken to be 1.085)
##
##     Null deviance: 552.21  on 99  degrees of freedom
## Residual deviance: 103.04  on 95  degrees of freedom
## AIC: 298.8
##
## Number of Fisher Scoring iterations: 2
```

p-values show statistical significance of linear and quadratic terms, which agrees with the CV results.

Question 6

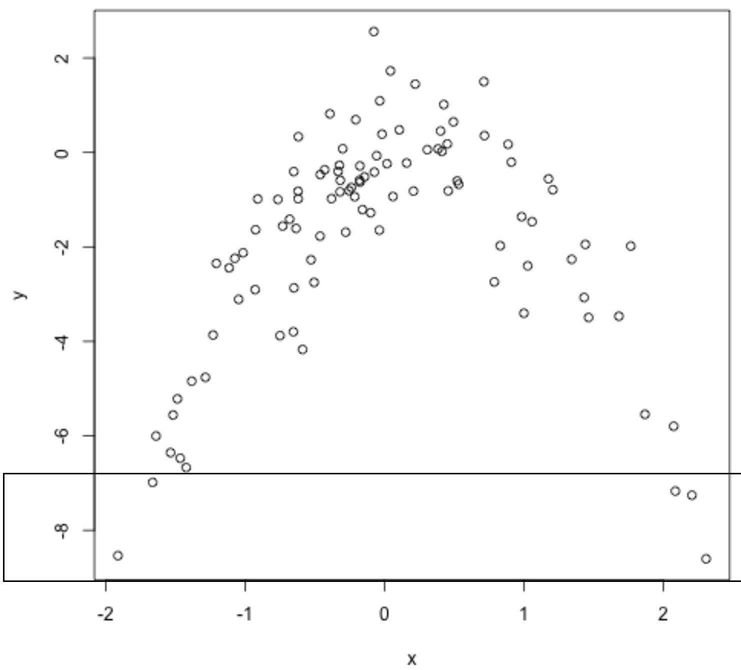
(a)

```
Set.seed(1)
y = rnorm(100)
x = rnorm(100)
y = x - 2 * x^2 + rnorm(100)
```

$$n = 100, p = 2\ and\ Y = X - 2X_2 + \epsilon$$

(b)

```
plot(x,y)
```



Quadratic plot. from about -2 to 2. from about -8 to 2.

(c)

```
library(boot)
Data = data.frame(x, y)
set.seed(1)
# i.
glm.fit = glm(y ~ x)
cv.glm(Data, glm.fit)$delta
```

```
## [1] 5.891 5.889
```

```
# ii.
glm.fit = glm(y ~ poly(x, 2))
cv.glm(Data, glm.fit)$delta
```

```
## [1] 1.087 1.086
```

```
# iii.
glm.fit = glm(y ~ poly(x, 3))
cv.glm(Data, glm.fit)$delta
```

```
## [1] 1.103 1.102
```

```
# iv.
glm.fit = glm(y ~ poly(x, 4))
cv.glm(Data, glm.fit)$delta
```

```
## [1] 1.115 1.114
```

(d)

```
set.seed(10)
# i.
glm.fit = glm(y ~ x)
cv.glm(Data, glm.fit)$delta
```

```
## [1] 5.891 5.889
```

```
# ii.
glm.fit = glm(y ~ poly(x, 2))
cv.glm(Data, glm.fit)$delta
```

```
## [1] 1.087 1.086
```

```
# iii.
glm.fit = glm(y ~ poly(x, 3))
cv.glm(Data, glm.fit)$delta
```

```
## [1] 1.103 1.102
```

```
# iv.
glm.fit = glm(y ~ poly(x, 4))
cv.glm(Data, glm.fit)$delta
```

```
## [1] 1.115 1.114
```

Exact same, because LOOCV will be the same since it evaluates n folds of a single observation.

(e) The quadratic polynomial had the lowest LOOCV test error rate. This was expected because it matches the true form of Y.

(f)

```
summary(glm.fit)
```

```
##
## Call:
```

```
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8913  -0.5244   0.0749   0.5932   2.7796
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.828      0.104  -17.55   <2e-16 ***
## poly(x, 4)1    2.316      1.041    2.22    0.029 *
## poly(x, 4)2  -21.059      1.041  -20.22   <2e-16 ***
## poly(x, 4)3   -0.305      1.041   -0.29    0.770
## poly(x, 4)4   -0.493.     1.041   -0.47    0.637
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
1
##
## (Dispersion parameter for gaussian family taken to be 1.085)
##
##     Null deviance: 552.21  on 99  degrees of freedom
## Residual deviance: 103.04  on 95  degrees of freedom
## AIC: 298.8
##
## Number of Fisher Scoring iterations: 2
```

p-values show statistical significance of linear and quadratic terms, which agrees with the CV results.