Chiayu Tu (Louis)

Stat 387 Assignment Three

Conceptual:

3. We now review k-fold cross-validation.

(a) Explain how k-fold cross-validation is implemented.

This method of evaluating a model involves dividing the set of observations into k groups or folds, where each fold is approximately the same size. The process then selects one of the k folds as a validation set, while the model is trained on the remaining k-1 folds. The resulting model is then used to predict the outcomes of the validation set, and the mean squared error, MSE1, is calculated.

This process is repeated k times, with each of the k folds being used exactly once as the validation set. In each repetition, the model is trained on the remaining k-1 folds, and the mean squared error is computed for the held-out fold. This results in k estimates of the test error.

To obtain a more accurate estimate of the model's performance, the k-fold CV estimate is calculated by taking the average of the k estimates of the test error.

(b) What are the advantages and disadvantages of k-fold cross-validation relative to:

i. The validation set approach?

Advantages of the validation set approach:

1. Simple and computationally efficient: The validation set approach involves training a single model and evaluating its performance on a separate validation set, which can be computationally efficient, especially when dealing with small datasets.
2. Allows for flexibility in model selection: The validation set approach allows for flexibility in selecting different models and hyperparameters to evaluate their performance on the validation set.

Disadvantages of the validation set approach:

1. Can provide a high variance estimate: Since the validation set approach involves training a model on a single random split of the data, the estimate of the model's performance can be sensitive to the specific split of the data into training and validation sets.
2. May not make optimal use of the available data: The validation set approach only uses a portion of the data for training and another portion for validation, which may not make optimal use of the available data, especially when the dataset is small.

ii. LOOCV?

Advantages of LOOCV:

1. Provides an unbiased estimate: LOOCV provides an unbiased estimate of the model's performance because it uses all the observations in the dataset for both training and validation.
2. Works well for small datasets: LOOCV can be a good choice for small datasets because it does not require dividing the dataset into subsets.

Disadvantages of LOOCV:

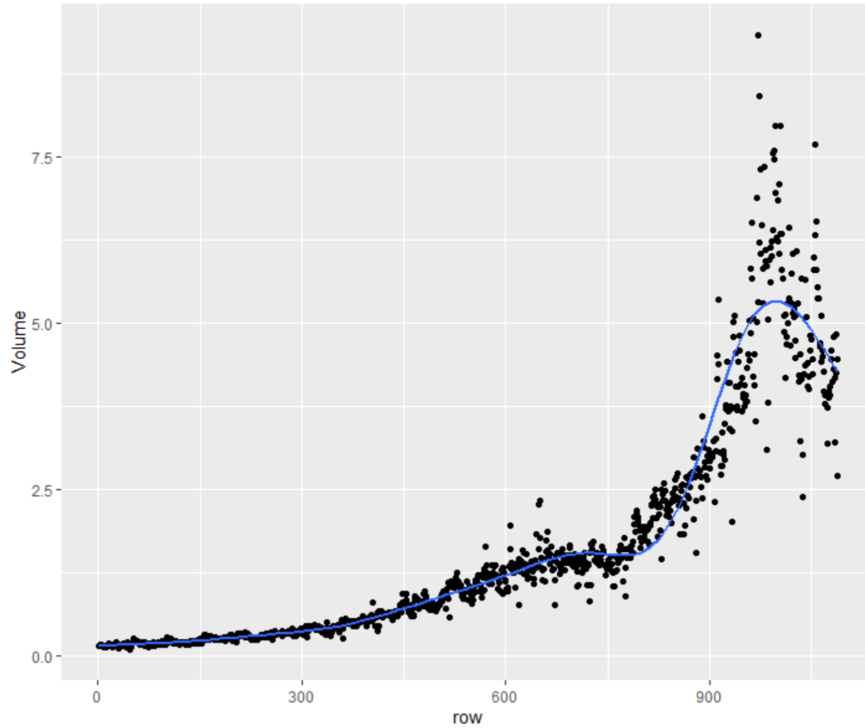
1. Computationally expensive: LOOCV requires training and evaluating the model k times, which can be computationally expensive, especially when the dataset is large.
2. More sensitive to outliers: LOOCV can be more sensitive to outliers than k-fold cross-validation because it uses each observation as a validation set, which can result in overfitting if there are outliers in the data.
3. Provides a high variance estimate: LOOCV provides a high variance estimate of the model's performance because it uses almost identical sets of observations for each model, which can result in a large variance in the estimates.

Applied:

13. This question should be answered using the Weekly data set, which is part of the ISLR2 package. This data is similar in nature to the Smarket data from this chapter's lab, except that it contains 1, 089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

(a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

```
      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) Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

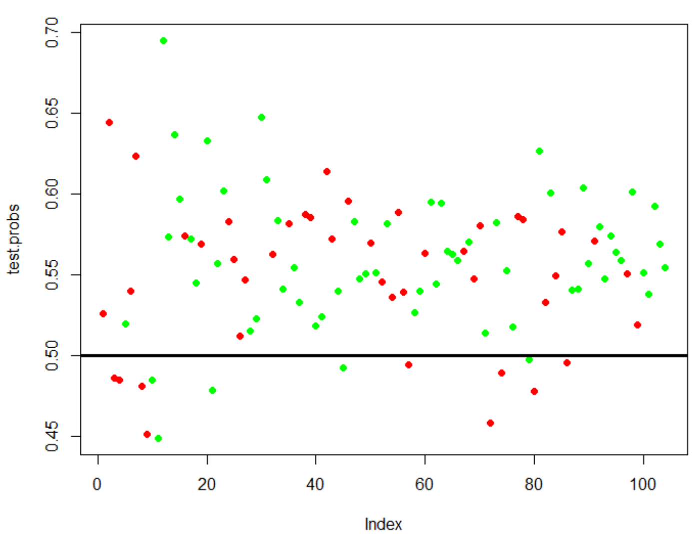
```
Call:
glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
    Volume, family = binomial, data = Weekly)

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) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

```
> probs = predict(logmod, type="response")
> preds = rep("Down", 1089)
> preds[probs > 0.5] = "Up"
> table(preds, Weekly$Direction)

preds  Down  Up
  Down   54  48
  Up    430 557
> mean(preds == Weekly$Direction)
[1] 0.5610652
```

(d) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).

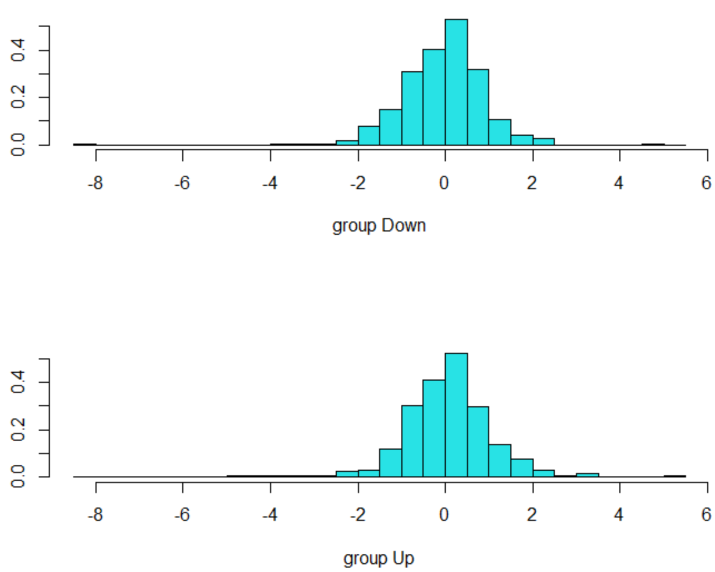
```
Call:
glm(formula = Direction ~ Lag2, family = binomial, data = train)

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
```



```
> test.pred = rep("Down", length(Weekly_test$Direction))
> test.pred[test.probs > 0.5] = "Up"
> table(test.pred, Weekly_test$Direction)

test.pred Down Up
     Down    9  5
     Up     34 56
> mean(test.pred == Weekly_test$Direction)
[1] 0.625
```

(e) Repeat (d) using LDA.

```
Call:
lda(Direction ~ Lag2, data = train.data)

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
> plot(lda.fit)
> lda.pred = predict(lda.fit, newdata=test.data, type="response")
> #lda.class = lda.pred$class
> table(lda.pred$class, test.data$Direction)

       Down Up
  Down    9  5
  Up     34 56
```



(f) Repeat (d) using QDA.

```
> # f
> qda.fit = qda(Direction~Lag2, data= train.data)
> qda.pred = predict(qda.fit, newdata=test.data, type="response")
> #qda.class = qda.pred$class
> table(qda.pred$class, test.data$Direction)

       Down Up
  Down    0  0
  Up     43 61
```

(g) Repeat (d) using KNN with K = 1.

```
> train.X = cbind(train.data$Lag2)
> test.X = cbind(test.data$Lag2)
> train.Y = cbind(train.data$Direction)
> knn.pred = knn(train.X, test.X, train.Y, k=1)
> table(knn.pred, test.data$Direction)

knn.pred Down Up
       1   21 30
       2   22 31
> knn3.pred = knn(train.X, test.X, train.Y, k=3)
> table(knn3.pred, test.data$Direction)

knn3.pred Down Up
        1   16 19
        2   27 42
```

(h) Repeat (d) using naive Bayes.

```
> # h
> library(e1071)
> nbayes.fit = naiveBayes(Direction~Lag2 ,data=Weekly)
> nbayes.pred = predict(nbayes.fit, test.data)
> table(nbayes.pred, test.data$Direction)

nbayes.pred Down Up
       Down    0  0
       Up     43 61
```

(i) Which of these methods appears to provide the best results on this data?

The regression model predicted the market correctly 62.5% of the time which is the highest of all the models, so we can know that the method appears to provide the best result.

8. We will now perform cross-validation on a simulated data set.

(a) Generate a simulated data set as follows:

```
> set.seed (1)
> x <- rnorm (100)
> y <- x - 2 * x^2 + rnorm (100)
```

In this data set, what is n and what is p? Write out the model used to generate the data in equation form.

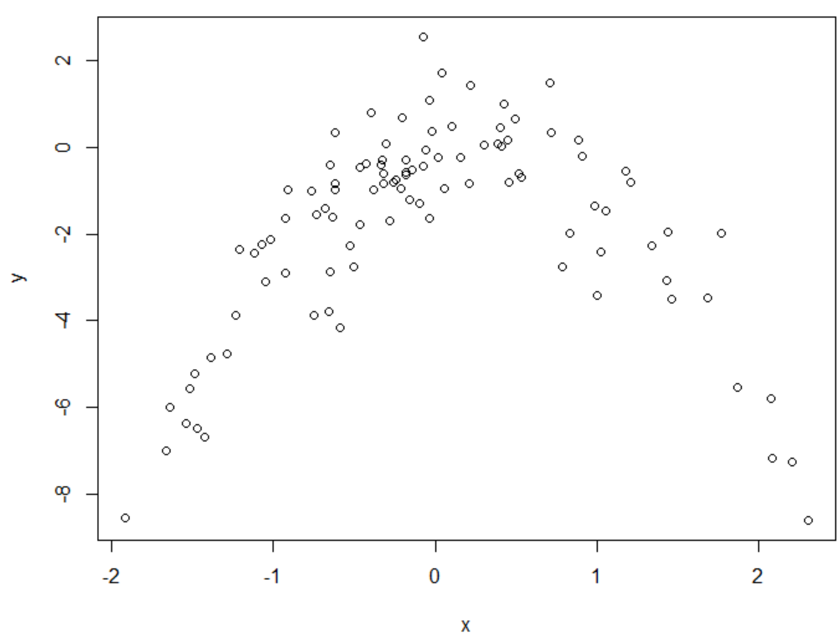
In this data set:

n = 100

p = 2

Y = X – 2*X^2 + ϵ, ϵ~N (0, 1)

(b) Create a scatterplot of X against Y . Comment on what you find.



(c) Set a random seed, and then compute the LOOCV errors that result from fitting the following four models using least squares:

i. $Y = \beta_0 + \beta_1 X + \epsilon$

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

ii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \epsilon$

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

iii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$

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

iv. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \epsilon$.

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

Note you may find it helpful to use the data.frame() function to create a single data set containing both X and Y .

R Code:

```
#a
library(ISLR)
library(tidyverse)
summary(Weekly)

Weekly %>% mutate(row = row_number()) %>%
  ggplot(aes(x = row, y = Volume)) +
  geom_point() +
  geom_smooth(se = FALSE)

# b
logmod <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
              data = Weekly,
              family = binomial)
summary(logmod)

# c
probs = predict(logmod, type="response")
preds = rep("Down", length(probs))
preds[probs > 0.5] = "Up"
table(preds, Weekly$Direction)
mean(preds == Weekly$Direction)

# d
train <- (Weekly$Year < 2009)
#Weekly_train <- Weekly[train,]
train.data = Weekly[train, ]
#Weekly_test <- Weekly[!train,]
```

```
test.data = Weekly[!train, ]

train.data

test.data

Direction_train <- Weekly_train$Direction

Direction_test <- Weekly_test$Direction


logistic.data <- glm(Direction ~ Lag2,

                data = train,

                family = binomial)


summary(logistic.data)


test.probs <- predict(logistic.data, test.data, type = "response")

testdirs = Weekly$Direction[Weekly$Year > 2008]

plot(test.probs,

   col = ifelse(testdirs == "Down", "red", "green"), pch = 16)

abline(h = 0.5, lwd = 3)


test.pred = rep("Down", length(Weekly_test$Direction))

test.pred[test.probs > 0.5] = "Up"

table(test.pred, Weekly_test$Direction)

mean(test.pred == Weekly_test$Direction)


# e

library(MASS)    # for LDA

lda.fit = lda(Direction~Lag2, data= train.data)

lda.fit

plot(lda.fit)

lda.pred = predict(lda.fit, test.data, type="response")
```

```
#lda.class = lda.pred$class
table(lda.pred$class, test.data$Direction)

# f
qda.fit = qda(Direction~Lag2, data= train.data)
qda.pred = predict(qda.fit, test.data, type="response")
#qda.class = qda.pred$class
table(qda.pred$class, test.data$Direction)

# g
library(class)    # for KNN
set.seed(1)
train.X = cbind(train.data$Lag2)
test.X = cbind(test.data$Lag2)
train.Y = cbind(train.data$Direction)
knn.pred = knn(train.X, test.X, train.Y, k=1)
table(knn.pred, test.data$Direction)

knn3.pred = knn(train.X, test.X, train.Y, k=3)
table(knn3.pred, test.data$Direction)

# h
library(e1071)
nbayes.fit = naiveBayes(Direction~Lag2 ,data=Weekly)
nbayes.pred = predict(nbayes.fit, test.data)
table(nbayes.pred, test.data$Direction)

# 2
set.seed(1)
```

```
y <- rnorm(100)

x <- rnorm(100)

y <- x - 2 * x^2 + rnorm(100)

plot(x, y)


# 3.1

library(boot)

set.seed(1)

Data <- data.frame(x, y)

fit.glm.1 <- glm(y ~ x)

cv.glm(Data, fit.glm.1)$delta[1]

# 3.2

fit.glm.2 <- glm(y ~ poly(x, 2))

cv.glm(Data, fit.glm.2)$delta[1]

# 3.3

fit.glm.3 <- glm(y ~ poly(x, 3))

cv.glm(Data, fit.glm.3)$delta[1]

# 3.4

fit.glm.4 <- glm(y ~ poly(x, 4))

cv.glm(Data, fit.glm.4)$delta[1]
```