STAT 4510/7510 - HW5

Yang Anton #14405729

Instructions: Please list your name and student number clearly. In order to receive credit for a problem, your solution must show sufficient detail so that the grader can determine how you obtained your answer.

Generate a WORD document using R markdown and submit a PDF generated from the WORD document. All R code should be included, as well as all output produced. Upload your work to the Canvas course site.

Problem 1

In this problem, we will once again consider the data found in `tumor.csv` from Homework 4.

(a) Filter the dataset to contain only the columns `Diagnosis`, `Radius`, `Texture`, `Smoothness`, `Concavity`, and `Fractal.Dimension`. Remember to change `Diagnosis` to a factor variable.

```
library(ISLR)
library(boot)
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

##
## Attaching package: 'lattice'

## The following object is masked from 'package:boot':
##
##     melanoma

set.seed(1)
tumor<-read.csv("tumor.csv")
data<-
tumor[,c("Diagnosis","Radius","Texture","Smoothness","Concavity","Fractal.Dim
ension")]
data$Diagnosis<-as.factor(data$Diagnosis)
```

(b) Conduct 5-fold cross-validation to select the best method between logistic regression (with a probability cutoff of 0.5), LDA, and QDA for classifying `Diagnosis` using all other predictors from (a). Which method is best?

```
set.seed(1)
train<-trainControl(method="cv",
                    number = 5)
```

```r
model1 <- train(Diagnosis ~ ., data = data, method = "glm", trControl =
train,family="binomial")

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

logistic_prediction<-predict(model1, data, type="prob")
cutoff<-0.5
predicted_classes<-ifelse(logistic_prediction[, "Malignant"]>cutoff,
"Malignant","Benign")
accuracy<-mean(predicted_classes == data$Diagnosis)
cv_error1<-1-accuracy
print(cv_error1)

## [1] 0.05272408

model2<-train(Diagnosis~., data=data, method = "lda", trControl=train)
cv_error2<-1-model2$results$Accuracy
print(cv_error2)

## [1] 0.07731989

model3<-train(Diagnosis~.,data=data, method ="qda", trControl=train)
cv_error3<-1-model3$results$Accuracy
print(cv_error3)

## [1] 0.05451179
```

According to the result produced, the best method is Logistic Regression for 5-fold cross-validation.

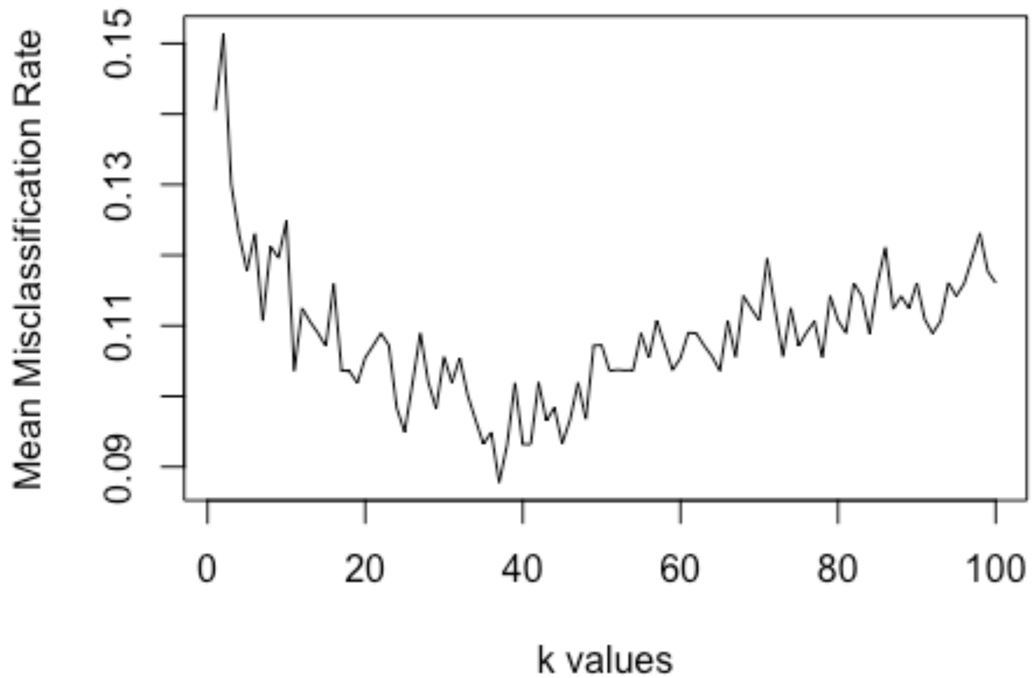
(c) Now, conduct 5-fold cross-validation for KNN using $k = 1$ to $k = 100$ nearest neighbors. Plot the mean misclassification rate over the five folds for each value of k. What is the best value of k to use?

```r
set.seed(1)
k<-1:100
mean_error<-numeric(length(k))

for (i in seq_along(k)){
  train_control<-trainControl(method="cv", number = 5)
  model4<-train(Diagnosis ~., data=data,
method="knn",trControl=train,tuneGrid = data.frame(k=k[i]))

  mean_error[i]<-1-model4$results$Accuracy
}

plot(k, mean_error, type="l", xlab="k values", ylab="Mean Misclassification
Rate")
```




```
best_k<-k[which.min(mean_error)]
best_k_rate<-mean_error[best_k]
print(best_k)

## [1] 37

print(best_k_rate)

## [1] 0.08778572
```

According to the 5-fold cross-validation for KNN, k value of 37 is the best a misclassification rate of 0.08778572.

(d) Comparing the results from parts (b) and (c), which method provides the best results for classification?

```
Error_Table<-data.frame("Logistic"=cv_error1,
                        "LDA" = cv_error2,
                        "QDA" = cv_error3,
                        "KNN" = best_k_rate)
best_method<-which.min(Error_Table)
print(best_method)

## Logistic
##        1
```

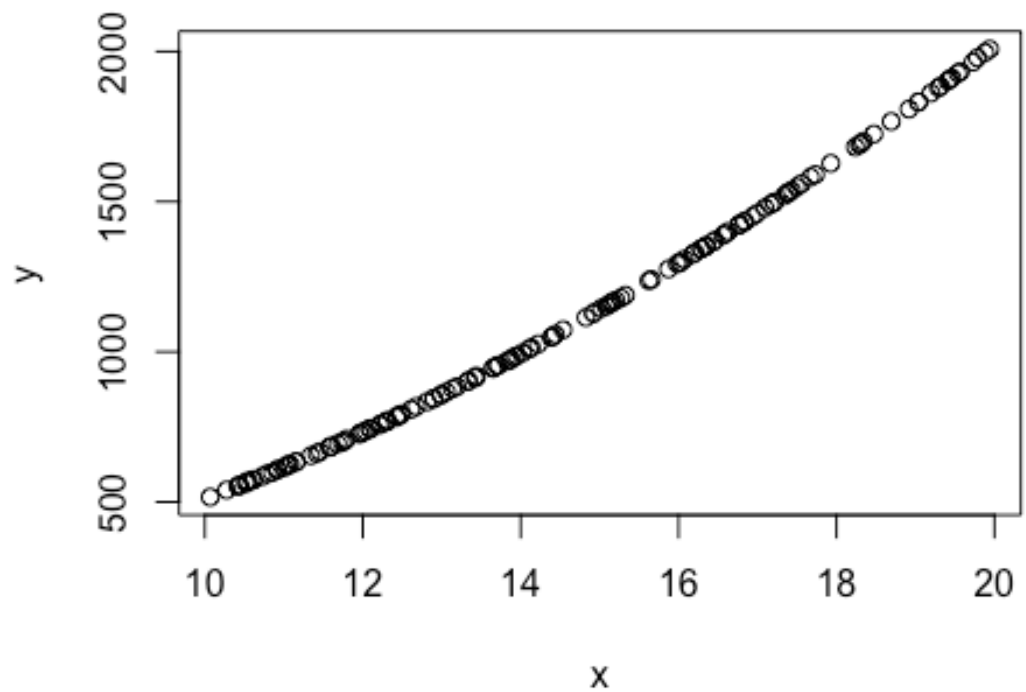
This shows that the Logistic Regression produces the best result with the lowest misclassification rate.

Problem 2

The file `HW5data.csv` contains a dataframe of (x, y) data.

(a) Make a plot of x vs. y and comment on the type of relationship you observe. Does it appear to be linear? Quadratic? Something else?

```
data2<-read.csv("HW5data.csv")
plot(data2$x,data2$y,xlab="x",ylab="y")
```



According to the plot, it appears that the relationship between x and y are strongly linear.

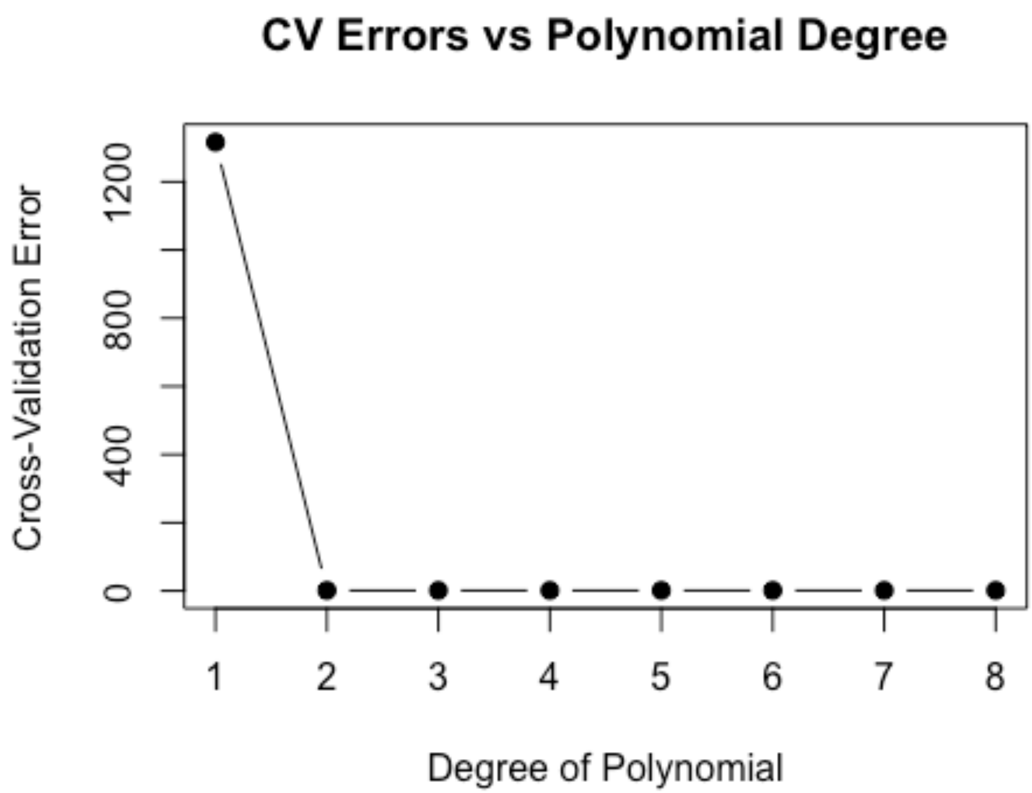
(b) Use LOOCV to fit a regression model with polynomial degrees ranging from degree 1 to degree 8. *(If you use the* `cv.glm()` *function, be sure to load the* `boot` *library.)* Plot the CV errors vs the degree of the polynomial. Which degree polynomial do you think fits best?

```
set.seed(1)
library(boot)

mse_5<-numeric(8)
```

```
for (i in 1:8){
  glm_model<-glm(y~poly(x,i), data=data2)
  loocv<-cv.glm(data=data.frame(x=data2$x,y=data2$y),glmfit = glm_model)
  mse_5[i] <- loocv$delta[1]
}

plot(1:8, mse_5, type = "b", pch = 19, xlab = "Degree of Polynomial", ylab =
"Cross-Validation Error", main = "CV Errors vs Polynomial Degree")
```




```
print(mse_5)

## [1] 1316.108861    1.242827    1.246210    1.264479    1.282904
1.305092
## [7]    1.260793    1.298818
```

According to the result, the polynomial degree 2 fits best with lowest MSE of 1.242827.

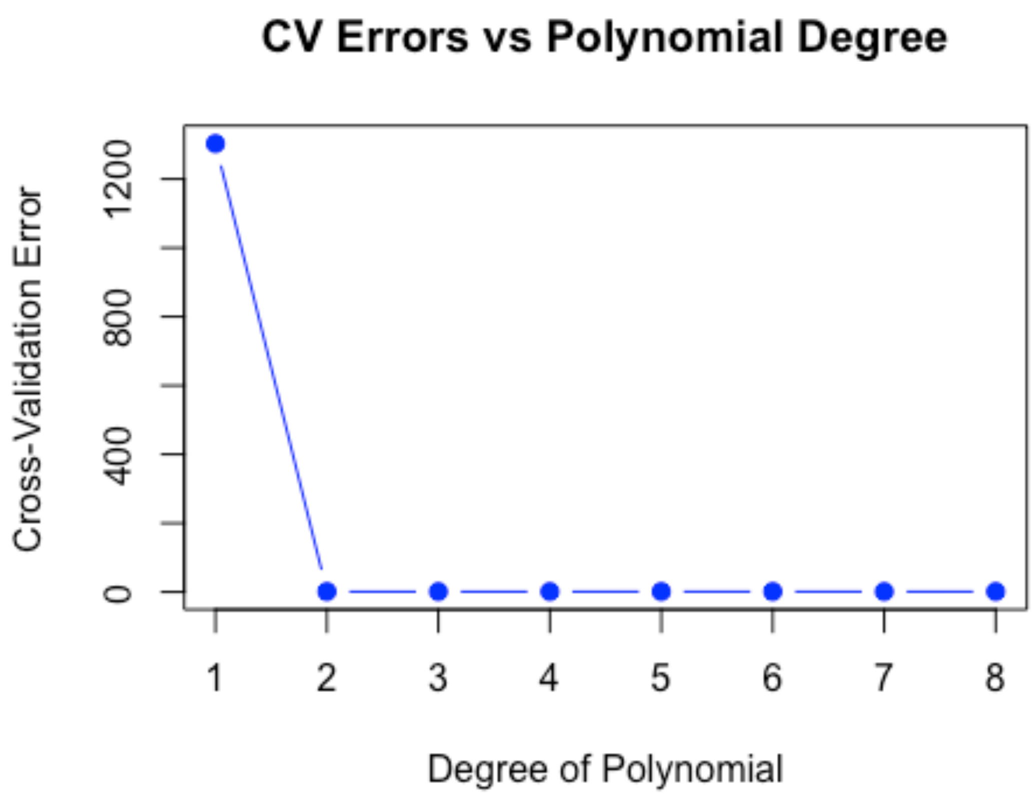
(c) Repeat part (b), using 10-fold cross-validation. Add the plot of the CV errors to the plot you made in part (b), using a different color. Which degree polynomial do you think fits best? Is it different from the result in part (b)? Which approach (LOOCV vs 10-fold CV) took longer to execute in R?

```
set.seed(1)
```

```
mse_10<-numeric(8)

for (i in 1:8) {
  glm_model <- glm(y ~ poly(x, i), data = data2)
  cv <- cv.glm(data = data.frame(x = data2$x, y = data2$y), glmfit =
glm_model, K = 10)
  mse_10[i] <- cv$delta[1]
}

plot(1:8, mse_10, type = "b", pch = 19, col = "blue", xlab = "Degree of
Polynomial", ylab = "Cross-Validation Error", main = "CV Errors vs Polynomial
Degree")
```




```
print(mse_10)

## [1] 1303.010927    1.245186    1.250842    1.263882    1.311583
1.328228
## [7]    1.271787    1.333404
```

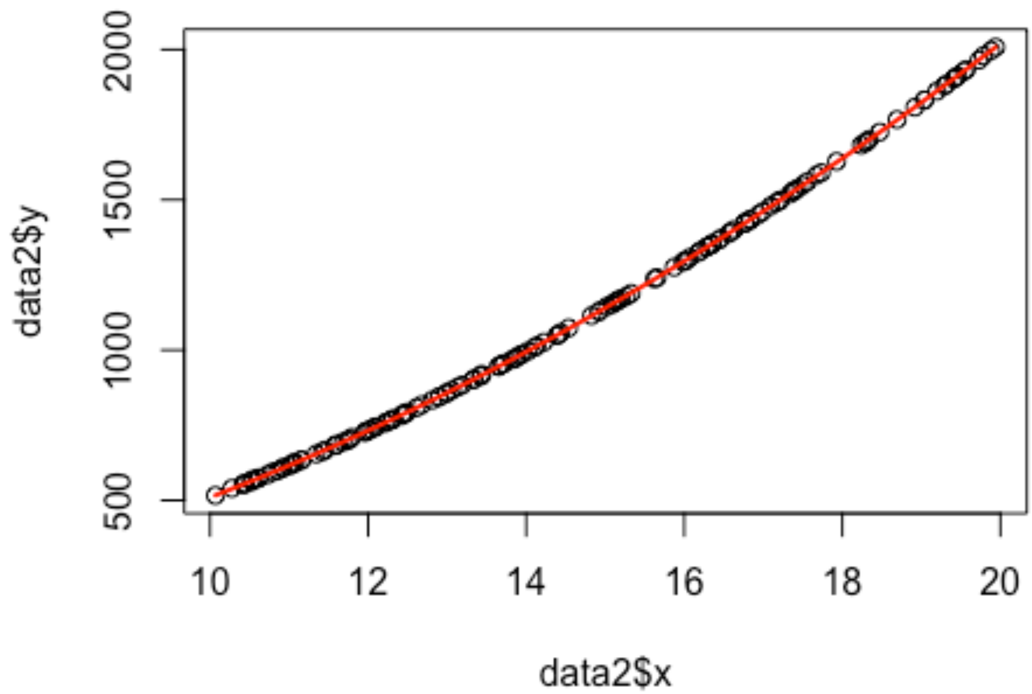
According to the result, the polynomial degree 2 fits best with lowest MSE of 1.245186 and the result is not so much different than part b. LOOCV took longer to execute in R. 10-fold CV is like immediate.

(d) Produce the scatterplot from part (a) again, and add to it the regression polynomial you selected from part (c). Does it appear to be a good fit?

```
plot(data2$x,data2$y)

glm_model<-glm(y~poly(x,2),data=data2)
x_seq<-seq(min(data2$x), max(data2$x), length.out=100)
y_pred<-predict(glm_model, newdata = data.frame(x=x_seq))

lines(x_seq, y_pred, col="red", lwd=2)
```



It appears to be a really good fit.

(e) Assume that the degree of the model you chose in part (c) is the actual degree of the polynomial that describes the data. Conduct a bootstrap analysis to estimate the coefficient parameters and their standard errors for this model. Based on these values, give a 95% confidence interval for β_1, the coefficient of x in the model. *(Note: You can approximate a 95% confidence interval using the formula $\hat{\beta}_1 \pm 2SE(\hat{\beta}_1)$.)*

```
coef_analysis<-function(data, i){
  d<-data[i,]
  fit<-glm(y~poly(x,2),data=d)
  coef(fit)
}
```

```r
boot_result<-boot(data=data2, statistic = coef_analysis, R=1000)
print(boot_result)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data2, statistic = coef_analysis, R = 1000)
##
##
## Bootstrap Statistics :
##      original      bias    std. error
## t1* 1126.3811   0.5976517    35.21219
## t2* 5141.4176 -33.5673646   215.42946
## t3*  435.9696  -5.6846605    20.69634
```

```r
beta1<-boot_result$t[, 2]
se_beta1<-sd(beta1)
beta1_original<-coef(glm_model)[2]

conf_interval<-c(beta1_original - 2*se_beta1, beta1_original + 2*se_beta1)
print(conf_interval)
```

```
## poly(x, 2)1 poly(x, 2)1
##    4710.559    5572.277
```

The 95% confidence interval is for x^2 estimate is (4710.559, 5567.604).