Biostats 546 HW 3

Ronald Buie

February 10, 2019

Due Via Online Submission to Canvas: Sunday, February 24 at 12 PM (Noon)

1. In this problem, you will make use of a (real) dataset of your choice in order to predict a binary response Y using predictors X1; : : : ;Xp. You should have p 5.

```
library(ISLR)

AutoData <- ISLR::Auto

AutoData$yearBinary <- "older"
AutoData[AutoData$year > 76,]$yearBinary <- "newer"
AutoData$yearBinary <- as.factor(AutoData$yearBinary)
AutoData$origin <- as.factor(AutoData$origin)
AutoData$cylinders <- as.factor(AutoData$cylinders)
AutoData$year <- NULL
AutoData$name <- NULL
```

(a) Describe the dataset. What is the response, and what are the predictors?

```
head(AutoData)
```

```
##   mpg cylinders displacement horsepower weight acceleration origin
## 1  18         8          307        130   3504         12.0      1
## 2  15         8          350        165   3693         11.5      1
## 3  18         8          318        150   3436         11.0      1
## 4  16         8          304        150   3433         12.0      1
## 5  17         8          302        140   3449         10.5      1
## 6  15         8          429        198   4341         10.0      1
##   yearBinary
## 1      older
## 2      older
## 3      older
## 4      older
## 5      older
## 6      older
```

```
summary(AutoData)
```

```
##       mpg        cylinders  displacement     horsepower        weight    
##  Min.   : 9.00   3:  4     Min.   : 68.0   Min.   : 46.0   Min.   :1613  
##  1st Qu.:17.00   4:199     1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225  
##  Median :22.75   5:  3     Median :151.0   Median : 93.5   Median :2804  
##  Mean   :23.45   6: 83     Mean   :194.4   Mean   :104.5   Mean   :2978  
##  3rd Qu.:29.00   8:103     3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615  
##  Max.   :46.60             Max.   :455.0   Max.   :230.0   Max.   :5140
##   acceleration   origin  yearBinary
##  Min.   : 8.00   1:245   newer:178
##  1st Qu.:13.78   2: 68   older:214
##  Median :15.50   3: 79
##  Mean   :15.54
##  3rd Qu.:17.02
##  Max.   :24.80
```

The Auto data set contains information about the performance, origin and model of 392 vehicles. We have created a binary indicator variable for the year where "older" is 70 to 76, and "newer" is > 76 and will use this as our outcome. our predictors are mpg, cylinders, displacement, horsepower, weight, acceleration, and origin.

(b) Fit a logistic regression model to predict Y using X1; : : : ;Xp. What is the classication error (i.e. the fraction of incorrectly classied observations) on the training set?

```
LogFitAutoData <- glm(yearBinary ~ ., data = AutoData, family="binomial")

LogFitAutoDataPredictions <- predict(LogFitAutoData, type="response")

LogPredictions <- rep("newer",392)
LogPredictions[LogFitAutoDataPredictions >.5]<- "older"

LogPredictionsErrorRate <- 1-mean(LogPredictions==AutoData$yearBinary)
```

The training error rate is 21%

(c) Use the validation set approach in order to estimate the test classication error. Report the error you obtain.

```
set.seed(1000)

trainSample <- sample(392, 196)

LogFitAutoDataVS <- glm(yearBinary ~ ., data = AutoData, family="binomial", subset = trainSample)

LogFitAutoDataVSPredictions <- predict(LogFitAutoDataVS, newdata = AutoData, type="response")

LogPredictionsVS <- rep("newer",392)
LogPredictionsVS[LogFitAutoDataVSPredictions >.5]<- "older"

LogPredictionsVSErrorRate <- 1-mean(LogPredictionsVS==AutoData$yearBinary)[-trainSample]
```

The training error rate for the Validation set approach is 21%

(d) Use the leave-one-out cross-validation approach in order to estimate the test classication error. Report the error you obtain.

```
library(boot)

LogFitAutoDataCVError <- cv.glm(AutoData, LogFitAutoData)
```

```
LogFitAutoDataCVError$delta
```

```
## [1] 0.1382103 0.1381981
```

The cross validation approach yiels a MSE of 0.1382103 and bias corrected MSE of 0.1381981

(e) Use the 5-fold cross-validation approach in order to estimate the test classi-cation error. Report the error you obtain.

```
library(boot)
set.seed(1000)
attach(AutoData)


LogFitAutoDataCVK5Error <- cv.glm(AutoData, LogFitAutoData, K=5)

LogFitAutoDataCVK5Error$delta[1]
```

```
## [1] 0.1393391
```

The cross validation approach yiels an MSE of 0.1382103 and bias corrected MSE of 0.1381981

(f) Comment on your ndings in (b)-(e).

The error rates of our logistic regression and validation set approaches were similar, 21%. Also similar, at 14% were the error rates of ourcross validation and 5-fold cross-validation approaches. The cross validation approaches identified models with superior fit than our

2. In this problem, you will make use of a (real) dataset of your choice in order to predict a continuous response Y using predictors X1; : : : ;X6. You need to choose a dataset with at least 6 predictors. If your dataset has more than 6 predictors, then please choose 6 of them now. In other words, you should have p = 6.

```
AutoData6 <- AutoData[,!(names(AutoData) %in% "origin")]
AutoData6$cylinders <- as.numeric(as.character(AutoData6$cylinders))
```

(a) Describe the dataset. What is the response, and what are the predictors? 1

```
head(AutoData6)
```

```
##   mpg cylinders displacement horsepower weight acceleration yearBinary
## 1  18         8          307        130   3504         12.0      older
## 2  15         8          350        165   3693         11.5      older
## 3  18         8          318        150   3436         11.0      older
## 4  16         8          304        150   3433         12.0      older
## 5  17         8          302        140   3449         10.5      older
## 6  15         8          429        198   4341         10.0      older
```

```
summary(AutoData6)
```

```
##       mpg          cylinders      displacement     horsepower
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0
##      weight      acceleration   yearBinary
##  Min.   :1613   Min.   : 8.00   newer:178
##  1st Qu.:2225   1st Qu.:13.78   older:214
##  Median :2804   Median :15.50
##  Mean   :2978   Mean   :15.54
##  3rd Qu.:3615   3rd Qu.:17.02
##  Max.   :5140   Max.   :24.80
```

For this question we chose the same data set as above, but only included mpg, cylinders, displacement, horsepower, weight, and acceleration as predictors, and our binary year as our outcome. Cylindars, previously a dummy variable, has been converted to a continuous integer to meet the p=6 requirememt

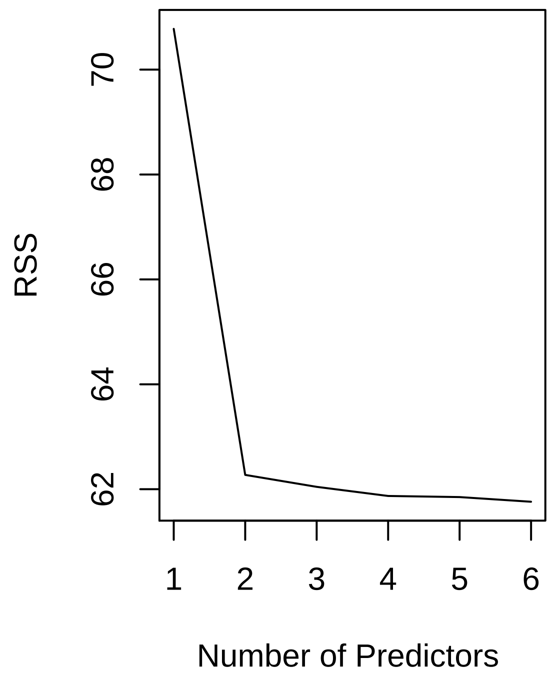
(b) Fit a least squares linear model using every possible subset of the features. How many models did you t?

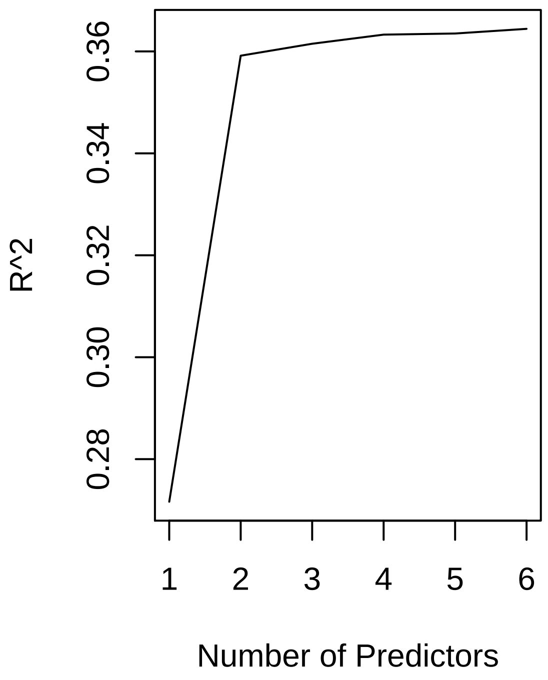
```r
library(leaps)
set.seed(1000)
a6Subsets <- regsubsets(yearBinary~., AutoData6)
a6SubsetsSummary <- summary(a6Subsets)
```

63 models are possible.

(c) Re-create Figure 6.1 in the textbook using your data. The left-hand panel should display (training set) RSS on the y-axis, and the right-hand panel should display the R2 on the y-axis. Both panels should display the number of predictors on the x-axis.

```r
par(mfrow=c(1,2))
plot(a6SubsetsSummary$rss ,xlab="Number of Predictors ",ylab="RSS",type="l")
plot(a6SubsetsSummary$rsq ,xlab="Number of Predictors", ylab="R^2",type="l")
```





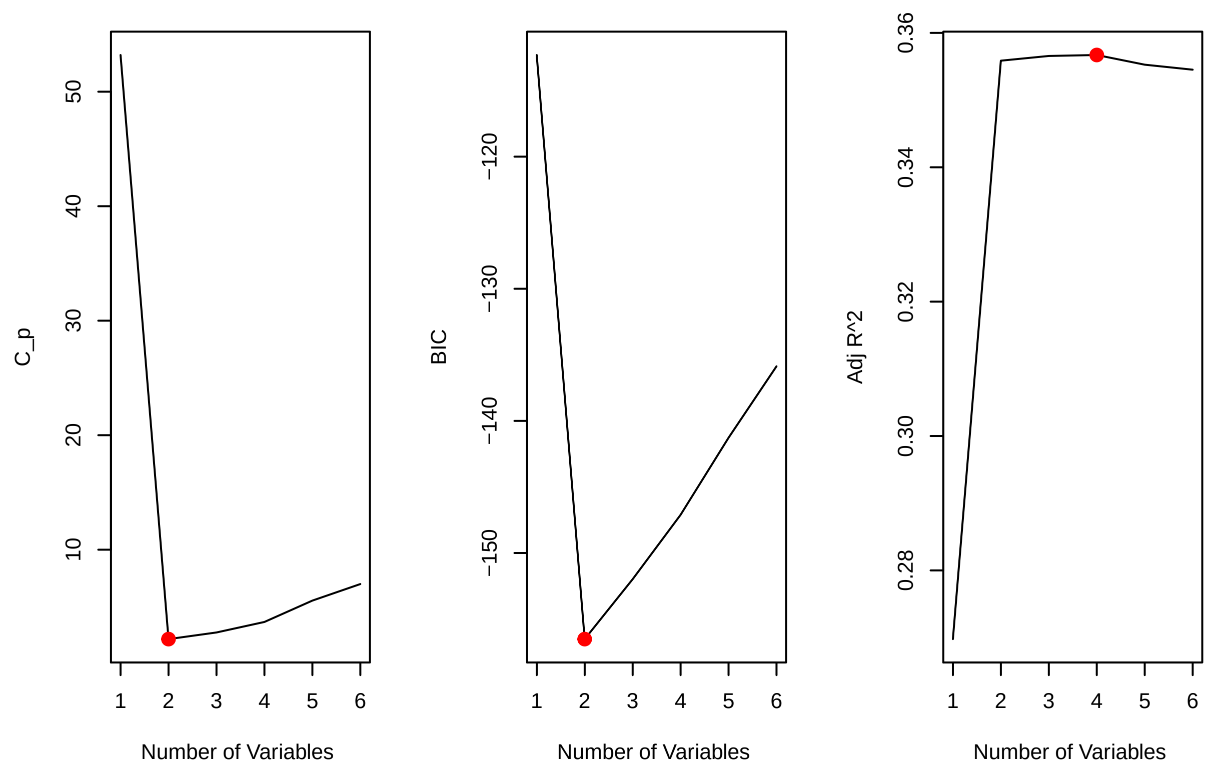
(d) Report the predictors in each of the models M0;M1; : : : ;Mp.

```
a6SubsetsSummary
```

```
## Subset selection object
## Call: regsubsets.formula(yearBinary ~ ., AutoData6)
## 6 Variables  (and intercept)
##              Forced in Forced out
## mpg              FALSE      FALSE
## cylinders        FALSE      FALSE
## displacement     FALSE      FALSE
## horsepower       FALSE      FALSE
## weight           FALSE      FALSE
## acceleration     FALSE      FALSE
## 1 subsets of each size up to 6
## Selection Algorithm: exhaustive
##          mpg cylinders displacement horsepower weight acceleration
## 1  ( 1 ) "*" " "       " "          " "        " "    " "
## 2  ( 1 ) "*" " "       " "          " "        "*"    " "
## 3  ( 1 ) "*" " "       "*"          " "        "*"    " "
## 4  ( 1 ) "*" "*"       "*"          " "        "*"    " "
## 5  ( 1 ) "*" "*"       "*"          "*"        "*"    " "
## 6  ( 1 ) "*" "*"       "*"          "*"        "*"    "*"
```

(e) Re-create Figure 6.2 in the textbook using your data. The y-axes for the three panels should be Cp, BIC, and adjusted R2; all x-axes should display the number of predictors.

```r
par(mfrow=c(1,3))
plot(a6SubsetsSummary$cp ,xlab="Number of Variables ",ylab="C_p", type='l')
points(which.min(a6SubsetsSummary$cp),min(a6SubsetsSummary$cp),col="red",cex=2,pch=20)
plot(a6SubsetsSummary$bic ,xlab="Number of Variables ",ylab="BIC", type='l')
points(which.min(a6SubsetsSummary$bic),min(a6SubsetsSummary$bic),col="red",cex=2,pch=20)
plot(a6SubsetsSummary$adjr2 ,xlab="Number of Variables ",ylab="Adj R^2", type='l')
points(which.max(a6SubsetsSummary$adjr2),max(a6SubsetsSummary$adjr2),col="red",cex=2,pch=20)
```



(f) Based on your results, what is the best" least squares linear model on this dataset? (Your answer should include not only the predictors, but also the coecient estimates.) Explain your answer.

Our best model based on two of our measures, C_p and BIC, contains 2 variables, mpg and weight.

```r
library(stargazer)
```

```
##
## Please cite as:
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
stargazer(round(coef(a6Subsets, 2), 6), no.space=TRUE, header=FALSE, float= FALSE, ci = TRUE, table.pla
```

(Intercept)	mpg	weight
3.924	-0.062	-0.0003

However, R^2 suggests the 4 variable model using mpg, cylinders, displacement, and weight.

```
library(stargazer)
stargazer(round(coef(a6Subsets, 4), 6), no.space=TRUE, header=FALSE, float= FALSE, ci = TRUE, table.pla
```

(Intercept)	mpg	cylinders	displacement	weight
4.078	-0.061	-0.040	0.001	-0.0004

3. Consider using the Auto data set to predict mpg using polynomial functions of horsepower in a least squares linear regression.

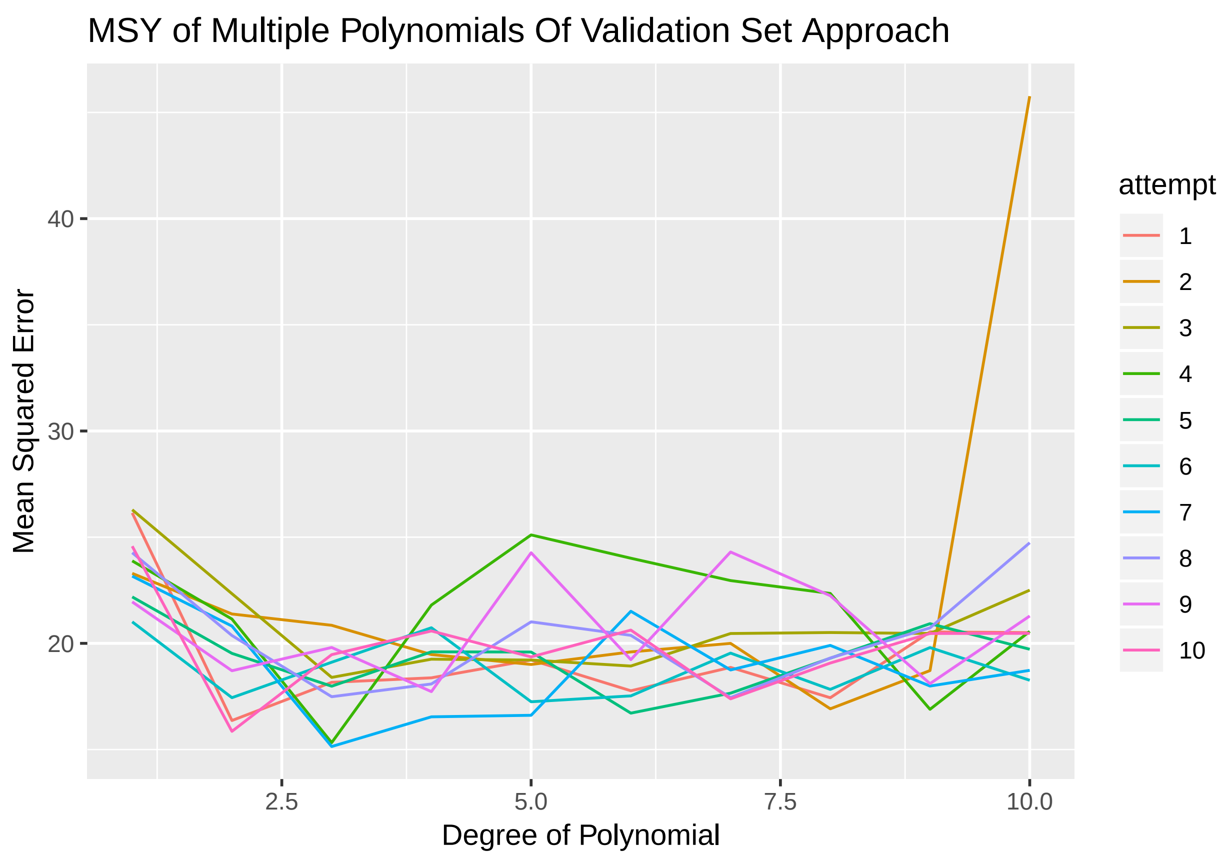
```
AutoDataPoly <- ISLR::Auto
AutoDataPoly <- AutoDataPoly[-9]
```

(a) Perform the validation set approach, and produce a plot like the one in the right-hand panel of Figure 5.2 of the textbook. Your answer won't look exactly the same as the results in Figure 5.2, since you'll be starting with a dierent random seed. Discuss your ndings. What degree polynomial is best, and why?

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

## The following object is masked from 'AutoData':
##
##     mpg
```

```
MSY <- data.frame(msy = numeric(), poly = numeric(), attempt = numeric())
counter <- 0
for(i in 1:10){
  set.seed(i)
  for(j in 1:10){
    counter <- counter+1
  train <- sample(nrow(AutoDataPoly), nrow(AutoDataPoly)/2)
  AutoDataPolyLM <- lm(mpg~poly(horsepower,j), data = AutoDataPoly, subset = train)
  MSY[counter,] <- c(mean((AutoDataPoly$mpg -predict(AutoDataPolyLM,AutoDataPoly))[-train]^2),j,i)
  }
}
MSY$attempt <- as.factor(MSY$attempt)
ggplot(MSY, aes(y=msy, x = poly, color = attempt)) +
  geom_line() +
  ggtitle("MSY of Multiple Polynomials Of Validation Set Approach") +
  xlab("Degree of Polynomial")+
  ylab("Mean Squared Error")
```



```
MeanMSY <- aggregate(msy ~ poly, MSY, mean)[2]
which.min(MeanMSY$msy)
```

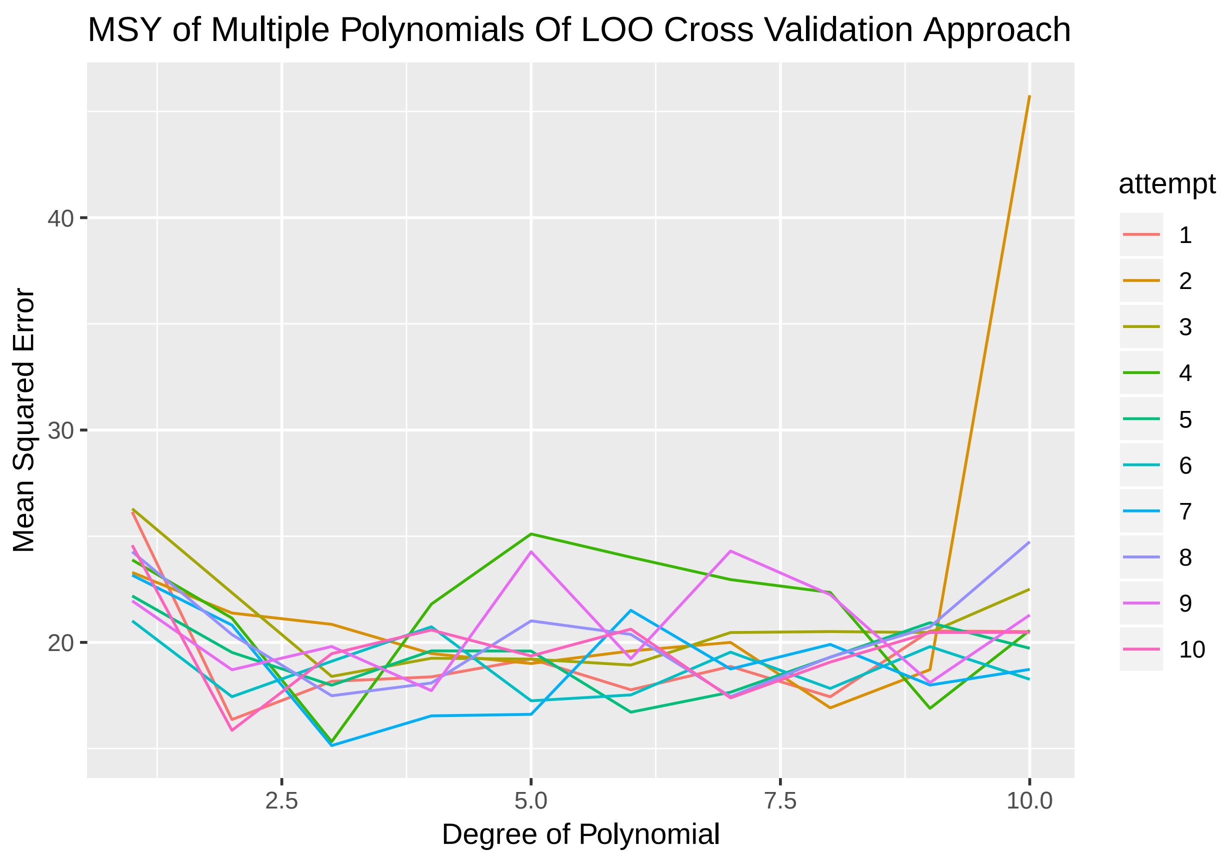
```
## [1] 3
```

The resulting chart shows the performance of increasing polynomials for different samples of our data. This provides a range of guesses for possible results when sampling from a larger data set. Polynomial 3 has the lowest mean MSY across all samples, and so I'd take that to be the best polynomial to use.

(b) Perform leave-one-out cross-validation, and produce a plot like the one in the left-hand panel of Figure 5.4 of the textbook. Discuss your ndings. What degree polynomial is best, and why?

```
library(ggplot2)
MSY <- data.frame(msy = numeric(), poly = numeric(), attempt = numeric())
counter <- 0
for(i in 1:10){
  set.seed(i)
  for(j in 1:10){
    counter <- counter+1
  train <- sample(nrow(AutoDataPoly), nrow(AutoDataPoly)/2)
  AutoDataPolyLM <- lm(mpg~poly(horsepower,j), data = AutoDataPoly, subset = train)
  MSY[counter,] <- c(mean((AutoDataPoly$mpg -predict(AutoDataPolyLM,AutoDataPoly))[-train]^2),j,i)
  }
}
MSY$attempt <- as.factor(MSY$attempt)
```

```r
ggplot(MSY, aes(y=msy, x = poly, color = attempt)) +
  geom_line() +
  ggtitle("MSY of Multiple Polynomials Of LOO Cross Validation Approach") +
  xlab("Degree of Polynomial")+
  ylab("Mean Squared Error")
```



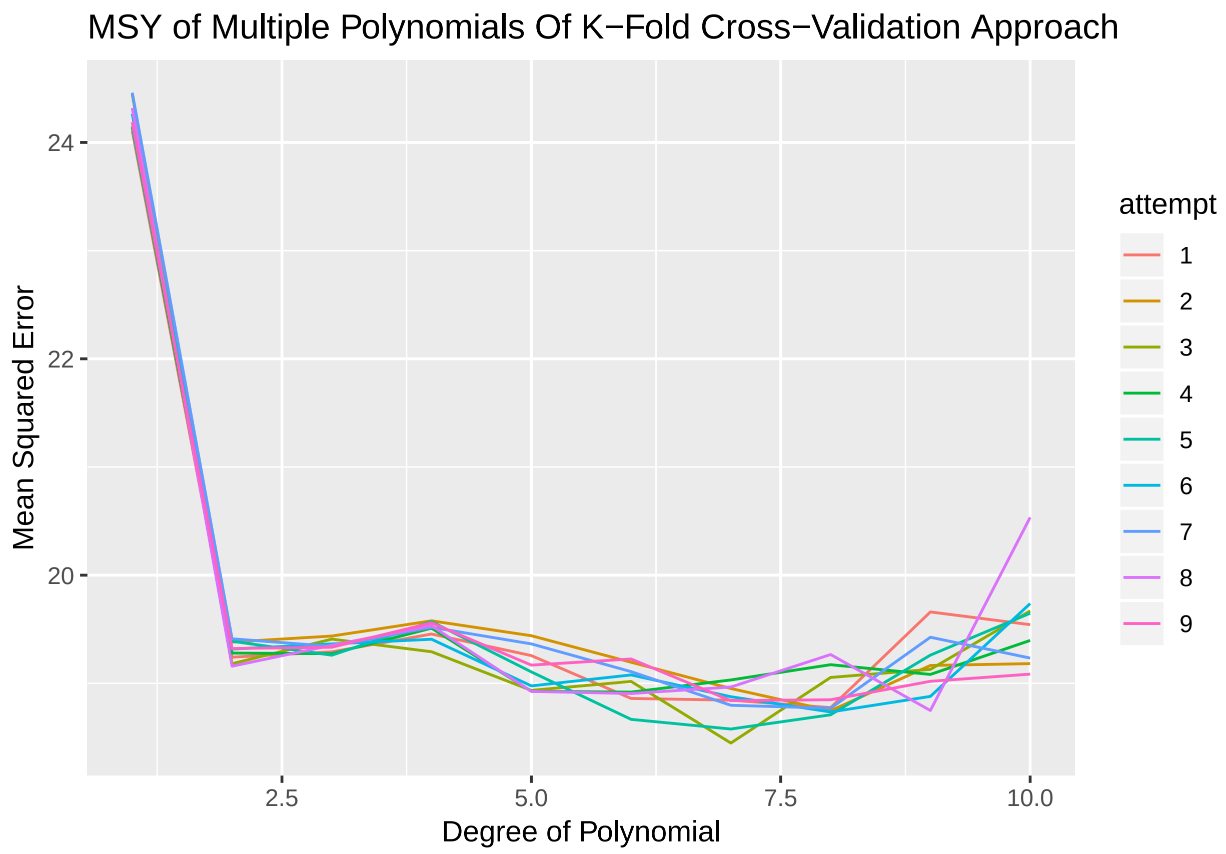
```r
MeanMSY <- aggregate(msy ~ poly, MSY, mean)[2]
which.min(MeanMSY$msy)
```

```
## [1] 3
```

The mean MSY of our estimates using the third polynomial is lower than the others and so considered our best performing.

(c) Perform 10-fold cross-validation, and produce a plot like the one in the right-hand panel of Figure 5.4 of the textbook. Discuss your ndings. What degree polynomial is best, and why?

```r
library(ggplot2)
library(boot)
MSY <- data.frame(msy = numeric(), poly = numeric(), attempt = numeric())
counter <- 0
for(i in 1:9){
  set.seed(i)

  for(j in 1:10){
    counter <- counter+1
  #train <- sample(nrow(AutoDataPoly), nrow(AutoDataPoly)/2)
  AutoDataPolyLM <- glm(mpg~poly(horsepower,j), data = AutoDataPoly)
  MSY[counter,] <- c(cv.glm(AutoDataPoly, AutoDataPolyLM, K=10)$delta[1],j,i)
  #MSY[counter,] <- c(mean((AutoDataPoly$mpg -predict(AutoDataPolyLM,AutoDataPoly))[-train]^2),j,i)
  }
}
MSY$attempt <- as.factor(MSY$attempt)
ggplot(MSY, aes(y=msy, x = poly, color = attempt)) +
  geom_line() +
  ggtitle("MSY of Multiple Polynomials Of K-Fold Cross-Validation Approach") +
  xlab("Degree of Polynomial")+
  ylab("Mean Squared Error")
```



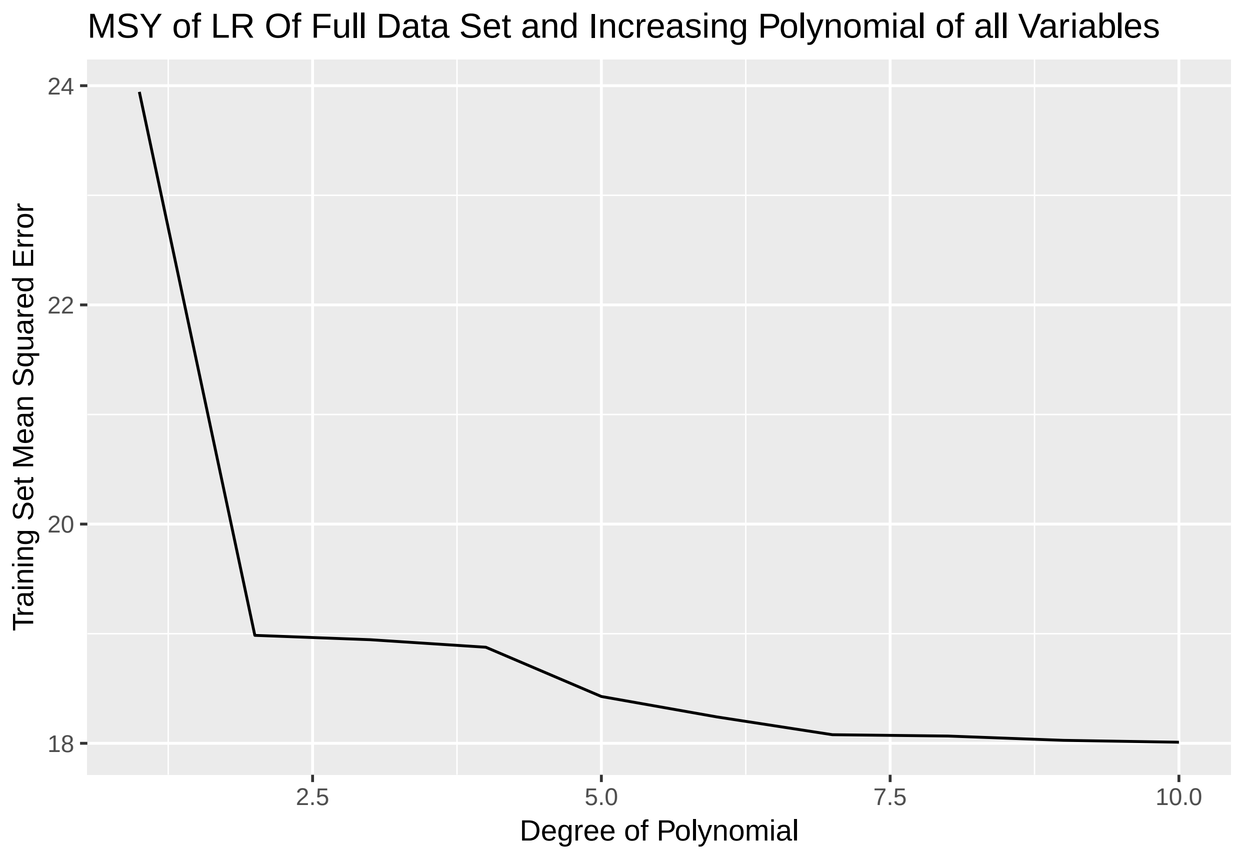
```
MeanMSY <- aggregate(msy ~ poly, MSY, mean)[2]
which.min(MeanMSY$msy)
```

```
## [1] 7
```

The mean MSE of the 7th polynomial performed the best of our 9 attempts.

(d) Fit a least squares linear model to predict mpg using polynomials of degrees from 1 to 10, using all available observations. Make a plot showing Degree of Polynomial" on the x-axis, and Training Set Mean Squared Error" on the y-axis. Discuss your ndings.

```r
library(ggplot2)
MSY <- data.frame(msy = numeric(), poly = numeric())
for(i in 1:10){
  AutoDataPolyLM <- glm(mpg~poly(horsepower,i), data = AutoDataPoly)
  MSY[i,] <- c(mean((AutoDataPoly$mpg -predict(AutoDataPolyLM,AutoDataPoly))^2),i)

}

ggplot(MSY, aes(y=msy, x = poly)) +
  geom_line() +
  ggtitle("MSY of LR Of Full Data Set and Increasing Polynomial of all Variables") +
  xlab("Degree of Polynomial")+
  ylab("Training Set Mean Squared Error")
```



```r
MeanMSY <- aggregate(msy ~ poly, MSY, mean)[2]
which.min(MeanMSY$msy)
```

```
## [1] 10
```

As the polynomial increases, so does the fit. So the MSE on the data decreases. Without the use of a validation approach, the result steadily improves MSE as the fit becomes tighter.

(e) Fit a least squares linear model to predict mpg using a degree-10 polynomial, using all available observations. Using the summary command in R, examine the output. Comment on the output, and discuss how this relates to your ndings in (a)(d).

```r
i<- 10
AutoDataPolyLM <- glm(mpg~poly(horsepower,i), data = AutoDataPoly)
summary(AutoDataPolyLM)
```

```
##
## Call:
## glm(formula = mpg ~ poly(horsepower, i), data = AutoDataPoly)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -15.7081   -2.5904   -0.1922    2.2859   14.8338
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)             23.4459     0.2174 107.840   <2e-16 ***
## poly(horsepower, i)1  -120.1377     4.3046 -27.909   <2e-16 ***
## poly(horsepower, i)2    44.0895     4.3046  10.242   <2e-16 ***
## poly(horsepower, i)3    -3.9488     4.3046  -0.917   0.3595
## poly(horsepower, i)4    -5.1878     4.3046  -1.205   0.2289
## poly(horsepower, i)5    13.2722     4.3046   3.083   0.0022 **
## poly(horsepower, i)6    -8.5462     4.3046  -1.985   0.0478 *
## poly(horsepower, i)7     7.9806     4.3046   1.854   0.0645 .
## poly(horsepower, i)8     2.1727     4.3046   0.505   0.6140
## poly(horsepower, i)9    -3.9182     4.3046  -0.910   0.3633
## poly(horsepower, i)10   -2.6146     4.3046  -0.607   0.5440
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 18.52949)
##
##     Null deviance: 23819.0  on 391  degrees of freedom
## Residual deviance:  7059.7  on 381  degrees of freedom
## AIC: 2269.7
##
## Number of Fisher Scoring iterations: 2
```

The output from the polynomial regression shows the probability that the chance of the polynomial of a given level (<= 10) is a better fit to the data. Our output sugget that 1st through 3rd order polynomials, 5th, 6th, and possibly 7th order polynomials of horsepower are a good fit for our data. These observations align with our observations in parts a-c. In A and B, our suggested best model uses the 3rd order polynomial, an order strongly suggested in the above summary results. In C, the 7th order is suggested, which not strongly supported in the above summary data, save that a higher order polynomial, such as 5 and possibly 6, is. The results of d are not trully comparable as no validation method was used. So the highest order polynomial would always be the best fit (of the training data only).

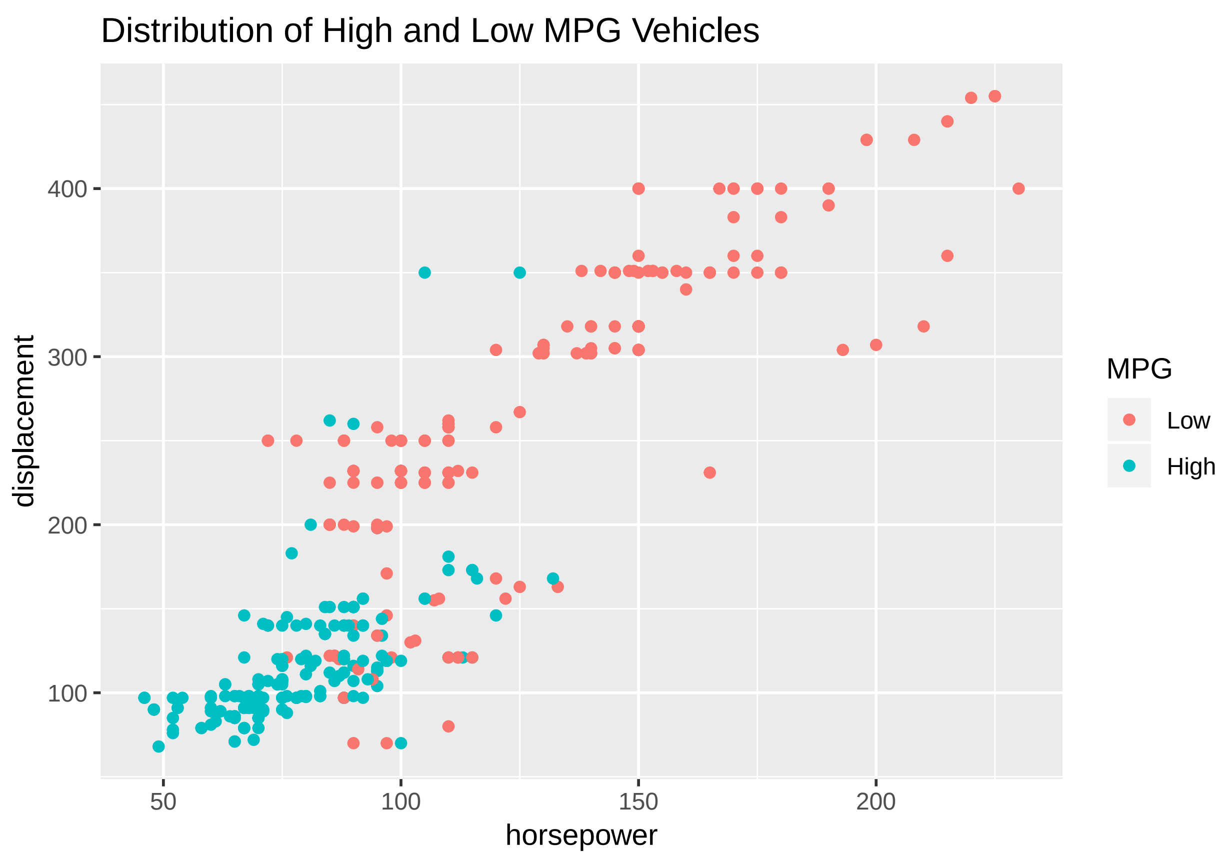
4. We will now continue with the Auto data set. Note that the R package class contains the knn function, which can be used to perform k-nearest neighbors classication.

(a) Create a binary variable, HighMPG, that equals 1 if a car's gas mileage is above the median in the Auto data set, and equals 0 if the car's gas mileage is below the median.

```r
library(ISLR)

FourthAutoData <- ISLR::Auto

FourthAutoData$HighMPG <- 0
FourthAutoData[FourthAutoData$mpg > median(FourthAutoData$mpg),]$HighMPG <- 1

AutoData$yearBinary <- "older"
AutoData[AutoData$year > 76,]$yearBinary <- "newer"
AutoData$yearBinary <- as.factor(AutoData$yearBinary)
AutoData$origin <- as.factor(AutoData$origin)
AutoData$cylinders <- as.factor(AutoData$cylinders)
AutoData$year <- NULL
AutoData$name <- NULL
```

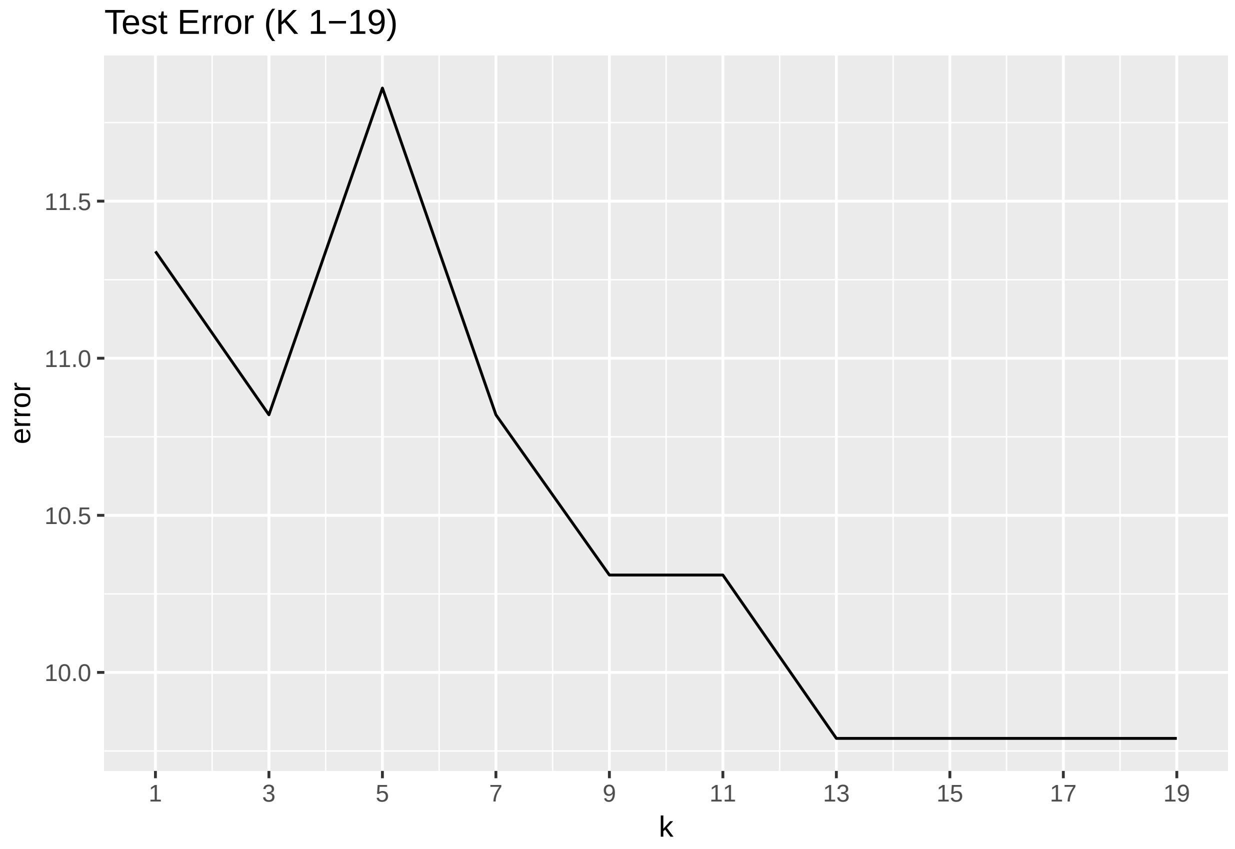
(b) Make a plot with horsepower on the x-axis, displacement on the y-axis, and with each of the cars in the Auto data set displayed as a point. The cars with gas mileage above the median should be displayed in one color, and the cars with gas mileage below the median should be displayed in another color. Be sure to create a legend and to label the axes appropriately.

```r
library(ggplot2)


ggplot(FourthAutoData, aes(x = horsepower, y = displacement, color = as.factor(HighMPG))) +
  geom_point() +
  labs(color = "MPG")+
  scale_color_hue(labels = c("Low", "High")) +
  ggtitle("Distribution of High and Low MPG Vehicles")
```



(c) Use the validation set approach in order to estimate the test error of knearest neighbors classication, when using horsepower and displacement to predict HighMPG. Since this is a classication problem, you can dee test error as the fraction of test set observations that are incorrectly classied. Make a plot of the estimated test error, as a function of k. Whatvalue of k gives you the smallest estimated test error? Comment on your results.

```
library(class)
set.seed(100)
FourthAutoData$train <- sample(0:1, nrow(FourthAutoData), replace = TRUE)
calculatedError <- data.frame(error = rep(0,10), k =  1:10)
kcounter <- 1
for( i in 0:9) {
  kvalue <- kcounter + i*2
KNNprediction <- knn(FourthAutoData[FourthAutoData$train == 1, c("horsepower", "displacement")],
    FourthAutoData[FourthAutoData$train == 0, c("horsepower", "displacement")],
    FourthAutoData[FourthAutoData$train == 1, "HighMPG"],
    k = kvalue)
calculatedError[i+1,] <- c(round(1 - mean(KNNprediction ==FourthAutoData[FourthAutoData$train == 0, "Hi
}

ggplot(calculatedError, aes(x = k, y = error)) +
  geom_line() +
  scale_x_continuous(breaks = seq(1, 19, 2)) +
  ggtitle("Test Error (K 1-19)")
```

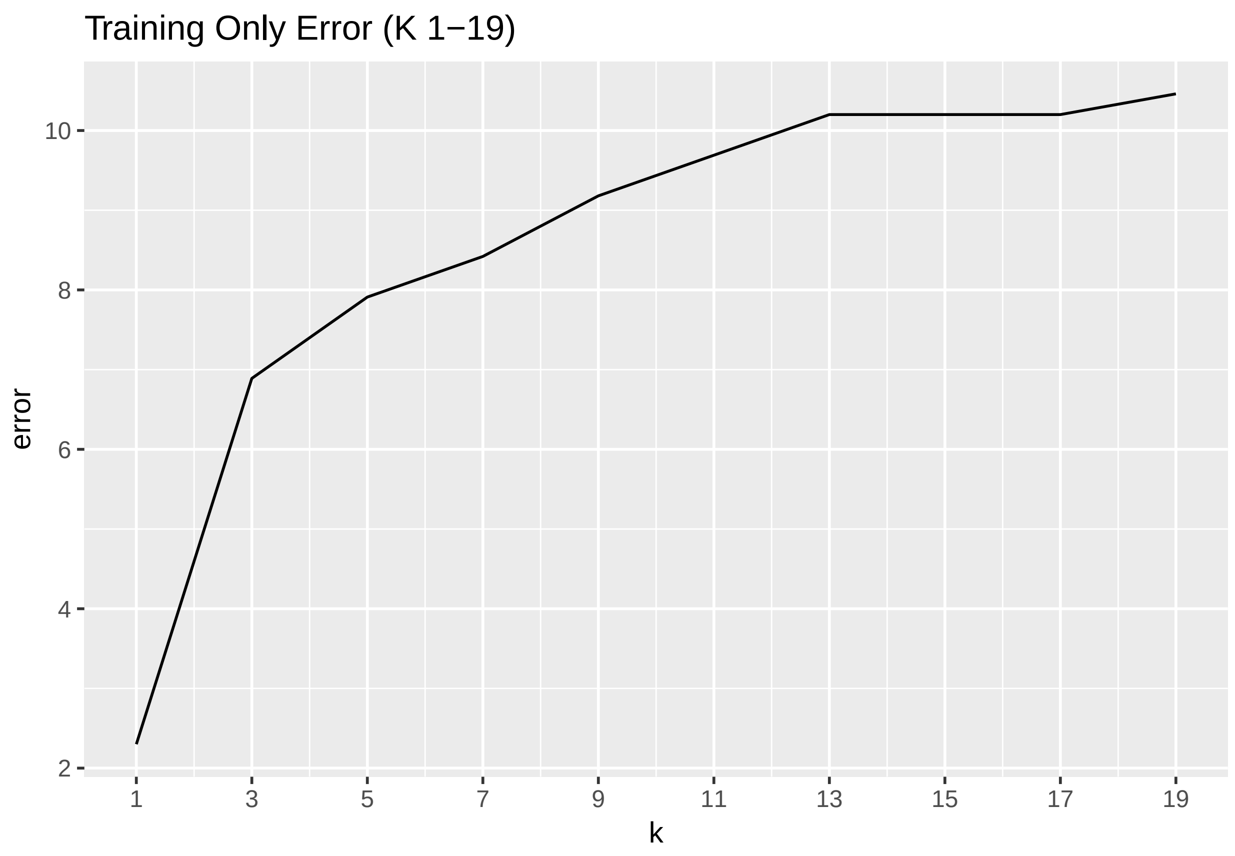


K of 7 to 19 (the highest tested) all performed the same, with an error rate of 9.79. These wereour best performing models.

Each progressive K level increases the number of observations grouped together. Displacement has a slightly larger range, and so more effect on the outcome of KNN. No improvement in predictive error was found from K=13 and up. The models with the lowest fits performed best against the test data.

(d) Now perform k-nearest neighbors regression on the full data set, for various values of k. Make a plot displaying the training error rate obtained, as a function of k, for the same values of k considered in (c). Comment on your results, and discuss how they relate to your ndings in (c). Hint: In (c), make sure to consider an appropriate range of values for k! I'd like to see values of k that are too small" (in terms of estimated test error) and also values of k that are too large".

```
library(class)

calculatedError <- data.frame(error = rep(0,10), k =  1:10)
kcounter <- 1
for( i in 0:9) {
  kvalue <- kcounter + i*2
KNNprediction <- knn(FourthAutoData[,c("horsepower", "displacement")],
    FourthAutoData[,c("horsepower", "displacement")],
    FourthAutoData[ ,"HighMPG"],
    k = kvalue)
calculatedError[i+1,] <- c(round(1 - mean(KNNprediction ==FourthAutoData[,"HighMPG"]),4) *100, kvalue)
}

ggplot(calculatedError, aes(x = k, y = error)) +
  geom_line() +
  scale_x_continuous(breaks = seq(1, 19, 2)) +
  ggtitle("Training Only Error (K 1-19)")
```



Our KNN using all data to predict all data results in a higher and higher error rate as KNN becomes larger. This is because, using the same data, we are not measuring bias, but instead the increase in our variance. Larger K's results in less accurate predictions as the predictions become more generalized from the actual data. By comparison, with a training and test set, the generalization away from the training set allows for some of the difference between the training and test data to improve accuracy. Notably, the error of the worst performing models in C are similar to the middling and best performing models in B, but the best performing models in B perform slightly better (K >= 13) than the worst performing models in C. Unspririsingly, the highest fitting model in C (K=1) and several other models (up to K=9) outperform all models in B. This would be a results of overfitting to our training data and drawing predictions on that same data.

5. Prove the following claim: The (training) RSS of the model y = 0 + is greater than or equal to the (training) RSS of the model = 0 + 1X + : 3

The formula for RSS $y = \beta_0 + \beta_1 X_1 + \epsilon$

$$RSS = \sum_{i=1}^{N}(Y_i - \hat{\beta}_0 - \hat{\beta}_1 X_i)^2$$

and for $y = \beta_0 + \epsilon$ is:

$$RSS = \sum_{i=1}^{N}(Y_i - \hat{\beta}_0)^2$$

The former is a subset of the later. However, the uncorrelated $\hat{\beta}_1$ of the former wil tend to 0 if uncorrelated, and if correlated, will tend to reduce $Y_i - \hat{\beta}_0 - \hat{\beta}_1 X_i$ or to move towards the less correlated, so larger RSS of $Y_i - \hat{\beta}_0 + 0$ or $Y_i - \hat{\beta}_0$