PSTAT 131 Homework Assignment 1

Marissa Santiago and Leticia Cruz

April 16, 2021

Prob1em 1

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(readr)
library(dplyr)
library(ISLR)
library(ggplot2)
library(plyr)
```

```
## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:purrr':
##
##     compact
```

```
algae <- read_table2("algaeBloom.txt", col_names=
               c('season','size','speed','mxPH','mnO2','Cl','NO3','NH4','oPO4','PO4','Chla','a1','a2',
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   season = col_character(),
##   size = col_character(),
##   speed = col_character(),
##   mxPH = col_double(),
##   mnO2 = col_double(),
##   Cl = col_double(),
##   NO3 = col_double(),
##   NH4 = col_double(),
##   oPO4 = col_double(),
##   PO4 = col_double(),
##   Chla = col_double(),
##   a1 = col_double(),
##   a2 = col_double(),
##   a3 = col_double(),
##   a4 = col_double(),
##   a5 = col_double(),
##   a6 = col_double(),
##   a7 = col_double()
## )
```

```
glimpse(algae)
```

```
## Rows: 200
## Columns: 18
## $ season <chr> "winter", "spring", "autumn", "spring", "autumn", "winter", "su~
## $ size   <chr> "small", "small", "small", "small", "small", "small", "small", ~
## $ speed  <chr> "medium", "medium", "medium", "medium", "medium", "high", "high~
## $ mxPH   <dbl> 8.00, 8.35, 8.10, 8.07, 8.06, 8.25, 8.15, 8.05, 8.70, 7.93, 7.7~
## $ mnO2   <dbl> 9.8, 8.0, 11.4, 4.8, 9.0, 13.1, 10.3, 10.6, 3.4, 9.9, 10.2, 11.~
## $ Cl     <dbl> 60.80, 57.75, 40.02, 77.36, 55.35, 65.75, 73.25, 59.07, 21.95, ~
## $ NO3    <dbl> 6.238, 1.288, 5.330, 2.302, 10.416, 9.248, 1.535, 4.990, 0.886,~
## $ NH4    <dbl> 578.00, 370.00, 346.67, 98.18, 233.70, 430.00, 110.00, 205.67, ~
## $ oPO4   <dbl> 105.00, 428.75, 125.67, 61.18, 58.22, 18.25, 61.25, 44.67, 36.3~
## $ PO4    <dbl> 170.00, 558.75, 187.06, 138.70, 97.58, 56.67, 111.75, 77.43, 71~
## $ Chla   <dbl> 50.000, 1.300, 15.600, 1.400, 10.500, 28.400, 3.200, 6.900, 5.5~
## $ a1     <dbl> 0.0, 1.4, 3.3, 3.1, 9.2, 15.1, 2.4, 18.2, 25.4, 17.0, 16.6, 32.~
## $ a2     <dbl> 0.0, 7.6, 53.6, 41.0, 2.9, 14.6, 1.2, 1.6, 5.4, 0.0, 0.0, 0.0, ~
## $ a3     <dbl> 0.0, 4.8, 1.9, 18.9, 7.5, 1.4, 3.2, 0.0, 2.5, 0.0, 0.0, 0.0, 2.~
## $ a4     <dbl> 0.0, 1.9, 0.0, 0.0, 0.0, 0.0, 3.9, 0.0, 0.0, 2.9, 0.0, 0.0, 0.0~
## $ a5     <dbl> 34.2, 6.7, 0.0, 1.4, 7.5, 22.5, 5.8, 5.5, 0.0, 0.0, 1.2, 0.0, 1~
## $ a6     <dbl> 8.3, 0.0, 0.0, 0.0, 4.1, 12.6, 6.8, 8.7, 0.0, 0.0, 0.0, 0.0, 0.~
## $ a7     <dbl> 0.0, 2.1, 9.7, 1.4, 1.0, 2.9, 0.0, 0.0, 0.0, 1.7, 6.0, 1.5, 2.1~
```

a)Number of observations for each season

```
algae %>%
  dplyr::group_by(season) %>%
  dplyr::summarise(n = n())
```

```
## # A tibble: 4 x 2
##   season     n
##   <chr>  <int>
## 1 autumn    40
## 2 spring    53
## 3 summer    45
## 4 winter    62
```

From the data we see above, we can see the total count of the observations are:
• Autumn = 40
• Spring = 53
• Summer = 45
• Winter = 62
b)

```
algae %>%
  summarise_at(.vars=vars(mnO2:Chla),.funs=funs(mean(.,na.rm=TRUE),
                                                var(.,na.rm=TRUE)))%>%
  t()
```

```
## Warning: `funs()` was deprecated in dplyr 0.8.0.
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
```

```
##                 [,1]
## mnO2_mean 9.118e+00
## Cl_mean   4.364e+01
## NO3_mean  3.282e+00
## NH4_mean  5.013e+02
## oPO4_mean 7.359e+01
## PO4_mean  1.379e+02
## Chla_mean 1.397e+01
## mnO2_var  5.718e+00
## Cl_var    2.193e+03
## NO3_var   1.426e+01
## NH4_var   3.852e+06
## oPO4_var  8.306e+03
## PO4_var   1.664e+04
## Chla_var  4.201e+02
```

We can confirm that there are missing values within the dataset using the function "is.na()" with the total count being 33.

- I used the length function to see how many counts of "TRUE" there were in the "missing" subset.
- The mean and variance is shown above as well using the *colMeans* function as well as *colvars* from a pacakge 'Resample' that I have learned from another class.
- For both the cases of mean and variance, the missing values were ignored.

The variance for the majority of the chemicals are much greater than their mean values which can indicate that the data points are very spread out from the average.

c)

```
algae %>%
  summarise_at(.vars=vars(mnO2:Chla), .funs=funs(median(.,na.rm=TRUE),mad(.,na.rm=TRUE))) %>%
  t()
```

```
##                 [,1]
## mnO2_median    9.800
## Cl_median     32.730
## NO3_median     2.675
## NH4_median   103.166
## oPO4_median   40.150
## PO4_median   103.285
## Chla_median    5.475
## mnO2_mad       2.053
## Cl_mad        33.250
## NO3_mad        2.172
## NH4_mad      111.618
## oPO4_mad      44.046
## PO4_mad      122.321
## Chla_mad       6.672
```

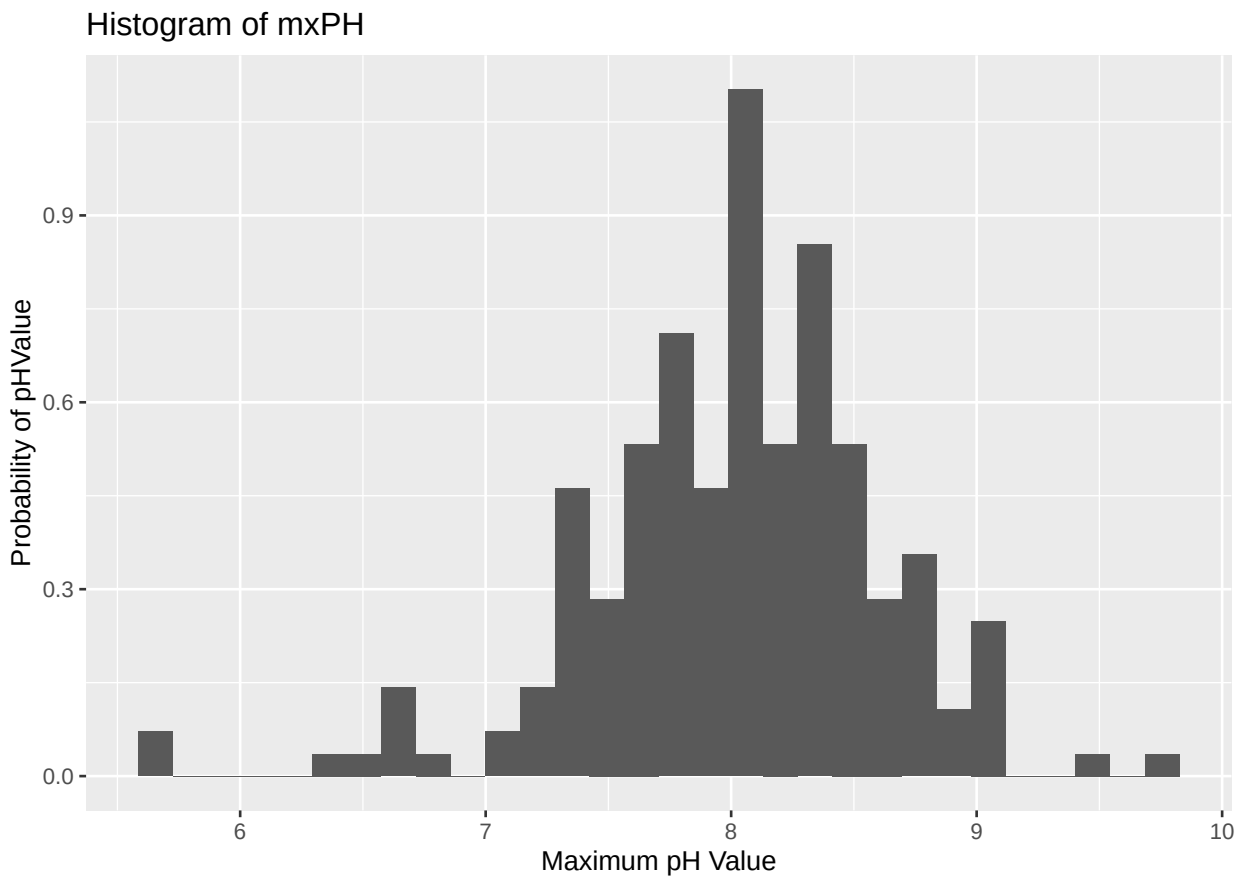
2. Construct box-plots, histograms, QQ-plots and kernel density estimates for these variables. Comment on features such as the distribution and outliers in these plots.
   When asked to construct a graph, you should always precede your graph by the R command/function that generated it properly annotated.

a) The histogram of the variable mxPH

```
algae %>%
  ggplot(aes(x=mxPH)) + geom_histogram(aes(y=..density..))+
  ggtitle("Histogram of mxPH") + ylab("Probability of pHValue")+
  xlab("Maximum pH Value")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing non-finite values (stat_bin).
```

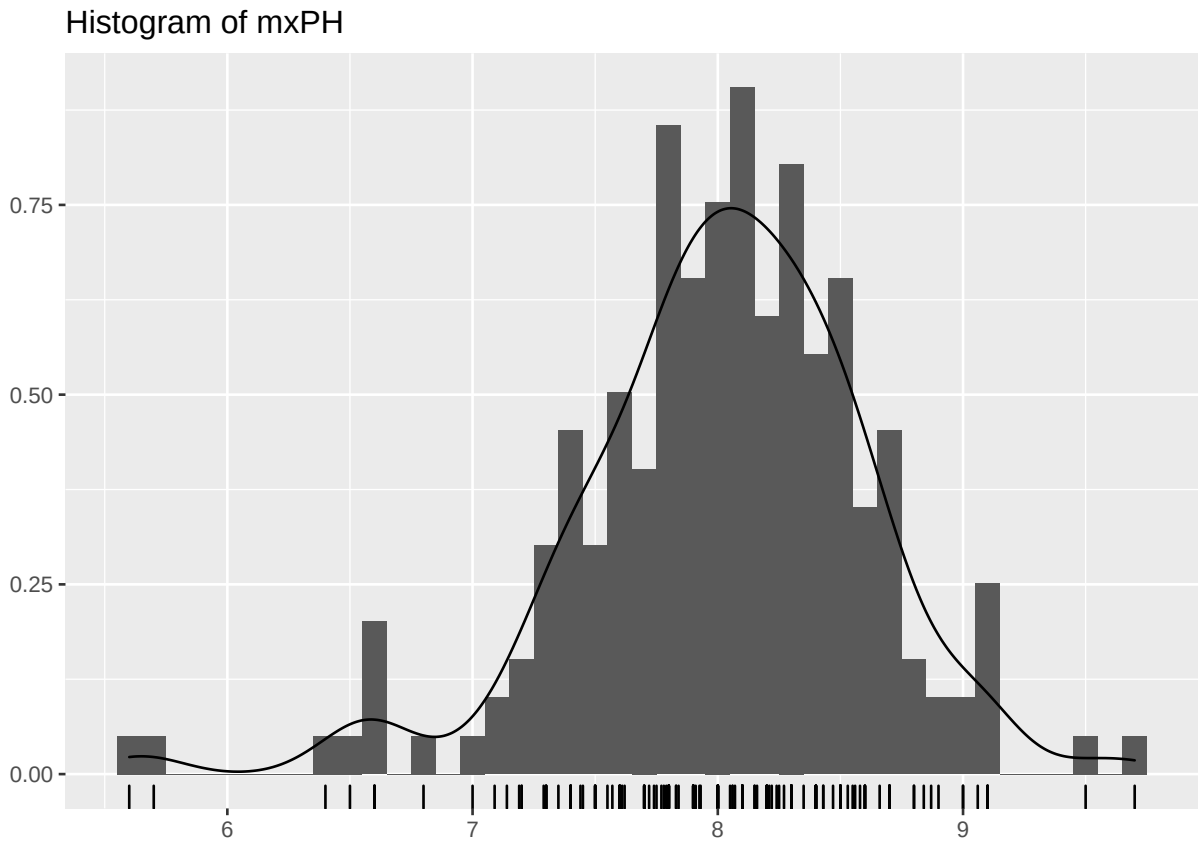


The values of variable mxPH follow a near normal distribution, with the values clustered around the mean value.

b)

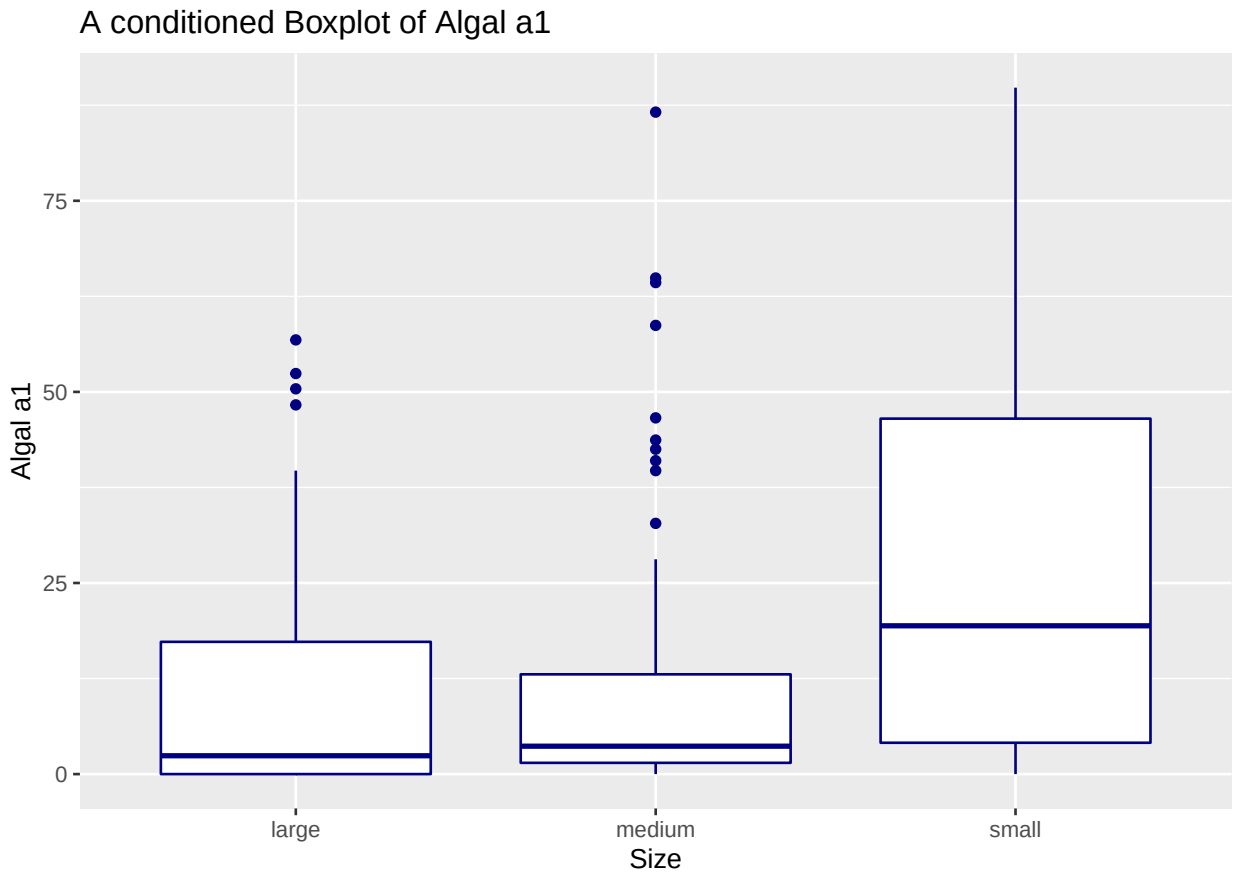
```
algae %>%
  ggplot(aes(mxPH))+
  geom_histogram(aes(y=..density..),binwidth=.1, na.rm=TRUE) +    geom_density() + geom_rug() + ggtitle
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density).
```



c) boxplot for algae a1

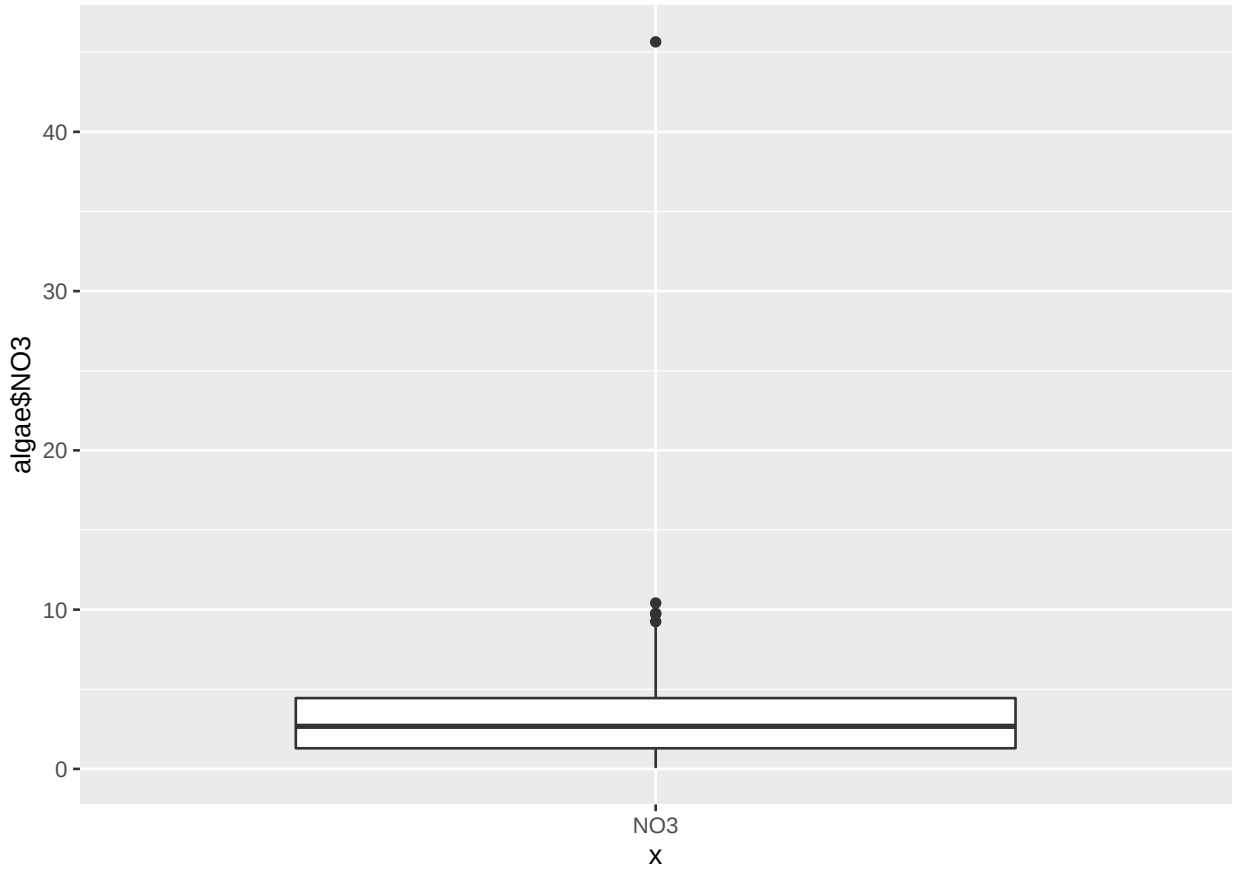
```
algae %>%
  ggplot(aes(y=a1,x=size)) +
  geom_boxplot(col = "navyblue") + ggtitle("A conditioned Boxplot of Algal a1") + xlab("Size") + ylab("
```



d) Outliers for NO3 and NH4

```
algae %>% ggplot(aes(x="NO3",y=algae$NO3)) + geom_boxplot()
```

```
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
```

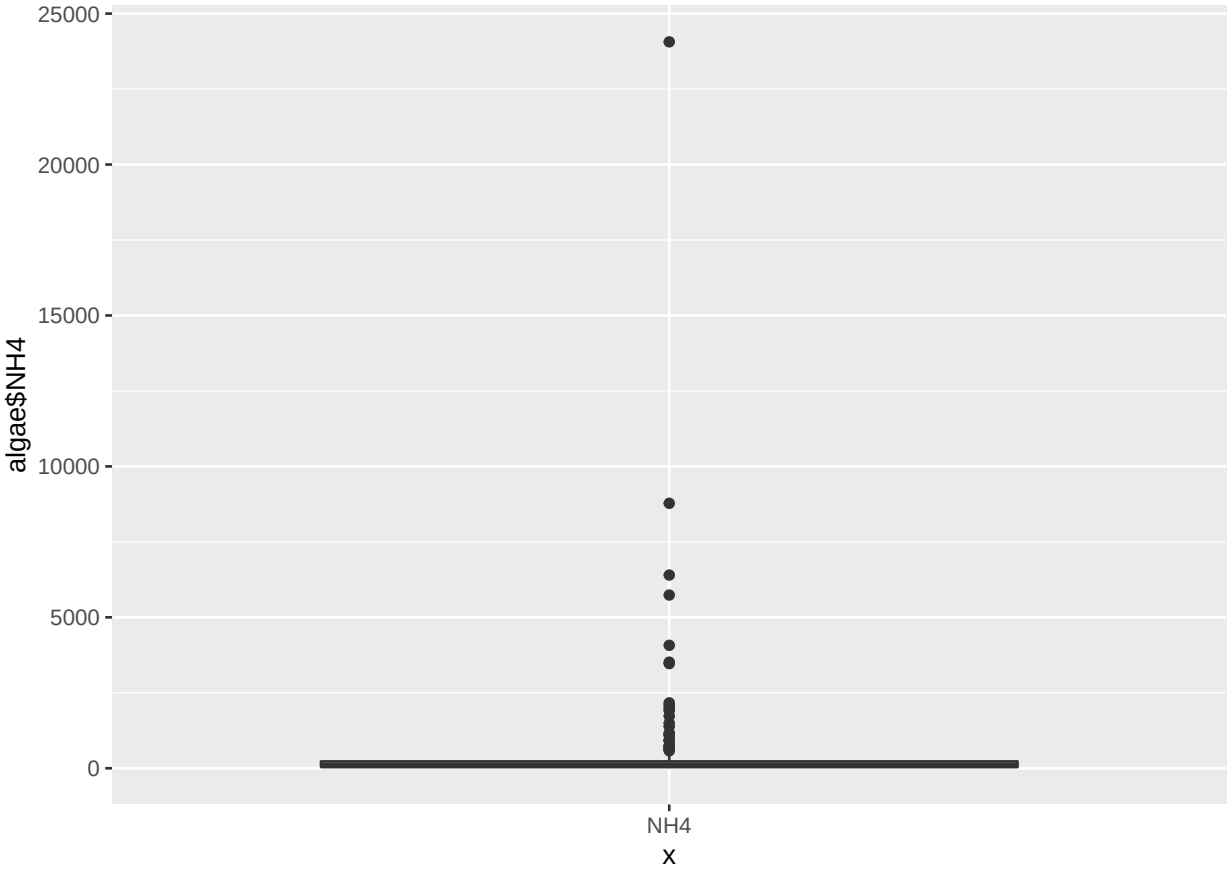



```
algae %>%
  filter(NO3 > (quantile(NO3, 0.75, na.rm=T) + 1.5*IQR(NO3,    na.rm=TRUE))) %>%
  select(NO3)
```

```
## # A tibble: 5 x 1
##     NO3
##   <dbl>
## 1 10.4
## 2  9.25
## 3  9.77
## 4  9.72
## 5 45.6
```

```
algae %>% ggplot(aes(x="NH4",y=algae$NH4)) + geom_boxplot()
```

```
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
```




```
algae %>%
  filter(NH4 > (quantile(NH4, 0.75, na.rm =TRUE) + 1.5*IQR(NH4, na.rm=TRUE))) %>%
  select(NH4)
```

```
## # A tibble: 27 x 1
##      NH4
##    <dbl>
##  1  578
##  2 8778.
##  3 1729
##  4 3515
##  5 6400
##  6 1911
##  7  648.
##  8 1386.
##  9 2083.
## 10 2167.
## # ... with 17 more rows
```

From the results we can see that there are 5 outliers for NO_3 and 27 outliers for NH_4. After viewing the boxplot graphs and confirming that there are outliers, we used the IQR function to determine what the outliers are.

e)

```
cat('The mean of NO3 =', mean(algae$NO3, na.rm = TRUE), '\n', 'The variance of NO3 =', var(algae$NO3, na
```

```
## The mean of NO3 = 3.282
##  The variance of NO3 = 14.26
```

```
cat('The median of NO3 =', median(algae$NO3, na.rm = TRUE), '\n', 'The MAD of NO3 =', mad(algae$NO3, na
```

```
## The median of NO3 = 2.675
##  The MAD of NO3 = 2.172
```

```
cat('The mean of NH4 =', mean(algae$NH4, na.rm = TRUE), '\n', 'The variance of NH4 =', var(algae$NH4, na
```

```
## The mean of NH4 = 501.3
##  The variance of NH4 = 3851585
```

```
cat('The median of NH4 =', median(algae$NH4, na.rm = TRUE), '\n', 'The MAD of NH4 =', mad(algae$NH4, na
```

```
## The median of NH4 = 103.2
##  The MAD of NH4 = 111.6
```

It appears that both the variance of NO3 and NH4 is significantly larger than the mean. When we look at the values for the median and MAD we can see that they are very similar. In conclusion, the median and MAD values are not influenced much from the outlier which means it is suceptible to skewing the data when there are extreme outliers in the data.

It appears that median and Mad tend to hold up more to outliers, this is caused because, by using the mean, it is suceptible to skewing the data when there are extreme outliers in the data.

Problem 3 a) Observations contain missing values

```
algae %>%
  summarise_at(.vars=vars(season:a7),.funs=funs(sum(is.na(.))))
```

```
## # A tibble: 1 x 18
##   season  size speed  mxPH  mnO2    Cl   NO3   NH4  oPO4   PO4  Chla    a1    a2
##    <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
## 1      0     0     0     1     2    10     2     2     2     2    12     0     0
## # ... with 5 more variables: a3 <int>, a4 <int>, a5 <int>, a6 <int>, a7 <int>
```

We can see that every predictor have missing values starting from mxPH to Chla. From this we can see that *mxPH* has 1 missing value, *mnO2* has 2 missing values, *Cl* has 10 missing values, *NO3* has 2 missing values, *NH4* has 2 missing values, *OPO4* has 2 missing values, *PO4* has 2 missing values, and *Chla* has 12 missing values. This brings us to a total of 33 missing values.

b) Removing observations with missing values

```
algae.del = filter(algae[complete.cases(algae),])
nrow(algae.del)
```

```
## [1] 184
```

There are a total of *184 observations* in algae.del.

c) Filling in the Unknowns with the Most Frequent Values

```
algae.med <- algae %>%
  mutate_at(vars(4:11), funs(ifelse(is.na(.) == TRUE, median(., na.rm=TRUE), .)))
nrow(algae.med)
```

```
## [1] 200
```

```
algae.med[48,5:11]
```

```
## # A tibble: 1 x 7
##    mnO2    Cl   NO3   NH4  oPO4   PO4  Chla
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  12.6     9  0.23    10     5     6   1.1
```

```
algae.med[62,5:11]
```

```
## # A tibble: 1 x 7
##    mnO2    Cl   NO3   NH4  oPO4   PO4  Chla
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1   9.8  32.7  2.68  103.  40.2    14  5.48
```

```
algae.med[199,5:11]
```

```
## # A tibble: 1 x 7
##    mnO2    Cl   NO3   NH4  oPO4   PO4  Chla
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1   7.6  32.7  2.68  103.  40.2  103.  5.48
```

There are a total or *200 observations* in algae.med

d) Imputing unknowns using correlations

```
df = data.frame(algae.del[, 4:11])
cor(df, use = "pairwise.complete.obs" )
```

```
##          mxPH     mnO2       Cl     NO3      NH4     oPO4     PO4    Chla
## mxPH  1.00000 -0.10269  0.14710 -0.1721 -0.15430  0.09023  0.1013  0.4318
## mnO2 -0.10269  1.00000 -0.26325  0.1179 -0.07827 -0.39375 -0.4640 -0.1312
## Cl    0.14710 -0.26325  1.00000  0.2110  0.06598  0.37926  0.4452  0.1430
## NO3  -0.17213  0.11791  0.21096  1.0000  0.72468  0.13301  0.1570  0.1455
## NH4  -0.15430 -0.07827  0.06598  0.7247  1.00000  0.21931  0.1994  0.0912
## oPO4  0.09023 -0.39375  0.37926  0.1330  0.21931  1.00000  0.9120  0.1069
## PO4   0.10133 -0.46396  0.44519  0.1570  0.19940  0.91196  1.0000  0.2485
## Chla  0.43182 -0.13122  0.14296  0.1455  0.09120  0.10691  0.2485  1.0000
```

```
model = lm(PO4~oPO4, data = algae)
summary(model)
```

```
##
## Call:
## lm(formula = PO4 ~ oPO4, data = algae)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -110.1  -36.3  -12.7   23.3  217.0
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   42.897      4.808    8.92  3.3e-16 ***
## oPO4           1.293      0.041   31.54  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.4 on 195 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.836,  Adjusted R-squared:  0.835
## F-statistic:  994 on 1 and 195 DF,  p-value: <2e-16
```

```
predict(model,algae[28,"oPO4"])
```

```
##     1
## 48.07
```

```
#filling in the missing value in 'algae'
algae[28,"PO4"] = predict(model,algae[28,"oPO4"])
```

The value that we obtain for the missing value for PO4 based on oPO4 in the 28th observation is 48.07.

e) Imputation using only observed data can lead us to incorrect conclusions because this could add more bias to our data. This is because we are imputing values based off data that was already found.For example, replacing all missing values with the median or correlation of all values for all chemical parameters would skew the data and would give us innacurate results. In lecture 2; we learned that this is referred to as survivorship bias.

Problem 4 a)

```
nfold = 5
set.seed(66)
cv = sample(cut(1:nrow(algae.med), breaks=nfold, labels=FALSE))
cv
```

```
##   [1] 3 3 5 4 1 4 5 3 3 2 1 4 1 4 1 2 3 3 5 5 3 2 1 3 5 2 4 3 5 2 1 4 4 2 4 3 4
##  [38] 4 3 1 2 4 1 5 4 2 5 2 2 1 2 5 4 3 5 1 5 1 1 2 2 2 2 1 4 2 3 4 4 1 3 4 4 5
##  [75] 4 5 1 2 2 3 1 5 5 1 1 1 4 5 2 3 1 4 3 5 1 2 3 4 5 5 1 1 5 5 5 3 5 4 4 3 3
## [112] 5 2 3 4 1 3 2 3 5 5 5 4 1 2 3 3 5 2 3 2 1 2 3 4 4 3 2 3 1 2 1 5 5 2 1 1 4
## [149] 4 2 5 3 4 5 1 2 1 4 2 3 2 3 3 1 5 4 3 5 4 1 1 4 2 4 4 1 1 5 4 3 2 3 3 1 2
## [186] 2 1 3 5 5 4 5 1 2 3 5 2 5 4 2
```

b) Perform 5-fold cross-validation

```
do.chunk <- function(chunkid, chunkdef, dat){ # function argument
  train = (chunkdef != chunkid)

  Xtr = dat[train,1:11] # get training set
  Ytr = dat[train,12] # get true response values in trainig set

  Xvl = dat[!train,1:11] # get validation set
  Yvl = dat[!train,12] # get true response values in validation set

  lm.a1 <- lm(a1~., data = dat[train,1:12])
  predYtr = predict(lm.a1) # predict training values
  predYvl = predict(lm.a1,Xvl) # predict validation values
  data.frame(fold = chunkid,
    train.error = mean((predYtr - Ytr$a1)^2), # compute and store training error
    val.error = mean((predYvl - Yvl$a1)^2)) # compute and store test error
}
```

```
error.folds = NULL
allK = 1:5
set.seed(123)
for (j in allK){
    tmp = ldply(1:nfold, do.chunk, chunkdef=cv, dat=algae.med)
    error.folds = rbind(error.folds, tmp)
}
tmp
```

```
##   fold train.error val.error
## 1    1       290.4     285.4
## 2    2       240.6     506.6
## 3    3       296.3     256.5
## 4    4       281.0     400.1
## 5    5       299.8     257.6
```

Problem 5

```
algae.Test <- read_table2('algaeTest.txt', col_names=c('season','size','speed','mxPH','mnO2','Cl','NO3'
'NH4','oPO4','PO4','Chla','a1'), na=c('XXXXXXX'))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   season = col_character(),
##   size = col_character(),
##   speed = col_character(),
##   mxPH = col_double(),
##   mnO2 = col_double(),
##   Cl = col_double(),
##   NO3 = col_double(),
##   NH4 = col_double(),
##   oPO4 = col_double(),
```

```
##   PO4 = col_double(),
##   Chla = col_double(),
##   a1 = col_double()
## )
```

```
firstdata = algae.med[12]
newdata = algae.Test[12]
fit = lm(a1 ~ ., data = algae.med[1:12])
firstpredict = predict(fit, algae.med[1:11])
newpredict = predict(fit, algae.Test[1:11])
data.frame(train.error = mean((firstpredict - firstdata$a1)^2), val.error = mean((newpredict - newdata$a
```

```
##   train.error val.error
## 1       286.3     250.2
```

Yes, this is what is roughly expected based on the CV estimated test error from Question 4. The *train.error is 286.2661* which is very close to train.error predicted in Q.4. The *val.error is 250.1794* which is not close to any of the predicted val.error.

Problem 6

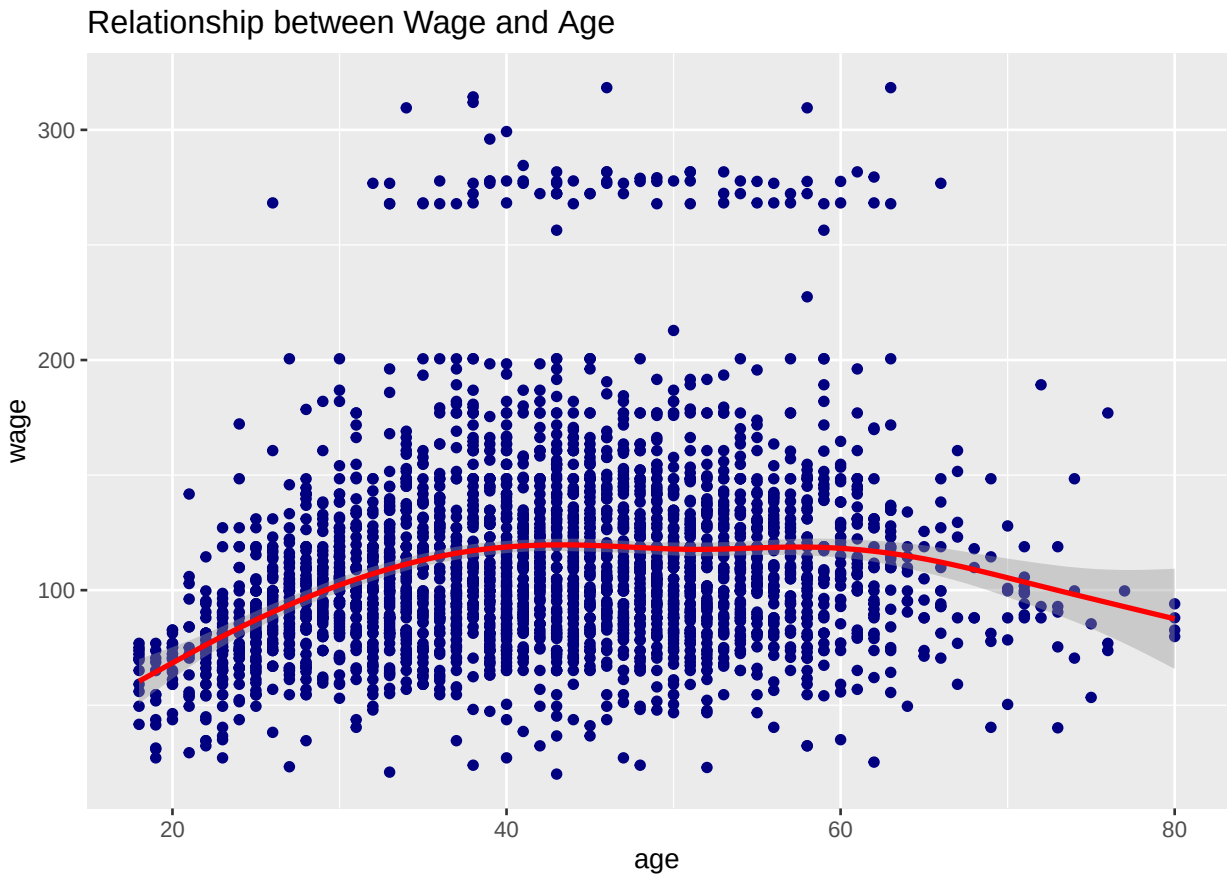
```
library(ISLR)
head(Wage)
```

```
##        year age           maritl     race       education             region
## 231655 2006  18 1. Never Married 1. White    1. < HS Grad 2. Middle Atlantic
## 86582  2004  24 1. Never Married 1. White 4. College Grad 2. Middle Atlantic
## 161300 2003  45       2. Married 1. White 3. Some College 2. Middle Atlantic
## 155159 2003  43       2. Married 3. Asian 4. College Grad 2. Middle Atlantic
## 11443  2005  50      4. Divorced 1. White      2. HS Grad 2. Middle Atlantic
## 376662 2008  54       2. Married 1. White 4. College Grad 2. Middle Atlantic
##              jobclass         health health_ins logwage   wage
## 231655  1. Industrial      1. <=Good      2. No   4.318  75.04
## 86582  2. Information 2. >=Very Good      2. No   4.255  70.48
## 161300  1. Industrial      1. <=Good     1. Yes   4.875 130.98
## 155159 2. Information 2. >=Very Good     1. Yes   5.041 154.69
## 11443  2. Information      1. <=Good     1. Yes   4.318  75.04
## 376662 2. Information 2. >=Very Good     1. Yes   4.845 127.12
```

a)

```
Wage %>%
  ggplot(aes(x = age, y = wage)) +
  geom_point(color='navyblue') +
  geom_smooth(color='red') +
  ggtitle("Relationship between Wage and Age")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



By looking at the plot; we really do not see a strong linear trend in pattern. This is similar to our expectations with wages steadily increasing until the age of 40. It then remains constant until about age 60. Following, we see that the wage goes down slowly until age 80 where the graph ends. This matches exactly what I expected because as time goes on from your twenties, we tend to work on our skills in higher education which give us better jobs/oportunities. This trend starts to decrease because once people retire they no longer have an increasing income; this is very apparent in the plot after age 60.

b. A polynomial function of age that best fits the wage data

```
for (p in c(0:10)) {
  if (p==0) {
    fit <-lm(Wage$wage~1)
  }
  else {
     fit <-lm(wage~poly(age, p, raw=F), data=Wage)
  }
 print(coef(summary(fit)))
}
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    111.7     0.7619   146.6        0
##                       Estimate Std. Error t value  Pr(>|t|)
## (Intercept)              111.7     0.7473  149.48 0.000e+00
## poly(age, p, raw = F)    447.1    40.9291   10.92 2.901e-27
##                        Estimate Std. Error t value  Pr(>|t|)
```

```
## (Intercept)               111.7     0.7302  152.98 0.000e+00
## poly(age, p, raw = F)1    447.1    39.9926   11.18 1.878e-28
## poly(age, p, raw = F)2   -478.3    39.9926  -11.96 3.077e-32
##                         Estimate Std. Error t value  Pr(>|t|)
## (Intercept)               111.7     0.7291 153.211 0.000e+00
## poly(age, p, raw = F)1    447.1    39.9335  11.195 1.571e-28
## poly(age, p, raw = F)2   -478.3    39.9335 -11.978 2.512e-32
## poly(age, p, raw = F)3    125.5    39.9335   3.143 1.687e-03
##                         Estimate Std. Error t value  Pr(>|t|)
## (Intercept)              111.70     0.7287 153.283 0.000e+00
## poly(age, p, raw = F)1   447.07    39.9148  11.201 1.485e-28
## poly(age, p, raw = F)2  -478.32    39.9148 -11.983 2.356e-32
## poly(age, p, raw = F)3   125.52    39.9148   3.145 1.679e-03
## poly(age, p, raw = F)4   -77.91    39.9148  -1.952 5.104e-02
##                         Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)              111.70     0.7288 153.2780 0.000e+00
## poly(age, p, raw = F)1   447.07    39.9161  11.2002 1.491e-28
## poly(age, p, raw = F)2  -478.32    39.9161 -11.9830 2.368e-32
## poly(age, p, raw = F)3   125.52    39.9161   3.1446 1.679e-03
## poly(age, p, raw = F)4   -77.91    39.9161  -1.9519 5.105e-02
## poly(age, p, raw = F)5   -35.81    39.9161  -0.8972 3.697e-01
##                         Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)              111.70     0.7286 153.3156 0.000e+00
## poly(age, p, raw = F)1   447.07    39.9063  11.2029 1.448e-28
## poly(age, p, raw = F)2  -478.32    39.9063 -11.9860 2.290e-32
## poly(age, p, raw = F)3   125.52    39.9063   3.1454 1.675e-03
## poly(age, p, raw = F)4   -77.91    39.9063  -1.9524 5.099e-02
## poly(age, p, raw = F)5   -35.81    39.9063  -0.8974 3.696e-01
## poly(age, p, raw = F)6    62.71    39.9063   1.5714 1.162e-01
##                         Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)              111.70     0.7285 153.3311 0.000e+00
## poly(age, p, raw = F)1   447.07    39.9023  11.2041 1.431e-28
## poly(age, p, raw = F)2  -478.32    39.9023 -11.9872 2.260e-32
## poly(age, p, raw = F)3   125.52    39.9023   3.1457 1.673e-03
## poly(age, p, raw = F)4   -77.91    39.9023  -1.9526 5.097e-02
## poly(age, p, raw = F)5   -35.81    39.9023  -0.8975 3.695e-01
## poly(age, p, raw = F)6    62.71    39.9023   1.5715 1.162e-01
## poly(age, p, raw = F)7    50.55    39.9023   1.2668 2.053e-01
##                         Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)              111.70     0.7286 153.3075 0.000e+00
## poly(age, p, raw = F)1   447.07    39.9084  11.2023 1.459e-28
## poly(age, p, raw = F)2  -478.32    39.9084 -11.9853 2.309e-32
## poly(age, p, raw = F)3   125.52    39.9084   3.1452 1.676e-03
## poly(age, p, raw = F)4   -77.91    39.9084  -1.9523 5.100e-02
## poly(age, p, raw = F)5   -35.81    39.9084  -0.8974 3.696e-01
## poly(age, p, raw = F)6    62.71    39.9084   1.5713 1.162e-01
## poly(age, p, raw = F)7    50.55    39.9084   1.2666 2.054e-01
## poly(age, p, raw = F)8   -11.25    39.9084  -0.2820 7.780e-01
##                         Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)              111.70     0.7282 153.3947 0.000e+00
## poly(age, p, raw = F)1   447.07    39.8857  11.2087 1.362e-28
## poly(age, p, raw = F)2  -478.32    39.8857 -11.9922 2.136e-32
## poly(age, p, raw = F)3   125.52    39.8857   3.1470 1.666e-03
## poly(age, p, raw = F)4   -77.91    39.8857  -1.9534 5.087e-02
```

```
## poly(age, p, raw = F)5   -35.81    39.8857  -0.8979 3.693e-01
## poly(age, p, raw = F)6    62.71    39.8857   1.5722 1.160e-01
## poly(age, p, raw = F)7    50.55    39.8857   1.2674 2.051e-01
## poly(age, p, raw = F)8   -11.25    39.8857  -0.2822 7.778e-01
## poly(age, p, raw = F)9   -83.69    39.8857  -2.0983 3.596e-02
##                         Estimate Std. Error   t value  Pr(>|t|)
## (Intercept)              111.704     0.7283 153.36910 0.000e+00
## poly(age, p, raw = F)1   447.068    39.8924  11.20685 1.390e-28
## poly(age, p, raw = F)2  -478.316    39.8924 -11.99015 2.187e-32
## poly(age, p, raw = F)3   125.522    39.8924   3.14651 1.669e-03
## poly(age, p, raw = F)4   -77.911    39.8924  -1.95303 5.091e-02
## poly(age, p, raw = F)5   -35.813    39.8924  -0.89774 3.694e-01
## poly(age, p, raw = F)6    62.708    39.8924   1.57192 1.161e-01
## poly(age, p, raw = F)7    50.550    39.8924   1.26715 2.052e-01
## poly(age, p, raw = F)8   -11.255    39.8924  -0.28213 7.779e-01
## poly(age, p, raw = F)9   -83.692    39.8924  -2.09794 3.599e-02
## poly(age, p, raw = F)10    1.624    39.8924   0.04071 9.675e-01
```

```
chunk1 <- sample(cut(1:nrow(Wage), breaks=5, label=FALSE))
do.chunk1 <- function(chunkid, chunkdef, dat, k){ # function argument
train = (chunkdef != chunkid)
tr = dat[train,] # get training set
test = dat[!train,] # get true response values in validation set
if(k==0) {
   lm.wage <- lm(wage~1, data=tr)
}
else {
  lm.wage=lm(wage~poly(age, degree=k, raw=FALSE), data=tr)
}
predYtr = predict(lm.wage) # predict training values
predYvl = predict(lm.wage, newdata=test) # predict validation values

data.frame(fold = chunkid,
train.error = mean((predYtr - tr$wage)^2), # compute and store training error
val.error = mean((predYvl - test$wage)^2)) # compute and store test error
}
error.chunk=NULL
Wage1 <- Wage %>%
  select(age, wage)

for (p in 0:10) {
  tmp = ldply(1:5, do.chunk1, chunkdef=chunk1, dat=Wage1, k=p)
  tmp$model=p
  #avg.train.error = mean(tmp$train.error)
  #avg.test.error= mean(tmp$val.error)
  error.chunk=rbind(error.chunk, tmp)
}
```

c)

```
mean.errors = error.chunk %>% select(-fold) %>%
  group_by(model) %>%
  mutate_at(.vars=vars(train.error,val.error),.funs=funs(mean))
```

```
val.e<-c(1742.098,1676.774,1601.178,1596.555,1594.413,
         1595.981,1594.424,1597.184,1597.676,1596.117,1596.718)
tr.e<-c(1740.539,1673.772,1597.435,1592.114,1590.103,1589.468,
        1588.168,1586.933,1586.824,1584.398,1584.334)
p1 <- c(0:10)
Avg.Error <- data.frame(Model=p1, ValError=val.e, TrainError=tr.e)
Avg.Error
```

```
##    Model ValError TrainError
## 1      0     1742       1741
## 2      1     1677       1674
## 3      2     1601       1597
## 4      3     1597       1592
## 5      4     1594       1590
## 6      5     1596       1589
## 7      6     1594       1588
## 8      7     1597       1587
## 9      8     1598       1587
## 10     9     1596       1584
## 11    10     1597       1584
```

```
min(Avg.Error$ValError)
```

```
## [1] 1594
```

```
ggplot()+geom_point(aes(x=p1, y=val.e), color="yellow") + geom_point(aes(x=p1, y=tr.e), color ="green")
  ggtitle("Plot of Average Training Error & Average Test Error") + xlab("Model") +ylab("Error")
```

As p increases, the models become more flexible so we can see that both the training and test error decrease.Both of these errors start to stabilize once we reach higher order polynomials.
The yellow points represent the test error and the green points represent the training error.
The best model to choose is model 4 with the fourth order polynomial. This model has the lowest validaton error. When printing out the errors, the values between Model 4 and 6 are indistinguishable because the values are rounded to the nearest whole number but when comparing values, Model 4 still has the lowest error.