Assignment 1

PSTAT 231

Villaseñor-Derbez J.C. | 8749749

```r
# Default options
knitr::opts_chunk$set(echo = TRUE,
                      message = F,
                      warning = F)

# Load packages
suppressPackageStartupMessages({
  library(here)
  library(tidyverse)
})
```

```r
algae <- read_table2(here("raw_data", "h1", "algaeBloom.txt"),
                     col_names = c('season','size','speed','mxPH','mnO2','Cl','NO3','NH4',
                                   'oPO4','PO4','Chla','a1','a2','a3','a4','a5','a6','a7'),
                     na = "XXXXXXX", col_types = cols())
```

1 Descriptive summary statistics

1.1 Count the number of observations in each season using `summarize()`

```r
algae %>%
  group_by(season) %>%
  summarize(n = n()) %>%
  knitr::kable(caption = "Number of oobservations in each season.")
```

1.2 Are there missing values? Calculate the mean and variance of each chemical (Ignore a1 through a7). What do you notice about the magnitude of the two quantities for different chemicals?

```r
#Number of observations with missing values
algae %>%
  select(1:12) %>%
  apply(FUN = function(x){any(is.na(x))}, MARGIN = 1) %>%
  sum()
```

```
## [1] 16
```

The above shows that 16 observations have missing values for at least one variable. The number of missing values for each variable are shown in the table below.

```r
# Use a lambda-like function to count the NAs in each varialbe
algae %>%
  select(1:12) %>%
  summarize_all(function(x){sum(is.na(x))}) %>%
```

Table 1: Number of oobservations in each season.

season	n
autumn	40
spring	53
summer	45
winter	62

Table 2: Number of missing values for each variable in the algae dataset.

variable	n_missing
season	0
size	0
speed	0
mxPH	1
mnO2	2
Cl	10
NO3	2
NH4	2
oPO4	2
PO4	2
Chla	12
a1	0

```r
  gather(variable, n_missing) %>%
  knitr::kable(caption = "Number of missing values for each variable in the algae dataset.")
```

Many variables have missing values, so I'll use the `na.rm = T` argument when summarizing the variables to get descriptive statistics.

```r
algae %>%
  select(-c(1:3, matches("a."))) %>%
  gather(variable, value) %>%
  group_by(variable) %>%
  summarize(mean = mean(value, na.rm = T),
            variance = var(value, na.rm = T)) %>%
  knitr::kable(caption = "Mean and variance for the eight variables of interest.")
```

These descriptive statistis show that the units in which the variables are measured result in very different scales. NH_4, for example, is in the order of 10^2, while O_2 and pH have smaller orders of magnitude.

1.3 Compute median and MAD of each chemical and compare the two sets of quantities (*i.e.*, mean & variance vs. median & MAD). What do you notice?

```r
mad <- function(x, ...) {
  median_x <- median(x, ...)
  abs_diff <- abs(x - median_x)
  mad <- median(abs_diff, ...)

  return(mad)
}
```

Table 3: Mean and variance for the eight variables of interest.

variable	mean	variance
Chla	13.971197	4.200827e+02
Cl	43.636279	2.193172e+03
mnO2	9.117778	5.718089e+00
mxPH	8.011734	3.579693e-01
NH4	501.295828	3.851585e+06
NO3	3.282389	1.426176e+01
oPO4	73.590596	8.305850e+03
PO4	137.882101	1.663938e+04

Table 4: Median and median absolute deviation (MAD) for the eight variables of interest.

variable	median	MAD
Chla	5.4750	4.5000
Cl	32.7300	22.4265
mnO2	9.8000	1.3850
mxPH	8.0600	0.3400
NH4	103.1665	75.2850
NO3	2.6750	1.4650
oPO4	40.1500	29.7085
PO4	103.2855	82.5045

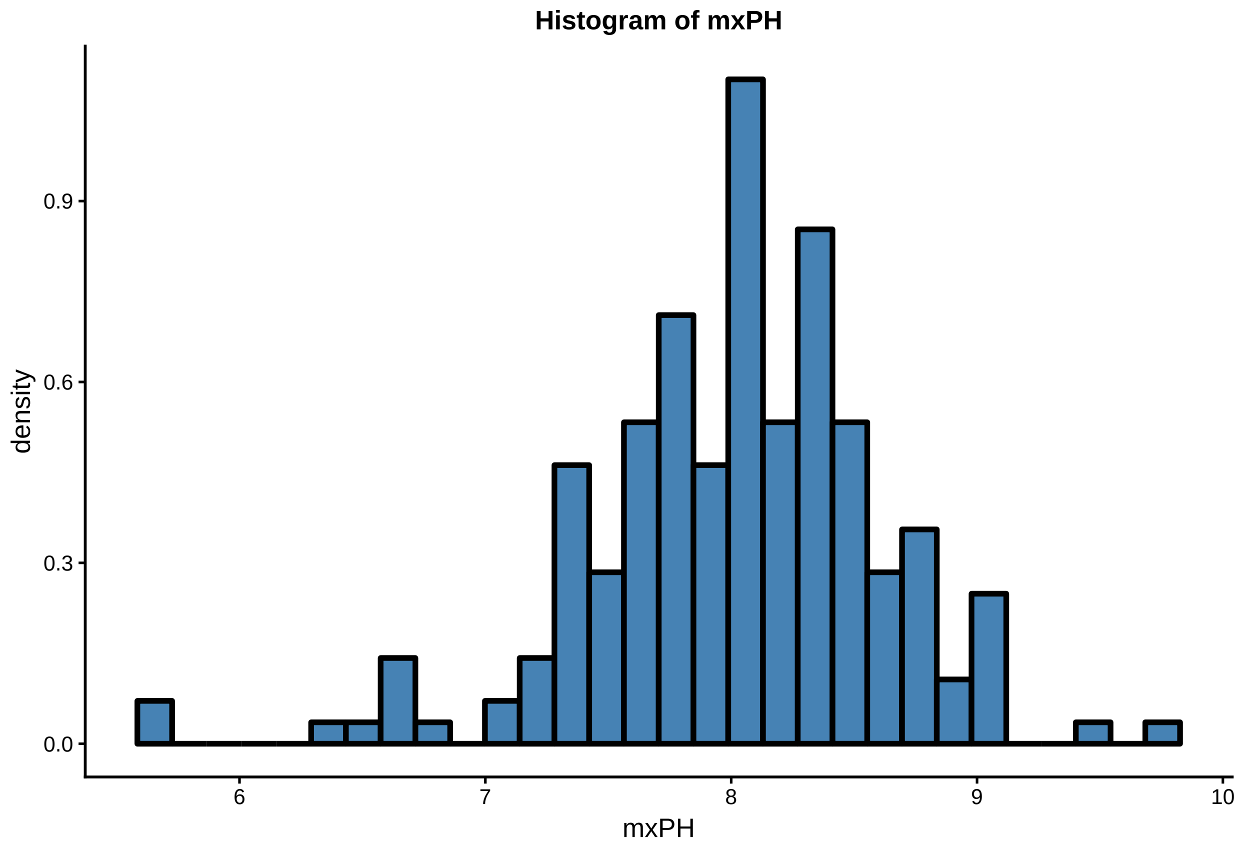
```r
algae %>%
  select(-c(1:3, matches("a."))) %>%
  gather(variable, value) %>%
  group_by(variable) %>%
  summarize(median = median(value, na.rm = T),
            MAD = mad(value, na.rm = T)) %>%
  knitr::kable(caption = "Median and median absolute deviation (MAD) for the eight variables of interes
```

The mean and median are measures of central tendency, while variance and MAD are measures of spread. The median often yields values lower than the mean, implying that our data have a long tail to the right. The MAD is orders of magnitude less, because it is not a squared measure.

2 Data visualization

2.1 Produce a histogram of mxPH with the title 'Histogram of mxPH' based on algae data set.

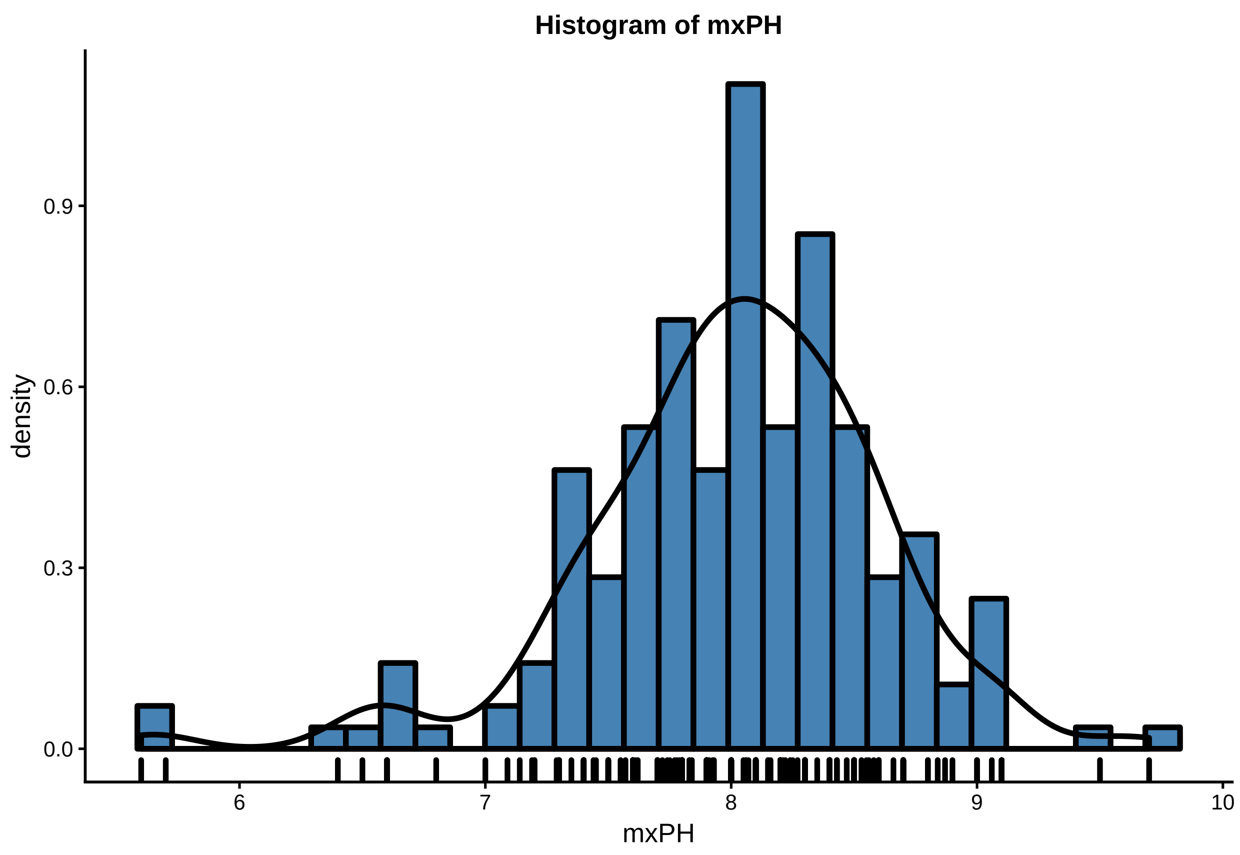
```r
ggplot(data = algae, mapping = aes(x = mxPH)) +
  geom_histogram(aes(y = stat(density))) +
  ggtitle("Histogram of mxPH")
```



The histogram above shows that pH is left-skewed.

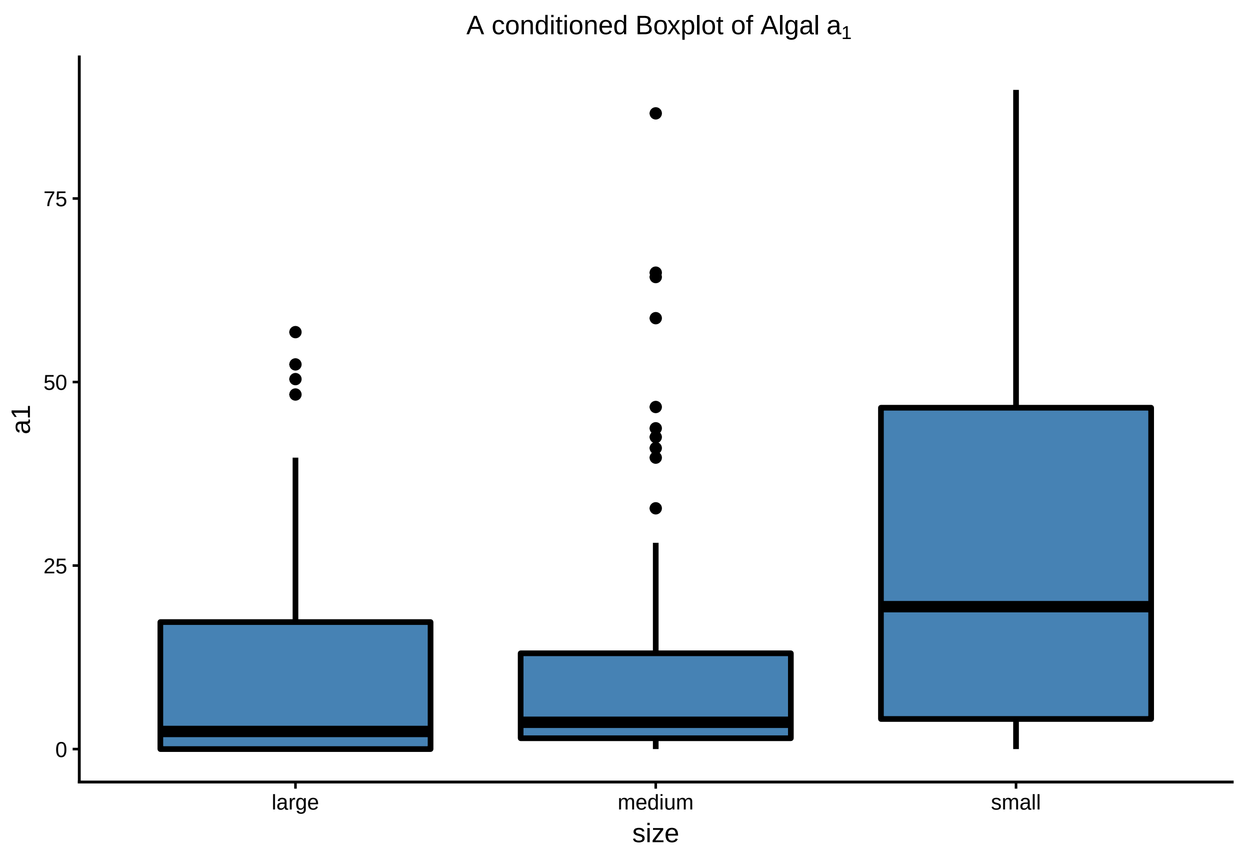
2.2 Add a density curve using `geom_density()` and rug plots using `geom_rug()` to above histogram

```r
ggplot(data = algae, mapping = aes(x = mxPH)) +
  geom_histogram(aes(y = stat(density)), fill = "steelblue", color = "black") +
  geom_density() +
  geom_rug() +
  ggtitle("Histogram of mxPH")
```



2.3 Create a boxplot with the title 'A conditioned Boxplot of Algal a1' for a1 grouped by size.

```
ggplot(data = algae, mapping = aes(x = size, y = a1)) +
  geom_boxplot(fill = "steelblue", color = "black", size = 1) +
  startR::ggtheme_plot() +
  ggtitle(quo("A conditioned Boxplot of Algal"~a[1]))
```



2.4 Are there any outliers for NO_3 and NH_4? How many observations would you consider as outliers? How did you arrive at this conclusion?

Both variables have outliers. Instead of graphically looking at this with boxplots, I calculated the interquartile range for each variable and defined lowe and upper bounds based on the lower and upper quartiles minus / plus 1.5 times the interquartile range.

```r
count_outliers <- function(x, ...) {
  # Get first and last quartiles
  qs <- quantile(x, probs = c(0.25, 0.75), ...)
  q1 <- qs[1]
  q3 <- qs[2]

  # Calculate interquartile range
  qr <- q3 - q1

  # Define lower and upper fence
  lf <- q1 - 1.5 * qr
  uf <- q3 + 1.5 * qr

  # I define outliers as poits that are not within the lower and upper fences
  sum(!between(x, lf, uf), ...)
}
```

Table 5: Number of outliers for two variables of interest.

variable	n_coutliers
NH4	27
NO3	5

Table 6: Measures of central tendency (mean, median) and spread (variance, MAD) for two variables of interest.

variable	mean	median	variance	MAD
NH4	501.295828	103.1665	3.851585e+06	75.285
NO3	3.282389	2.6750	1.426176e+01	1.465

```
algae %>%
  select(NH4, NO3) %>%
  summarize_all(count_outliers, na.rm = T) %>%
  gather(variable, n_coutliers) %>%
  knitr::kable(caption = "Number of outliers for two variables of interest.")
```

2.5 Compare mean & variance vs. median & MAD for NO_3 and NH_4. What do you notice? Can you conclude which set of measures is more robust when outliers are present?

The mean is significantly skewed for both measures, because it gives us a higher value relative to the median. The same happens with the variance. This suggests that, in the presence of outliers, median and MAD are better descriptors of the data.

```
algae %>%
  select(NH4, NO3) %>%
  gather(variable, value) %>%
  group_by(variable) %>%
  summarize(mean = mean(value, na.rm = T),
            median = median(value, na.rm = T),
            variance = var(value, na.rm = T),
            MAD = mad(value, na.rm = T)) %>%
  knitr::kable(caption = "Measures of central tendency (mean, median) and spread (variance, MAD) for tw
```

Table 7: Number of missing values for each variable in the algae dataset.

variable	n_missing
season	0
size	0
speed	0
mxPH	1
mnO2	2
Cl	10
NO3	2
NH4	2
oPO4	2
PO4	2
Chla	12
a1	0

3 Dealing with missing values

3.1 How many observations contain missing values? How many missing values are there in each variable?

```
#Number of observations with missing values
algae %>%
  select(1:12) %>%
  apply(FUN = function(x){any(is.na(x))}, MARGIN = 1) %>%
  sum()
```

```
## [1] 16
```

The above shows that 16 observations have missing values for at least one variable. The number of missing values for each variable are shown in the table below.

```
# Use a lambda-like function to count the NAs in each varialbe
algae %>%
  select(1:12) %>%
  summarize_all(function(x){sum(is.na(x))}) %>%
  gather(variable, n_missing) %>%
  knitr::kable(caption = "Number of missing values for each variable in the algae dataset.")
```

3.2 Removing observations with missing values: use filter() function in dplyr package to observations with any missing value, and save the resulting dataset (without missing values) as `algae.del`. Report how many observations are in algae.del.

```
algae.del <- algae %>%
  select(1:12) %>%
  drop_na()
```

The resulting dataset has 184 observations now.

Table 8: Values of each chemical for the 48th, 62nd and 199th observations.

mxPH	mnO2	Cl	NO3	NH4	oPO4	PO4	Chla
8.06	12.6	9.00	0.230	10.0000	5.00	6.0000	1.100
6.40	9.8	32.73	2.675	103.1665	40.15	14.0000	5.475
8.00	7.6	32.73	2.675	103.1665	40.15	103.2855	5.475

3.3 Imputing unknowns with measures of central tendency. Use mutate_at() and ifelse() in dplyr to fill in missing values for each chemical with its median, and save the imputed dataset as algae.med. Report the number of observations in `algae.med`. Display the values of each chemical for the 48th , 62th and 199th obsevation in `algae.med`.

```r
# Median imputation
algae.med <- algae %>%
  select(1:12) %>%
  mutate_at(c(4:11),
            function(x){ifelse(is.na(x), median(x, na.rm = T), x)})
```

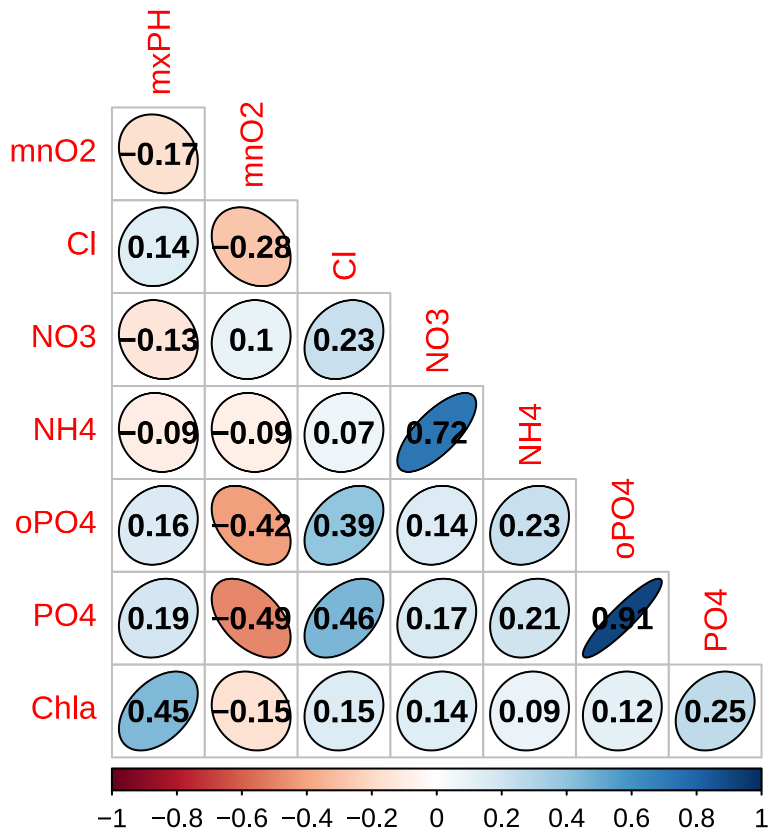
`algae.med` has 200 observations and 12 variables. The values of each chemical for the 48th, 62nd and 199th observations are shown below:

```r
algae.med %>%
  select(4:11) %>%
  filter(rownames(.) %in% c(48, 62, 199)) %>%
  knitr::kable(caption = "Values of each chemical for the 48th, 62nd and 199th observations.")
```

3.4 Imputing unknowns using correlations. Compute pairwise correlation between the continuous (chemical) variables. Then, fill in the missing value for PO4 based on oPO4 in the 28th observation. What is the value you obtain?

The following correlogram shows pearson's correlation coefficient (r) for all pairwise correlations of chemical variables in the dataset.

```r
algae %>%
  select(4:11) %>%
  cor(use = "pairwise.complete.obs") %>%
  corrplot::corrplot(type = "lower",
                     method = "ellipse",
                     addCoef.col = "black",
                     diag = F,
                     outline = T)
```



```r
PO4_lm <- lm(PO4 ~ oPO4, data = algae)

algae.med %>%
  select(PO4, oPO4) %>%
  filter(rownames(.) %in% c(28)) %>%
  mutate(PO4 = predict(PO4_lm, .))
```

```
## # A tibble: 1 x 2
##     PO4  oPO4
##   <dbl> <dbl>
## 1  48.1    4.
```

3.5 Questioning missing data assumptions. When might imputation using only the observed data lead you to incorrect conclusions?

Median imputation may be wrong when the data are not missing at random, and the value of the variable had something to do with it's detection. For example, if a given sensor is only callibrated to detect values between 1 and 10 (whatever the units may be), a true value of 40 may lead to absence of an observation. Also, for example, measurements not taken may result in missing values. For example, wave height may be omitted during a storm, because measuring wavehieght might represent a risk. Therefore, median imputation would result in a significant underestimate of the event.

4 Cross-validation

4.1 First randomly partition data into 5 equal sized chunks

```r
partition <- function(id, k = 5) {
  cut(x = id, breaks = k, label = F) %>%
    sample()
}

set.seed(42)

algae.med.part <- algae.med %>%
  mutate(partition = partition(id = 1:nrow(.), k = 5))
```

4.2 Perform 5-fold cross-validation with training error and validation errors of each chunk determined above

```r
#Define a function that returns a data.frame with training and testing errors

cv_err <- function(data, fold) {

  # Training data excludes id
  train <- data %>%
    filter(partition != fold) %>%
    select(-partition)

  # Testing data is id
  test <- data %>%
    filter(partition == fold) %>%
    select(-partition)

  model <- lm(a1 ~ ., data = train)

  train_err <- (train$a1 - predict(model)) ^ 2
  test_err <- (test$a1 - predict(model, test)) ^ 2

  tibble(train_error = mean(train_err),
         test_error = mean(test_err))
}

# Use purrr's functional programming approach to CV
kfold_errs <- tibble(fold = c(1:5)) %>%
  mutate(cv = map(fold, cv_err, data = algae.med.part)) %>%
  unnest()

knitr::kable(kfold_errs,
             caption = "Training and testing error in the 5-fold CV procedure.")
```

Table 9: Training and testing error in the 5-fold CV procedure.

fold	train_error	test_error
1	267.9506	384.2389
2	288.7568	291.0777
3	266.1087	461.7686
4	286.0354	338.5993
5	286.2211	420.7718

Table 10: True training and testing error.

train_error	test_error
286.2661	250.1794

5 BAD PROBLEM

First read in the new data

```r
algae.test <- read_table2(here("raw_data", "h1", "algaeTest.txt"),
                          col_names = c('season','size','speed','mxPH','mnO2','Cl','NO3',
                                        'NH4','oPO4','PO4','Chla','a1'),
                          na = c('XXXXXXX'),
                          col_types = cols())
```

Unexpectedly, testing the model on the new data results in much lower testing error.

```r
#Generate a model with all data
model_all_data <- lm(a1 ~ ., data = algae.med)

# Calculate testing and training errors
train_err <- (algae.med$a1 - predict(model_all_data)) ^ 2
test_err <- (algae.test$a1 - predict(model_all_data, algae.test)) ^ 2

tibble(train_error = mean(train_err),
       test_error = mean(test_err)) %>%
  knitr::kable(caption = "True training and testing error.")
```

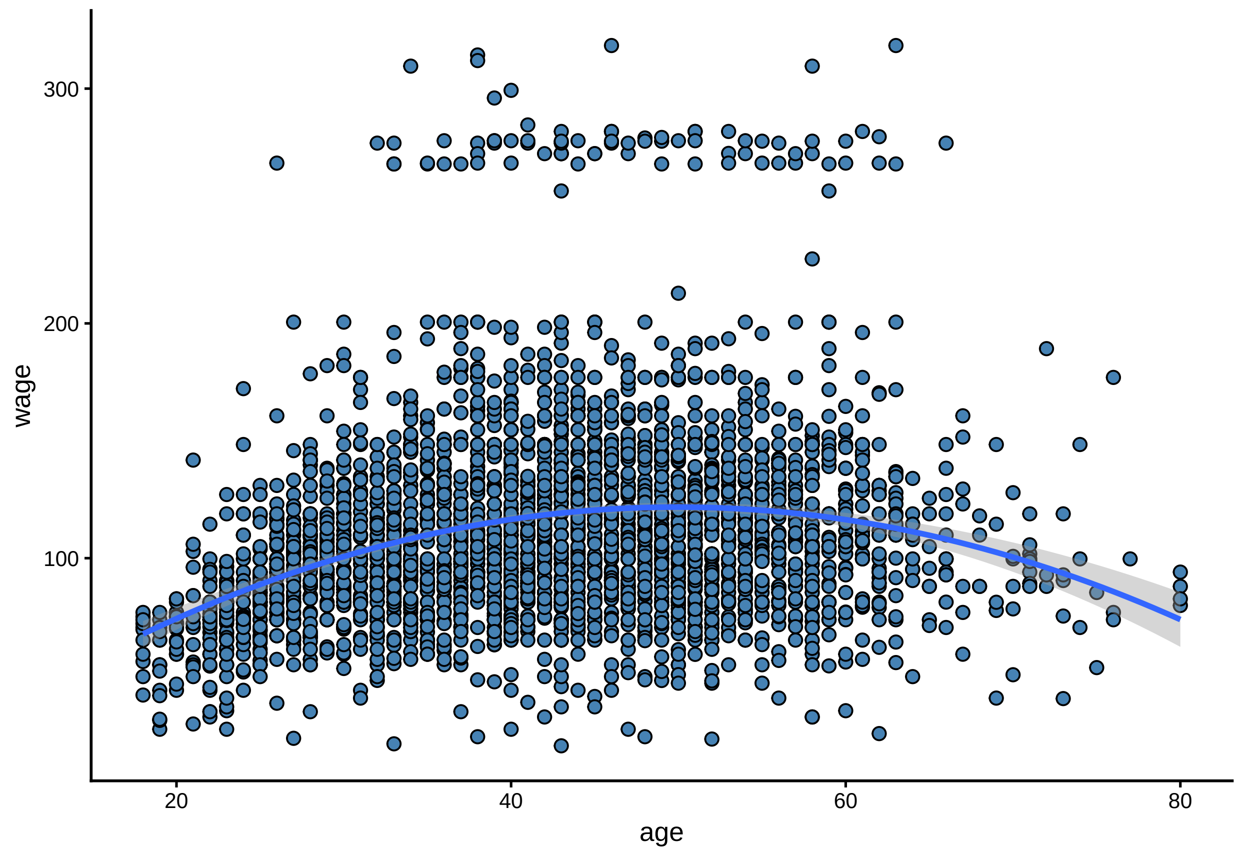
6 Cross Validation (CV) for Model Selection

```r
library(ISLR)
```

6.1 Plot wages as a function of age using `ggplot`. Your plot should include the datapoints (`geom_point()`) as well as a smooth fit to the data (`geom_smooth()`). Based on your visualization, what is the general pattern of wages as a function of age? Does this match what you expect?

Wages seem to be higher for people between 30 and 60 years old. FOr this ages, we also see some clear outliers up in the 300's, probably representing executive positions. The data follow the pattern I expected, where wages peak somewhere along the 40 year-old region, and slowly come down again.

```r
ggplot(data = Wage, mapping = aes(x = age, y = wage)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x + I(x^2))
```



6.2 In this part of the problem, we will find a polynomial function of age that best fits the wage data. For each polynomial function between p = 0, 1, 2, . . . 10

6.2.1 Fit a linear regression to predict wages as a function of age, age2 , . . . agep (you should include an intercept as well). Note that p = 0 model is an "intercept-only" model.

6.2.2 Use 5-fold cross validation to estimate the test error for this model. Save both the test error and the training error.

First, create two functions. The first function fits a specified model to a specified chunck of the data (fold), and performs theCV error estimate. The second one calls the first one after randomply (seeded) partitioning the data.

```r
cv_wages_err <- function(fold, model, data) {
  # Training data excludes id
  train <- data %>%
    filter(partition != fold) %>%
    select(-partition)

  # Testing data is id
  test <- data %>%
    filter(partition == fold) %>%
    select(-partition)

  model <- lm(as.formula(model), data = train)

  train_err <- (train$wage - predict(model)) ^ 2
  test_err <- (test$wage - predict(model, test)) ^ 2

  tibble(train_error = mean(train_err),
         test_error = mean(test_err))
}


cv_wages <- function(model, data, k = 5) {
  set.seed(42)

  data_part <- data %>%
    mutate(partition = partition(id = 1:nrow(.), k = k))

  tibble(fold = c(1:5)) %>%
    mutate(cv = map(fold, cv_wages_err, model = model, data = data_part)) %>%
    unnest()
}
```

Then, set up a tibble with the formulas I want to use. And use purrr to call the function above many times.

```r
wages_cv_results <- list(model = c("wage ~ 1",
                                   "wage ~ poly(age, 1)",
                                   "wage ~ poly(age, 2)",
                                   "wage ~ poly(age, 3)",
                                   "wage ~ poly(age, 4)",
                                   "wage ~ poly(age, 5)",
```

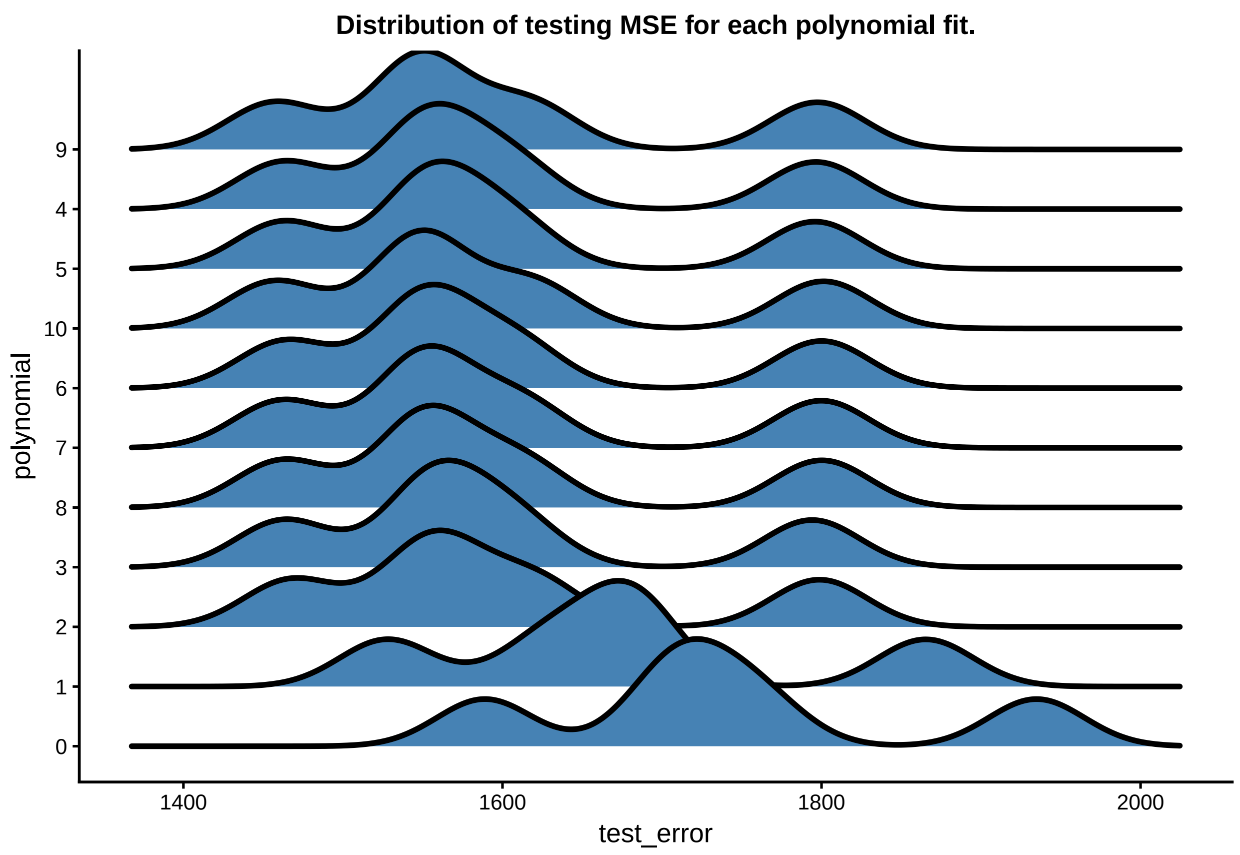
Table 11: Mean testing error for different polynomial fits (ordered by smaller - larger).

polynomial	mean_test_err
9	1593.554
4	1594.141
5	1594.633
10	1594.651
6	1594.758
7	1594.941
8	1595.428
3	1596.172
2	1600.051
1	1676.036
0	1741.658

```r
                                 "wage ~ poly(age, 6)",
                                 "wage ~ poly(age, 7)",
                                 "wage ~ poly(age, 8)",
                                 "wage ~ poly(age, 9)",
                                 "wage ~ poly(age, 10)")) %>%
  as_tibble() %>%
  mutate(cv = map(model, cv_wages, data = Wage, k = 5)) %>%
  unnest(.id = "polynomial") %>%
  mutate(polynomial = as.numeric(polynomial) - 1)
```

```r
wages_cv_results %>%
  group_by(polynomial) %>%
  summarize(mean_test_err = mean(test_error)) %>%
  arrange(mean_test_err) %>%
  knitr::kable(caption = "Mean testing error for different polynomial fits (ordered by smaller - larger
```

```r
wages_cv_results %>%
  mutate(polynomial = as.factor(polynomial),
         polynomial = fct_reorder(polynomial, test_error, mean, .desc = T)) %>%
  ggplot(mapping = aes(x = test_error, y = polynomial)) +
  ggridges::geom_density_ridges(fill = "steelblue", color = "black", size = 1) +
  ggtitle("Distribution of testing MSE for each polynomial fit.")
```


Distribution of testing MSE for each polynomial fit.
polynomial
9
4
5
10
6
7
8
3
2
1
0
1400
1600
1800
2000
test_error

7 (231 Only) The bias-variance tradeof. Prove that the mean squared error can be decomposed into the variance plus bias squared.

The mean squared error for an estimator can be given by:

$$MSE = \mathbb{E}[(\hat{\theta} - \theta)^2]$$

We can expand the squared binomial and take advantage of the fact that θ is not a random variable, and that it is equal to it's expectation independent of any distribution that describes it. The above then becomes:

$$\mathbb{E}[\hat{\theta}^2] + \theta^2 - 2\mathbb{E}[\hat{\theta}]\theta$$

The Bias squared is given by:

$$\begin{aligned} Bias^2(\hat{\theta}, \theta) &= (\mathbb{E}[\hat{\theta}] - \theta)^2 \\ &= \mathbb{E}^2[\hat{\theta}] + \theta^2 - 2\mathbb{E}[\hat{\theta}]\theta \end{aligned}$$

The variance is given by:

$$Var(\hat{\theta}) = \mathbb{E}[\hat{\theta}^2] - \mathbb{E}^2[\theta]$$

We can put the squared bias and variance together to form the following:

$$Bias^2(\hat{\theta}, \theta) + Var(\hat{\theta}) = \mathbb{E}^2[\hat{\theta}] + \theta^2 - 2\mathbb{E}[\hat{\theta}]\theta + \mathbb{E}[\hat{\theta}^2] - \mathbb{E}^2[\theta]$$

The first term of the bias and the last term of the variance cancel out ($\mathbb{E}^2[\theta]$), leaving us with:

$$Bias^2(\hat{\theta}, \theta) + Var(\hat{\theta}) = \theta^2 - 2\mathbb{E}[\hat{\theta}]\theta + \mathbb{E}[\hat{\theta}^2]$$

This last expression is the same as the initial expression for MSE, showing how MSE can be decomposed into bias squared and variance.

8 (231 Only) Distances

The two proposed measures are:

$$d(x, y) = ||x - y||_2$$

and

$$d(x, y) = ||x - y||_\infty$$

The first is the Euclidean distance, and it is computed via the pythagorean formula as:

$$d(x, y) = d(y, x) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2}$$

Which generalizes to:

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

We know that square numbers are non-negative, so this measure satisfies the *positivity* property. Since $(x_i - y_i)^2 = (y_i - x_i)^2$, the *symmetry* property is also satisfied.