STATS701 Homework 1

Jordan Farrer

2016-09-25

Contents

1 Setup

Full repo: https://github.com/jrfarrer/stats701_hw1/

Published file: https://jrfarrer.github.io/stats701_hw1/

Begin by setting up the R session, creating a logger function, and loading packages.

```r
# Set the seed for reproducibility
set.seed(44)
# Set the locale of the session so languages other than English can be used
invisible(Sys.setlocale("LC_ALL", "en_US.UTF-8"))
# Prevent printing in scientific notation
options(scipen = 999, digits = 4)

# Create a logger function
logger <- function(msg, level = "info", file = log_file) {
    cat(paste0("[", format(Sys.time(), "%Y-%m-%d %H:%M:%S.%OS"), "][", level, "] ", msg, "\n"), file = s
}

# Set the project directory
base_dir <- ''
data_dir <- paste0(base_dir, "data/")
code_dir <- paste0(base_dir, "code/")
viz_dir <- paste0(base_dir, "viz/")

dir.create(data_dir, showWarnings = FALSE)
dir.create(code_dir, showWarnings = FALSE)
dir.create(viz_dir, showWarnings = FALSE)
```

```r
# Create a function that will be used to load/install packages
fn_load_packages <- function(p) {
  if (!is.element(p, installed.packages()[,1]) || (p =="DT" && !(packageVersion(p) > "0.1"))) {
    if (p == "DT") {
      devtools::install_github('rstudio/DT')
    } else {
      install.packages(p, dep = TRUE, repos = 'http://cran.us.r-project.org')
    }
  }
  a <- suppressPackageStartupMessages(require(p, character.only = TRUE))
  if (a) {
    logger(paste0("Loaded package ", p, " version ", packageVersion(p)))
  } else {
    logger(paste0("Unable to load packages ", p))
  }
}
# Create a vector of packages
packages <- c('dplyr','tidyr','readr','stringr','ggplot2','ggthemes','knitr','readxl',
              'broom','forecast','ISLR','GGally','gridExtra','leaps','extrafont')
# Use function to load the required packages
invisible(lapply(packages, fn_load_packages))
```

```
## [2016-09-24 15:24:17.17][info] Loaded package dplyr version 0.5.0
## [2016-09-24 15:24:17.17][info] Loaded package tidyr version 0.6.0.9000
## [2016-09-24 15:24:17.17][info] Loaded package readr version 1.0.0
## [2016-09-24 15:24:17.17][info] Loaded package stringr version 1.1.0
## [2016-09-24 15:24:17.17][info] Loaded package ggplot2 version 2.1.0
## [2016-09-24 15:24:17.17][info] Loaded package ggthemes version 3.2.0
## [2016-09-24 15:24:17.17][info] Loaded package knitr version 1.13
## [2016-09-24 15:24:17.17][info] Loaded package readxl version 0.1.1
## [2016-09-24 15:24:18.18][info] Loaded package broom version 0.4.1
```

```
## [2016-09-24 15:24:18.18][info] Loaded package forecast version 7.1
## [2016-09-24 15:24:18.18][info] Loaded package ISLR version 1.0
## [2016-09-24 15:24:18.18][info] Loaded package GGally version 1.2.0
## [2016-09-24 15:24:18.18][info] Loaded package gridExtra version 2.2.1
## [2016-09-24 15:24:18.18][info] Loaded package leaps version 2.9
## [2016-09-24 15:24:18.18][info] Loaded package extrafont version 0.17
```

```
# Installs fonts
#system(paste0("cp -r ",viz_dir,"fonts/. ~/Library/Fonts/"))
```

```
# Create a color palette
pal538 <- ggthemes_data$fivethirtyeight

# Create a theme to use throughout the analysis
theme_jrf <- function(base_size = 8, base_family = "DecimaMonoPro") {
    theme(
        plot.background = element_rect(fill = "#F0F0F0", colour = "#606063"),
        panel.background = element_rect(fill = "#F0F0F0", colour = NA),
        panel.border = element_blank(),
        panel.grid.major =   element_line(colour = "#D7D7D8"),
        panel.grid.minor =   element_line(colour = "#D7D7D8", size = 0.25),
        panel.margin =       unit(0.25, "lines"),
        panel.margin.x =     NULL,
        panel.margin.y =     NULL,
        axis.ticks.x = element_blank(),
        axis.ticks.y = element_blank(),
        axis.title = element_text(colour = "#A0A0A3"),
        axis.text.x = element_text(vjust = 1, family = 'Helvetica', colour = '#3C3C3C'),
        axis.text.y = element_text(hjust = 1, family = 'Helvetica', colour = '#3C3C3C'),
        legend.background = element_blank(),
        legend.key = element_blank(),
        plot.title = element_text(face = 'bold', colour = '#3C3C3C', hjust = 0),
        text = element_text(size = 9, family = "DecimaMonoPro"),
        title = element_text(family = "DecimaMonoPro-Bold")

    )
}
```

2 Question 2

2.1 Data Loading

Let's use Hadley's readr package to load the dataset, using the col_name parameter to set the column names of the tibble.

```
# Load the csv with meaningful column names
survey_results <- read_csv(paste0(data_dir,'Survey_results_final.csv'), skip = 1,
                           col_names = c('hitid','hittypeid','title','description','keywords',
                            'reward','creationtime','maxassignments','requesterannotation',
                            'assignmentdurationinseconds','autoapprovaldelayinseconds',
                            'expiration','numberofsimilarhits','lifetimeinseconds',
                            'assignmentid','workerid','assignmentstatus','accepttime',
```

```r
                             'submittime','autoapprovaltime','approvaltime','rejectiontime',
                             'requesterfeedback','worktime','lifetimeapprovalrate',
                             'last30daysapprovalrate','last7daysapprovalrate','age',
                             'education','gender','income','sirius','wharton','approve','reject'))

# Print a few records in the tibble
survey_results %>%
    select(age, education, gender, income, sirius, wharton, worktime)
```

```
## # A tibble: 1,764 × 7
##      age                                    education gender
##    <chr>                                        <chr>  <chr>
## 1     21 Some college, no diploma; or Associate's degree Female
## 2     56 Some college, no diploma; or Associate's degree Female
## 3     40                Graduate or professional degree Female
## 4     52                Graduate or professional degree Female
## 5     33        Bachelor's degree or other 4-year degree   Male
## 6     55 Some college, no diploma; or Associate's degree   Male
## 7     24 Some college, no diploma; or Associate's degree Female
## 8     40        Bachelor's degree or other 4-year degree Female
## 9     35        Bachelor's degree or other 4-year degree Female
## 10    62 Some college, no diploma; or Associate's degree Female
## # ... with 1,754 more rows, and 4 more variables: income <chr>,
## #   sirius <chr>, wharton <chr>, worktime <int>
```

```r
# Put into a new tibble we'll use for cleaning (there will be a final later)
survey_results_cleaning <- survey_results
```

2.2 Data Cleaning

We'll sequentially clean each of the primary variables of the dataset and create exploratory summaries.

2.2.1 Age

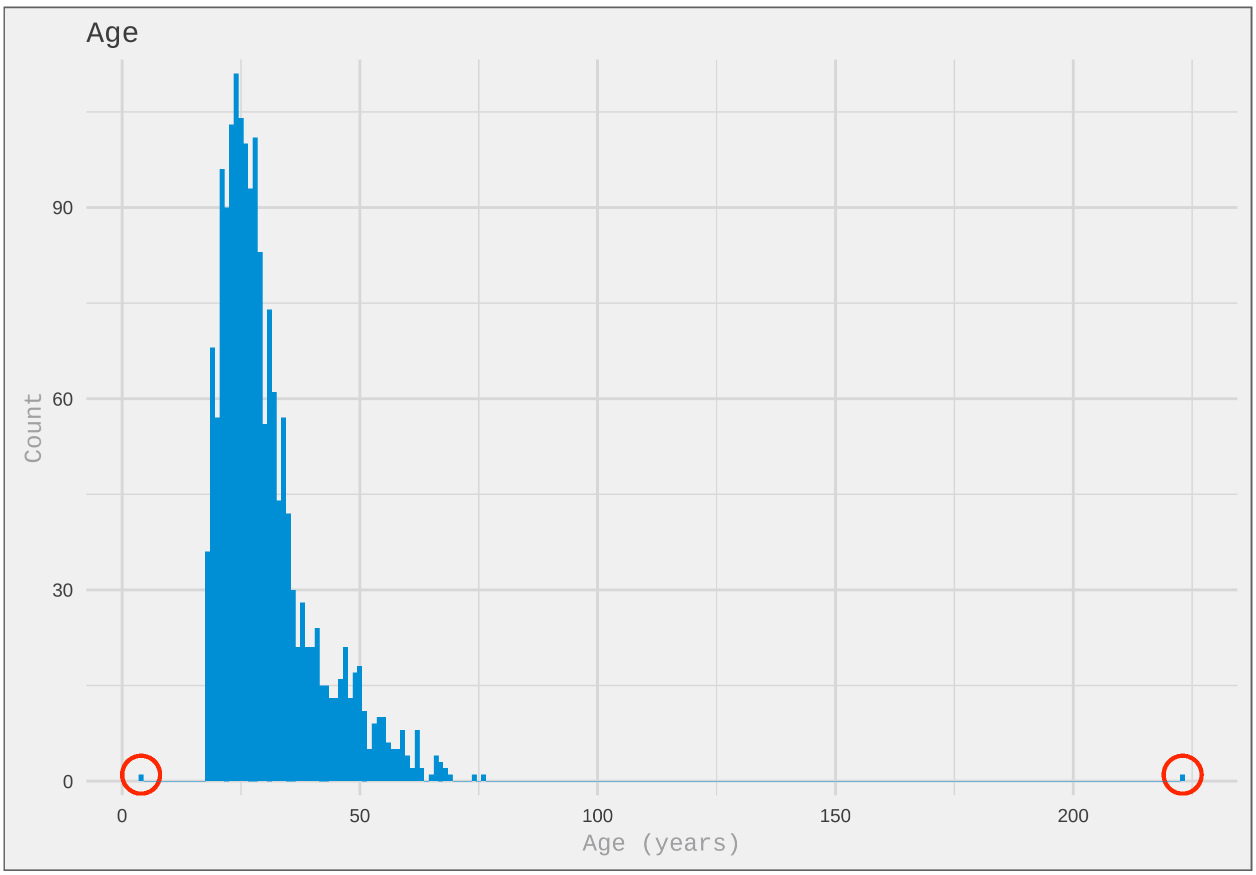
Let's quickly summarize the age variable, noting that it is a character.

```r
survey_results_cleaning %>% group_by(age) %>% summarise(cnt = n()) %>% arrange(age)
```

```
## # A tibble: 59 × 2
##      age   cnt
##    <chr> <int>
## 1     18    35
## 2     19    68
## 3     20    57
## 4     21    96
## 5     22    90
## 6    223     1
## 7     23   103
## 8     24   111
## 9     25   104
## 10    26   100
## # ... with 49 more rows
```

We correct some errant values, using our judgement as **data analysts** and plot a histogram.

```
survey_results_cleaning <-
    survey_results %>%
    mutate(
        age2 = ifelse(age == 'Eighteen (18)', "18", ifelse(age == 'female', NA, ifelse(age == "27`", "2
        , age2 = as.integer(age2)
    )

ggplot(survey_results_cleaning, aes(x = age2)) +
    geom_point(aes(x = 4, y = 1), shape = 1, colour = pal538['red'], fill = NA, size = 6, stroke = 1) +
    geom_point(aes(x = 223, y = 1), shape = 1, colour = pal538['red'], fill = NA, size = 6, stroke = 1)
    geom_histogram(binwidth = 1, fill = pal538['blue']) +
    theme_jrf() +
    scale_x_continuous(expand = c(0.05, 0.01)) + scale_y_continuous(expand = c(0.02, 0.01)) +
    labs(title = "Age", y = "Count", x = "Age (years)")
```



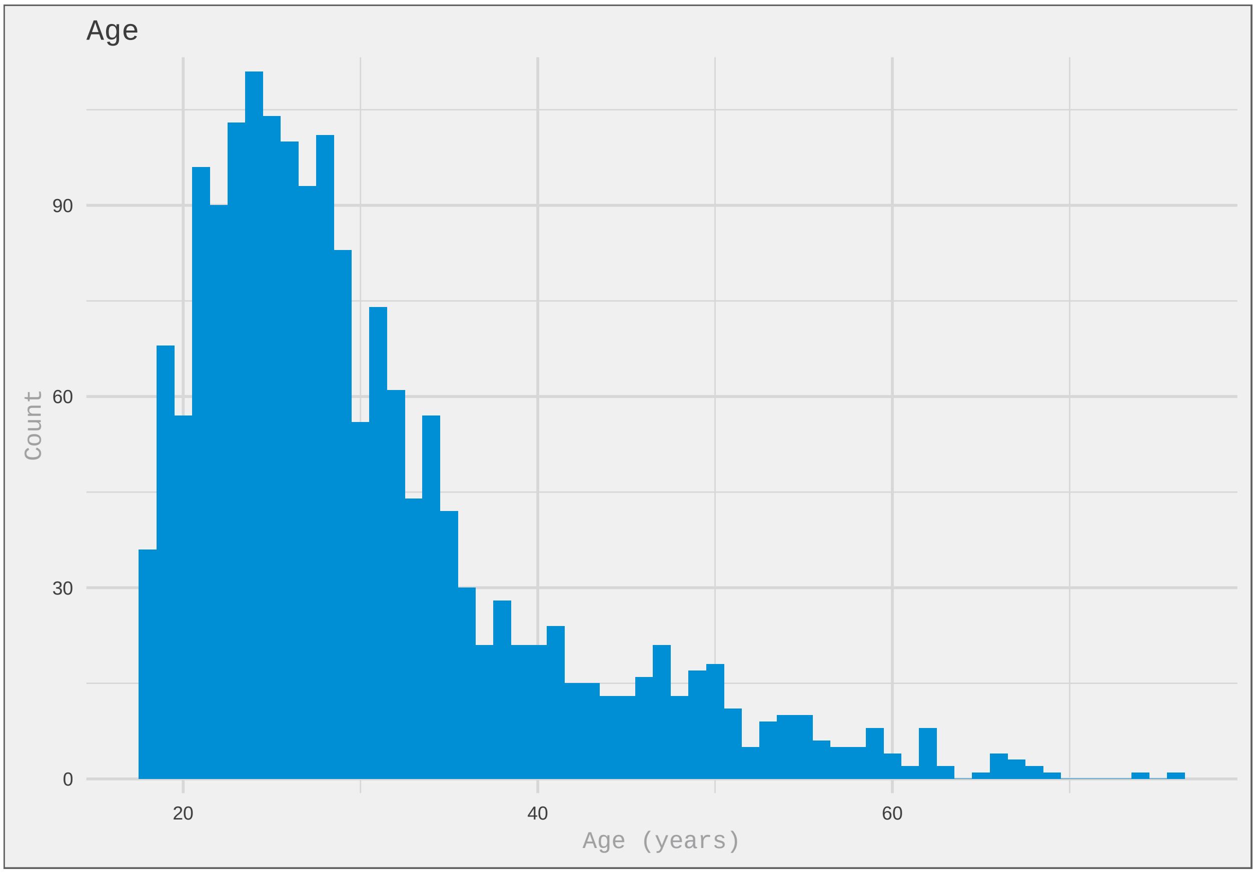
It looks like we still missed some bad values.

```
sort(unique(survey_results_cleaning$age2))
```

```
##  [1]   4  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33
## [18]  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50
## [35]  51  52  53  54  55  56  57  58  59  60  61  62  63  65  66  67  68
## [52]  69  74  76 223
```

We fix those too and plot the histogram.

```
survey_results_cleaning <-
    survey_results_cleaning %>%
    mutate(
        age3 = ifelse(age2 %in% c(4, 223), NA, age2)
    )

ggplot(survey_results_cleaning, aes(x = age3)) + geom_histogram(binwidth = 1, fill = pal538['blue']) +
    theme_jrf() +
    scale_x_continuous(expand = c(0.05, 0.01)) + scale_y_continuous(expand = c(0.02, 0.01)) +
    labs(title = "Age", y = "Count", x = "Age (years)")
```



2.2.2 Education

Let's look at the unique values and counts.

```
survey_results_cleaning %>% group_by(education) %>% summarise(cnt = n()) %>% arrange(education)
```

```
## # A tibble: 7 × 2
##                                        education   cnt
##                                            <chr> <int>
## 1        Bachelor's degree or other 4-year degree   614
## 2                 Graduate or professional degree   181
```

```
## 3            High school graduate (or equivalent)   193
## 4      Less than 12 years; no high school diploma    10
## 5                                            Other     2
## 6                                       select one    19
## 7 Some college, no diploma; or Associate's degree   745
```

It appears that 19 respondents left the survey on the default which read 'select one'. We'll update that to 'Other' and modify this variable to be a factor.

```
survey_results_cleaning <-
    survey_results_cleaning %>%
    mutate(
        education2 = ifelse(education == "select one", "Other", education)
        , education2 = factor(education2, levels = c('Less than 12 years; no high school diploma'
                                                       , 'High school graduate (or equivalent)'
                                                       , 'Some college, no diploma; or Associate's deg
                                                       , 'Bachelor's degree or other 4-year degree'
                                                       , 'Graduate or professional degree'
                                                       , 'Other'))
    )

survey_results_cleaning %>% group_by(education2) %>% summarise(cnt = n()) %>% arrange(education2)
```

```
## # A tibble: 6 × 2
##                                       education2   cnt
##                                           <fctr> <int>
## 1      Less than 12 years; no high school diploma    10
## 2            High school graduate (or equivalent)   193
## 3 Some college, no diploma; or Associate's degree   745
## 4        Bachelor's degree or other 4-year degree   614
## 5                 Graduate or professional degree   181
## 6                                            Other    21
```

2.2.3 Gender

We'll summarise the gender variable.

```
survey_results_cleaning %>% group_by(gender) %>% summarise(cnt = n()) %>% arrange(gender)
```

```
## # A tibble: 3 × 2
##   gender   cnt
##    <chr> <int>
## 1 Female   745
## 2   Male  1013
## 3   <NA>     6
```

We update this to be a factor.

```
survey_results_cleaning <-
    survey_results_cleaning %>%
    mutate(gender2 = as.factor(gender))
```

```
survey_results_cleaning %>%
    group_by(gender2) %>%
    summarise(cnt = n()) %>%
    arrange(gender2) %>%
    mutate(prop = cnt / sum(cnt))
```

```
## # A tibble: 3 × 3
##   gender2   cnt     prop
##    <fctr> <int>    <dbl>
## 1  Female   745 0.422336
## 2    Male  1013 0.574263
## 3      NA     6 0.003401
```

2.2.4 Income

```
survey_results_cleaning %>% group_by(income) %>% summarise(cnt = n()) %>% arrange(income)
```

```
## # A tibble: 7 × 2
##               income   cnt
##                <chr> <int>
## 1  $15,000 - $30,000   367
## 2  $30,000 - $50,000   429
## 3  $50,000 - $75,000   377
## 4 $75,000 - $150,000   329
## 5     Above $150,000    47
## 6  Less than $15,000   209
## 7               <NA>     6
```

Let's convert this to a factor variable.

```
survey_results_cleaning <-
    survey_results_cleaning %>%
    mutate(
        income2 = factor(income, levels = c('Less than $15,000'
                                            , '$15,000 - $30,000'
                                            , '$30,000 - $50,000'
                                            , '$50,000 - $75,000'
                                            , '$75,000 - $150,000'
                                            , 'Above $150,000'))
    )

survey_results_cleaning %>% group_by(income2) %>% summarise(cnt = n()) %>% arrange(income2)
```

```
## # A tibble: 7 × 2
##              income2   cnt
##               <fctr> <int>
## 1  Less than $15,000   209
## 2  $15,000 - $30,000   367
## 3  $30,000 - $50,000   429
```

```
## 4  $50,000 - $75,000   377
## 5 $75,000 - $150,000   329
## 6     Above $150,000    47
## 7                 NA     6
```

2.2.5 Sirius and Wharton

```
survey_results_cleaning %>% group_by(sirius) %>% summarise(cnt = n()) %>% arrange(sirius)
```

```
## # A tibble: 3 × 2
##   sirius   cnt
##    <chr> <int>
## 1     No   399
## 2    Yes  1360
## 3   <NA>     5
```

```
survey_results_cleaning %>% group_by(wharton) %>% summarise(cnt = n()) %>% arrange(wharton)
```

```
## # A tibble: 3 × 2
##   wharton   cnt
##     <chr> <int>
## 1      No  1690
## 2     Yes    70
## 3    <NA>     4
```

Let's convert these to factors for better analysis capabilities.

```
survey_results_cleaning <-
    survey_results_cleaning %>%
    mutate(
        sirius2 = factor(sirius, levels = c("Yes","No"))
        , wharton2 = factor(wharton, levels = c("Yes","No"))
    )

survey_results_cleaning %>% group_by(sirius2) %>% summarise(cnt = n()) %>% arrange(sirius2)
```

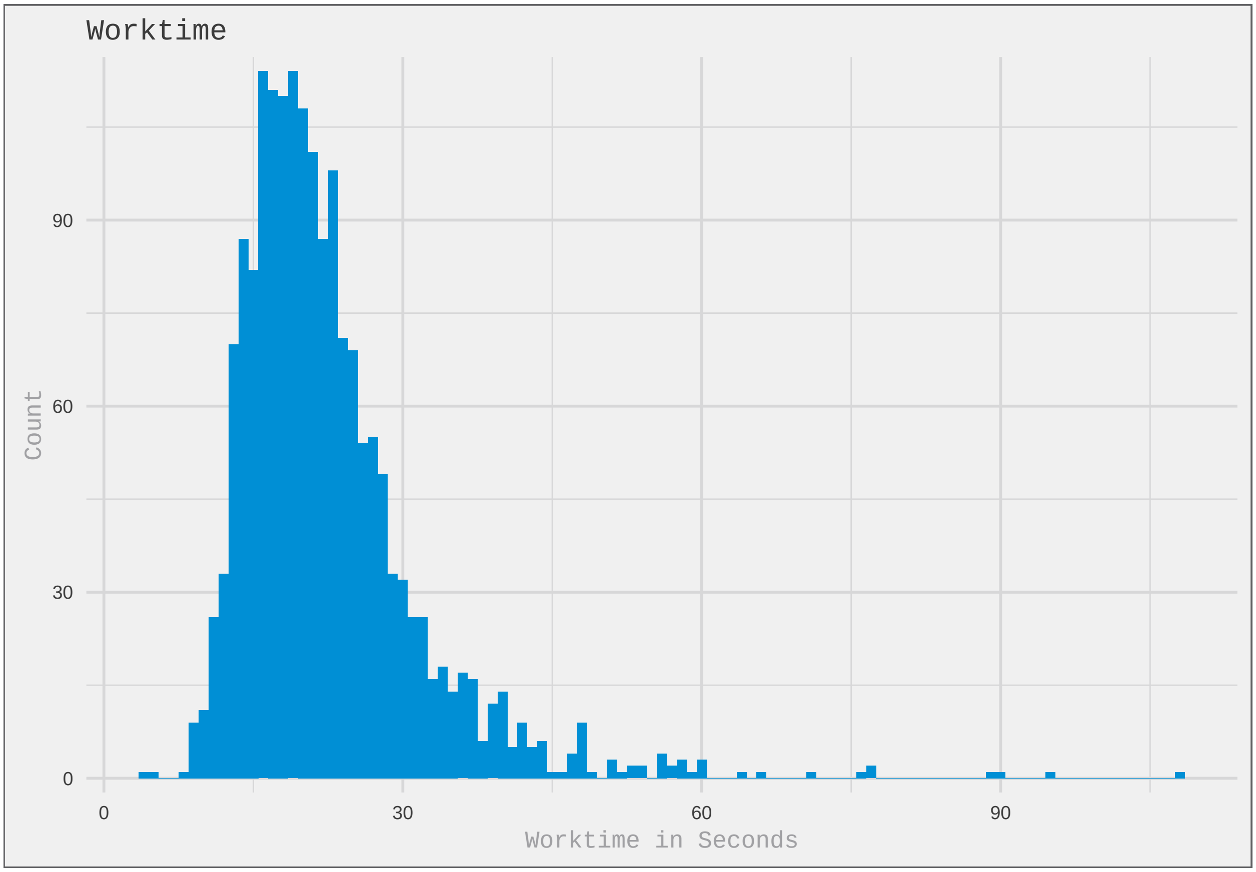
```
## # A tibble: 3 × 2
##   sirius2   cnt
##    <fctr> <int>
## 1     Yes  1360
## 2      No   399
## 3      NA     5
```

```
survey_results_cleaning %>% group_by(wharton2) %>% summarise(cnt = n()) %>% arrange(wharton2)
```

```
## # A tibble: 3 × 2
##   wharton2   cnt
##     <fctr> <int>
## 1      Yes    70
## 2       No  1690
## 3       NA     4
```

2.2.6 Worktime

```
ggplot(survey_results_cleaning, aes(x = worktime)) + geom_histogram(binwidth = 1, fill = pal538['blue'])
    theme_jrf() +
    scale_x_continuous(expand = c(0.05, 0.01)) + scale_y_continuous(expand = c(0.02, 0.01)) +
    labs(title = "Worktime", y = "Count", x = "Worktime in Seconds")
```



2.2.7 Final Data Frame

We select and rename the columns.

```
survey_results_cleaning <-
    survey_results_cleaning %>%
    select(age3, education2, gender2, income2, sirius2, wharton2, worktime) %>%
    rename(
        age = age3
        , education = education2
        , gender = gender2
        , income = income2
        , sirius = sirius2
        , wharton = wharton2
    )
```

Let's review the records with missing data.

```r
survey_results_cleaning[!complete.cases(survey_results_cleaning), ] %>% print(width = Inf)
```

```
## # A tibble: 20 × 7
##      age                                      education gender
##    <int>                                         <fctr> <fctr>
## 1     53 Some college, no diploma; or Associate’s degree   Male
## 2     19 Some college, no diploma; or Associate’s degree   Male
## 3     NA                                          Other   Male
## 4     29        Bachelor’s degree or other 4-year degree Female
## 5     32 Some college, no diploma; or Associate’s degree   Male
## 6     36                  Graduate or professional degree Female
## 7     47                  Graduate or professional degree     NA
## 8     NA            High school graduate (or equivalent)   Male
## 9     21                                          Other   Male
## 10    49            High school graduate (or equivalent)   Male
## 11    25            High school graduate (or equivalent) Female
## 12    NA Some college, no diploma; or Associate’s degree Female
## 13    35 Some college, no diploma; or Associate’s degree Female
## 14    47                  Graduate or professional degree     NA
## 15    29 Some college, no diploma; or Associate’s degree     NA
## 16    31                  Graduate or professional degree     NA
## 17    NA        Bachelor’s degree or other 4-year degree   Male
## 18    25 Some college, no diploma; or Associate’s degree     NA
## 19    34        Bachelor’s degree or other 4-year degree Female
## 20    67 Some college, no diploma; or Associate’s degree     NA
##                 income sirius wharton worktime
##                 <fctr> <fctr>  <fctr>    <int>
## 1                   NA    Yes      No       28
## 2  $75,000 - $150,000     NA      No       25
## 3                   NA     NA      NA        5
## 4                   NA    Yes      No       22
## 5   $15,000 - $30,000     NA      No       37
## 6  $75,000 - $150,000     NA      No       20
## 7   $30,000 - $50,000    Yes      No       54
## 8   $30,000 - $50,000     No      No       11
## 9                   NA     NA      NA        4
## 10                  NA     No      No       14
## 11  $30,000 - $50,000    Yes      NA       15
## 12     Above $150,000    Yes      No       21
## 13                  NA    Yes      No       18
## 14  $50,000 - $75,000    Yes      No       15
## 15  $15,000 - $30,000    Yes      No       19
## 16  $30,000 - $50,000     No      No       15
## 17  $50,000 - $75,000    Yes      No       22
## 18  Less than $15,000    Yes      No       19
## 19  $50,000 - $75,000    Yes      NA       16
## 20  $50,000 - $75,000     No      No       32
```

We will remove the 7 records that have NAs for sirius or wharton. Without information about the response, we will have trouble making an estimate of p, the porportion of Sirius listeners who listened to Business Radio Powered by the Wharton School.

```r
survey_results_cleaning %>%
    filter(is.na(sirius) | is.na(wharton))
```

```
## # A tibble: 7 × 7
##     age                                       education gender
##   <int>                                          <fctr> <fctr>
## 1    19 Some college, no diploma; or Associate’s degree   Male
## 2    NA                                           Other   Male
## 3    32 Some college, no diploma; or Associate’s degree   Male
## 4    36                  Graduate or professional degree Female
## 5    21                                           Other   Male
## 6    25             High school graduate (or equivalent) Female
## 7    34         Bachelor’s degree or other 4-year degree Female
## # ... with 4 more variables: income <fctr>, sirius <fctr>, wharton <fctr>,
## #   worktime <int>
```

We put together a final data frame.

```r
survey_results_final <-
    survey_results_cleaning %>%
    filter(!is.na(sirius) & !is.na(wharton))
```

2.3 Summary

We previously listened to the podcast Planet Money’s episode about Amazon’s Mechanical Turk program.

```r
sapply(survey_results_final, summary)
```

```
## $age
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    18.0    23.0    28.0    30.4    34.0    76.0       3
##
## $education
##      Less than 12 years; no high school diploma
##                                              10
##            High school graduate (or equivalent)
##                                             192
## Some college, no diploma; or Associate’s degree
##                                             743
##        Bachelor’s degree or other 4-year degree
##                                             613
##                 Graduate or professional degree
##                                             180
##                                           Other
##                                              19
##
## $gender
## Female   Male   NA's
##    742   1009      6
##
## $income
```

```
##  Less than $15,000  $15,000 - $30,000  $30,000 - $50,000
##                209                366                428
##  $50,000 - $75,000 $75,000 - $150,000     Above $150,000
##                376                327                 47
##               NA's
##                  4
##
## $sirius
##  Yes   No
## 1358  399
##
## $wharton
##  Yes   No
##   70 1687
##
## $worktime
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     8.0    17.0    21.0    22.5    26.0   108.0
```

Variable	Class	Description
Age	Integer	The age in years of the survey respondent
Education	Factor	Level of education attain by the survey respondent
Gender	Factor	Gender indicated by the survey respondent (Male or Female)

Variable	Class	Description
Income	Factor	Income level provided by the survey respondent
Sirius	Factor	Response to “Have you ever listened to Sirius Radio?”
Wharton	Factor	Response to “Have you ever listened to Sirius Business Radio by Wharton?”
Worktime	Integer	Number of second spent completing the survey

2.4 Sample properties

2.4.1 (1)

On the surface, we have no reason to believe that the MTURK dataset could be representative of the US population. Knowledge of MTURK is not universial and attracts particular types of individuals willing to perform many small tasks for a minor reward (from Planet Money podcast).

First, we quickly see that the porportion of Sirius listeners is much higher than the given proportion. If the US population is 321.4 million, then the proportion of Sirius listeners is

$$\frac{51.6}{321.4} = 0.1605$$

However, we quickly see that in the survey data from MTURK, the proportion of Sirius listeners is much higher.

```r
(sirius_prop <- survey_results_final %>% summarise(prop_sirius = sum(sirius == "Yes") / n()))
```

```
## # A tibble: 1 × 1
##   prop_sirius
##         <dbl>
## 1      0.7729
```

We see that of the survey respondents, 77.29% say that have listened to Sirius.

Second, in order to answer the question "Does this appear to be a random sample from the US population?" empirically we can look at the four characteristics in our final dataset

1. Age
2. Gender
3. Education
4. Income

For age and gender, we download a table called "Population by Age" from US Census Bureau's Current Population Survey in 2012.

```r
census_age_gender <- read_csv(url("http://www.census.gov/population/age/data/files/2012/2012gender_tabl
                              skip = 6,
                              col_names = c("age", "all","all_percent","male","male_percent","female","
                              col_types = cols(age = col_character(),
                                               all = col_number(),
                                               all_percent = col_number(),
                                               male = col_number(),
                                               male_percent = col_number(),
                                               female = col_number(),
                                               female_percent = col_number())
                              )

census_age_gender
```

```
## # A tibble: 34 × 7
##                age    all all_percent   male male_percent female
##              <chr>  <dbl>       <dbl>  <dbl>        <dbl>  <dbl>
## 1         All ages 308827       100.0 151175        100.0 157653
## 2   .Under 5 years  20110         6.5  10273          6.8   9837
## 3    .5 to 9 years  20416         6.6  10427          6.9   9989
## 4  .10 to 14 years  20605         6.7  10529          7.0  10076
## 5  .15 to 19 years  21239         6.9  10840          7.2  10399
## 6  .20 to 24 years  21878         7.1  10987          7.3  10891
## 7  .25 to 29 years  20893         6.8  10430          6.9  10464
## 8  .30 to 34 years  20326         6.6  10034          6.6  10292
## 9  .35 to 39 years  19140         6.2   9421          6.2   9719
## 10 .40 to 44 years  20787         6.7  10255          6.8  10532
## # ... with 24 more rows, and 1 more variables: female_percent <dbl>
```

We will need to bucket our MTURK dataset to match the categories of the Census Bureau's. In doing so we remove the 109 records with an age 18-19 and without a listed gender.

```
actual <-
    survey_results_final %>%
        filter(age >= 20 & !is.na(gender)) %>%
        mutate(age_bucket = paste0(floor(age / 10), "0 to ", floor(age / 10),"9 years")) %>%
        group_by(age_bucket, gender) %>%
        summarise(
            n = n()
        ) %>%
        ungroup() %>%
        mutate(source = "Actual") %>%
        select(source, age_bucket, gender, n)

actual_size <- sum(actual$n)

actual
```

```
## # A tibble: 12 × 4
##    source     age_bucket gender     n
##     <chr>          <chr> <fctr> <int>
## 1  Actual 20 to 29 years Female   375
## 2  Actual 20 to 29 years   Male   559
## 3  Actual 30 to 39 years Female   182
## 4  Actual 30 to 39 years   Male   248
## 5  Actual 40 to 49 years Female    89
## 6  Actual 40 to 49 years   Male    77
## 7  Actual 50 to 59 years Female    53
## 8  Actual 50 to 59 years   Male    34
## 9  Actual 60 to 69 years Female    15
## 10 Actual 60 to 69 years   Male    11
## 11 Actual 70 to 79 years Female     1
## 12 Actual 70 to 79 years   Male     1
```

Next we clean the Census Bureau's dataset and scale the expected number of individuals to our dataset size of 1645.

```
expected <-
    census_age_gender %>%
        filter(row_number() <= 19) %>%
        select(age, male, female) %>%
        mutate(age = gsub("\\.","", age)) %>%
        filter(!(age %in% c('Under 5 years','All ages','5 to 9 years','10 to 14 years','15 to 19 years')
        mutate(age_bucket = paste0(substring(age,1,1), "0 to ", substring(age,1,1),"9 years")) %>%
        mutate(age_bucket = ifelse(age_bucket == "80 to 89 years", "80 years plus", age_bucket)) %>%
        select(-age) %>%
        gather(gender, n, -age_bucket) %>%
        group_by(age_bucket, gender) %>%
        summarise(n = sum(n)) %>%
        ungroup() %>%
        mutate(gender = paste0(toupper(substring(gender,1,1)), substring(gender, 2, 999))) %>%
        mutate(percent = n / sum(n)) %>%
        mutate(Expected = actual_size * percent) %>%
        select(age_bucket, gender, Expected) %>%
        gather(source, n, -age_bucket, -gender) %>%
```

```
        select(source, age_bucket, gender, n)

expected
```

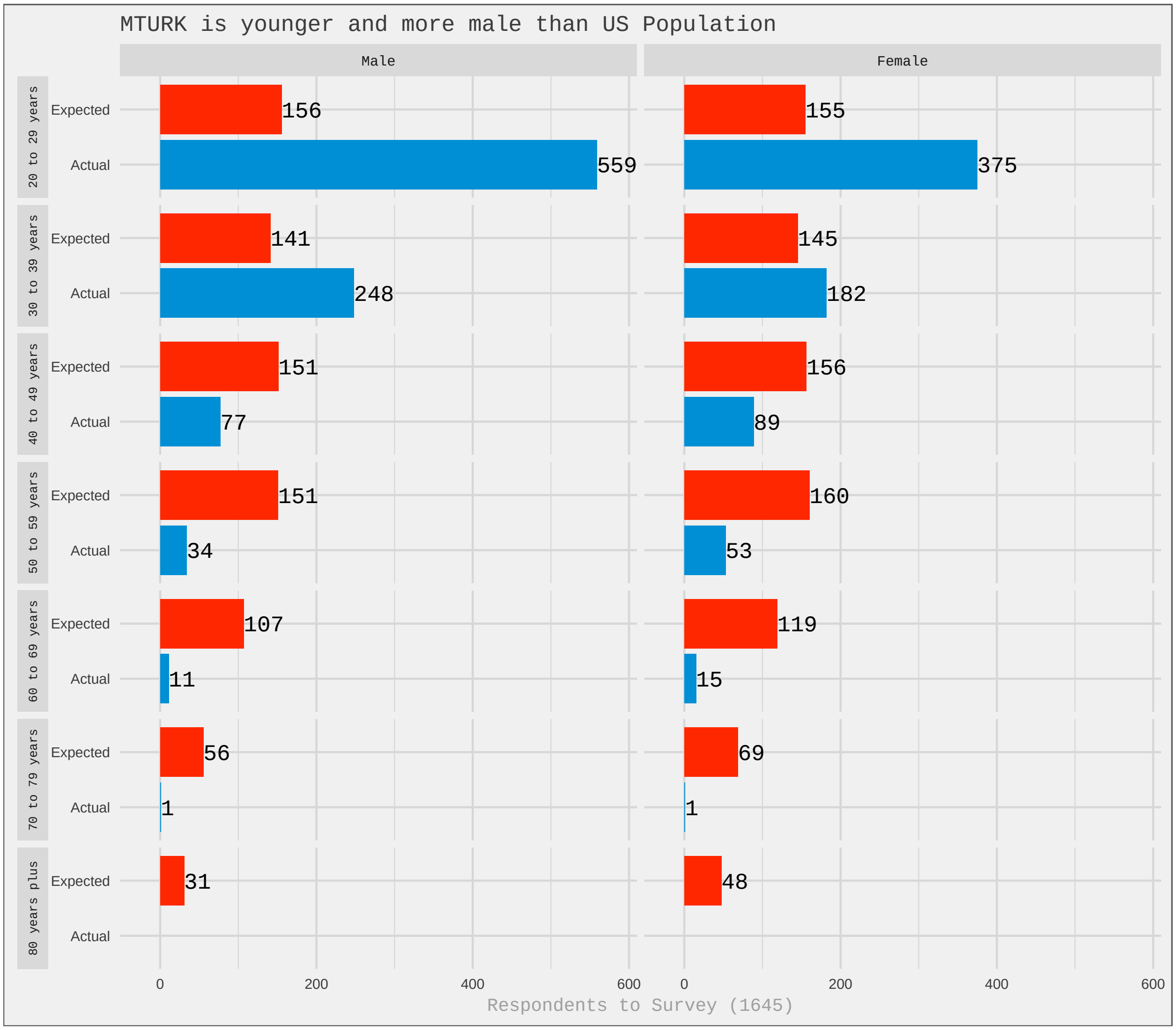
```
## # A tibble: 14 × 4
##      source     age_bucket gender      n
##       <chr>          <chr>  <chr>  <dbl>
## 1  Expected 20 to 29 years Female 155.12
## 2  Expected 20 to 29 years   Male 155.57
## 3  Expected 30 to 39 years Female 145.36
## 4  Expected 30 to 39 years   Male 141.32
## 5  Expected 40 to 49 years Female 156.41
## 6  Expected 40 to 49 years   Male 151.37
## 7  Expected 50 to 59 years Female 160.23
## 8  Expected 50 to 59 years   Male 150.98
## 9  Expected 60 to 69 years Female 118.85
## 10 Expected 60 to 69 years   Male 107.06
## 11 Expected 70 to 79 years Female  68.82
## 12 Expected 70 to 79 years   Male  55.50
## 13 Expected  80 years plus Female  47.66
## 14 Expected  80 years plus   Male  30.73
```

Then we can combine the two.

```
actual_expected <-
    union(actual, expected) %>%
        mutate(
            source = factor(source, levels = c("Actual","Expected"))
            , gender = factor(gender, levels = c("Male","Female"))
        )
```

We find that the MTURK sample is younger and more male the US population. For example, in a sample 1645 we would expect to find 155.5733 males, 20 to 29 years old. However, in the MTURK sample there are 559 males, 20 to 29 years old, or 403.4267 more than expected. In addition, in the US population we would expect 2, 2, 852.4531, 51.8% females and 2, 1, 792.5469, 48.2% males. However, the MTUK sample has 1, 2, 715, 43.5% females and 1, 1, 930, 56.5% males.

```
actual_expected %>%
    ggplot(aes(x = source, y = n, fill = source)) +
    geom_bar(stat = "identity") +
    coord_flip() +
    facet_grid(age_bucket ~ gender, switch = "y") +
    theme_jrf() +
    labs(title = "MTURK is younger and more male than US Population",
         y = paste0("Respondents to Survey (", actual_size, ")"), x = NULL) +
    scale_fill_manual(values = c(Actual = pal538['blue'][[1]], Expected = pal538['red'][[1]])) +
    guides(fill = FALSE) +
    geom_text(aes(label = round(n, 0)), hjust = 0, family = "DecimaMonoPro") +
    scale_y_continuous(expand = c(0.02, 40)) +
    theme(strip.text.y = element_text(size = 6))
```



Looking at education, we download data the US Census Bureau's Current Population Report that shows statistics on educational attainment. The data is by age and gender, but we aggregate the age section to the total population to compare to the MTURK sample. The table below shows expected vs actual proportions.

```
census_edu <- read_csv(url("https://www.census.gov/hhes/socdemo/education/data/cps/2015/Table%201-01.csv
                             skip = 5)
```

```
## Parsed with column specification:
## cols(
##   .default = col_number(),
##   X1 = col_character(),
##   None = col_integer(),
##   `Doctoral degree` = col_character(),
##   X18 = col_character(),
##   X19 = col_character(),
##   X20 = col_character()
## )

## See spec(...) for full column specifications.
```

```r
edu_expected <-
    census_edu %>%
    select(1, 3:17) %>%
    filter(row_number() %in% c(2:15)) %>%
    select(-X1) %>%
    mutate(`Doctoral degree` = as.integer(gsub(",","",`Doctoral degree`))) %>%
    summarise_each(funs(sum(., na.rm =TRUE))) %>%
    gather(education, n) %>%
    mutate(
        education = ifelse(education == "None", "Other",
                        ifelse(education %in% c("1st - 4th grade","5th - 6th grade","7th - 8th grade","9
                                                "10th grade","11th grade /2"), "Less than 12 years; n
                        ifelse(education == "High school graduate", "High school graduate (or equivalent
                        ifelse(education %in% c("Some college, no degree","Associate's degree, occupatio
                                              "Associate's degree, academic"),
                                              "Some college, no diploma; or Associate's degree",
                        ifelse(education == "Bachelor's degree", "Bachelor's degree or other 4-year deg
                        ifelse(education %in% c("Master's degree","Professional degree","Doctoral degree
                               "Graduate or professional degree", NA))))))
        , education = factor(education, levels = c('Less than 12 years; no high school diploma'
                                                   , 'High school graduate (or equivalent)'
                                                   , 'Some college, no diploma; or Associate's deg
                                                   , 'Bachelor's degree or other 4-year degree'
                                                   , 'Graduate or professional degree'
                                                   , 'Other'))
    ) %>%
    group_by(education) %>%
    summarise(expected_n = sum(n)) %>%
    ungroup() %>%
    mutate(expected = expected_n / sum(expected_n))

edu_actual <-
    survey_results_final %>%
    group_by(education) %>%
    summarise(actual_n = n()) %>%
    ungroup() %>%
    mutate(actual = actual_n / sum(actual_n))

comparison_tbl_edu <-
        inner_join(edu_expected, edu_actual, by = c("education")) %>%
        mutate(
            expected = paste0(round(100*expected,1), "%")
            , actual = paste0(round(100*actual,1), "%")
        ) %>%
        select(-actual_n, -expected_n)

comparison_tbl_edu
```

```
## # A tibble: 6 × 3
##                                          education expected actual
##                                             <fctr>    <chr>  <chr>
## 1      Less than 12 years; no high school diploma    11.7%   0.6%
## 2            High school graduate (or equivalent)    29.6%  10.9%
```

```
## 3 Some college, no diploma; or Associate's degree    27.8%  42.3%
## 4        Bachelor's degree or other 4-year degree    19.6%  34.9%
## 5                 Graduate or professional degree      11%  10.2%
## 6                                           Other     0.4%   1.1%
```

We find that the MTURK sample over indexes on people have been to college or graduated from college. Noteably, in a sample of the US population we would expect to find 27.8% of people to have 'Some college, no diploma; or Associate's degree', but in the MTURK sample 42.3% fit this category of educational attainment.

We use a proportions test to determine if the proportions are indeed different.

```
edu_matrix <- inner_join(edu_expected, edu_actual, by = c("education")) %>% select(expected_n, actual_n)
(edu_prop_test <- prop.test(edu_matrix))
```

```
##
##  6-sample test for equality of proportions without continuity
##  correction
##
## data:  edu_matrix
## X-squared = 760, df = 5, p-value <0.0000000000000002
## alternative hypothesis: two.sided
## sample estimates:
## prop 1 prop 2 prop 3 prop 4 prop 5 prop 6
## 0.9999 0.9991 0.9962 0.9955 0.9977 0.9926
```

Using 6-sample test for equality of proportions without continuity correction we have strong evidence against the null hypothesis that the proporotions in the education groups are the same. This provides further evidence that the MTURK sample does not represent the US population.

Looking at income, we download from the US Census Bureau statistics on personal income.

```
download.file("http://www2.census.gov/programs-surveys/cps/tables/pinc-01/2016/pinc01_1_1_1.xls",
              destfile = paste0(data_dir, 'pinc01_1_1_1.xls'), mode = "wb")
income <- read_excel(paste0(data_dir, 'pinc01_1_1_1.xls'), skip = 8)

income_expected <-
    income[, c(4:44)] %>%
    filter(row_number() == 2) %>%
    gather(income, n) %>%
    mutate(
        n = as.integer(n)
    ) %>%
    select(income, n) %>%
    mutate(
        income = ifelse(row_number() <= 6, "Less than $15,000",
                        ifelse(row_number() <= 12, "$15,000 - $30,000",
                               ifelse(row_number() <= 20, "$30,000 - $50,000",
                                      ifelse(row_number() <= 30, "$50,000 - $75,000",
                                             "Above $75,000"))))
        , income = factor(income, levels = c("Less than $15,000","$15,000 - $30,000", "$30,000 - $50,000",
                                             "$50,000 - $75,000", "Above $75,000"))
    ) %>%
    group_by(income) %>%
    summarise(
```

```
        n = sum(n)
    ) %>%
    ungroup() %>%
    mutate(expected = n / sum(n)) %>%
    mutate(expected_n = n) %>%
    select(income, expected_n, expected)

income_actual <-
    survey_results_final %>%
    filter(!is.na(income)) %>%
    mutate(
        income = as.character(income)
        , income = ifelse(income %in% c("$75,000 - $150,000","Above $150,000"), "Above $75,000", income
        , income = factor(income, levels = c("Less than $15,000","$15,000 - $30,000", "$30,000 - $50,00
                                            "$50,000 - $75,000", "Above $75,000"))
    ) %>%
    group_by(income) %>%
    summarise(
        n = n()
    ) %>%
    ungroup() %>%
    mutate(actual = n / sum(n)) %>%
    mutate(actual_n = n) %>%
    select(income, actual_n, actual)

comparison_tbl_income <-
    inner_join(income_expected, income_actual, by = c("income")) %>%
        mutate(
            expected = paste0(round(100*expected,1), "%")
            , actual = paste0(round(100*actual,1), "%")
        ) %>%
        select(-actual_n, -expected_n)

comparison_tbl_income
```

```
## # A tibble: 5 × 3
##              income expected actual
##              <fctr>    <chr>  <chr>
## 1 Less than $15,000    26.7%  11.9%
## 2 $15,000 - $30,000    22.8%  20.9%
## 3 $30,000 - $50,000    20.7%  24.4%
## 4 $50,000 - $75,000    14.1%  21.4%
## 5     Above $75,000    15.7%  21.3%
```

Looking at the table above we see there is a smaller percentage of lower income respondents than expected (26.7% vs. 11.9%). In addition, there is larger percentage of high earning respondents than expected (15.7% vs. 21.3%).

We use a proportions test to determine if the proportions are indeed different.

```
income_matrix <- inner_join(income_expected, income_actual, by = c("income")) %>% select(expected_n, ac
(income_prop_test <- prop.test(income_matrix))
```

```
##
##  5-sample test for equality of proportions without continuity
##  correction
##
## data:  income_matrix
## X-squared = 260, df = 4, p-value <0.0000000000000002
## alternative hypothesis: two.sided
## sample estimates:
## prop 1 prop 2 prop 3 prop 4 prop 5
## 0.9966 0.9930 0.9910 0.9883 0.9896
```

Using 5-sample test for equality of proportions without continuity correction we have strong evidence against the null hypothesis that the proporotions in the income groups are the same. This provides further evidence that the MTURK sample does not represent the US population.

2.4.2 (2)

Though we might be concerned that our sample does not represent the MTURK population as a whole we have no evidence to support this. There should be concern that someone who opts to participate in a survey about satellite radio might be more likely to be a satelitte radio listener (unless MTURK workers are much more likely to be Sirius listeners). However, we have no evidence to support this claim.

In thinking about this question we read the article "Who are these people?" Evaluating the demographic characteristics and political preferences of MTurk survey respondents.

2.4.3 (3)

In order to estimate the number of Wharton listeners in the US we will create a 95% confidence interval of the proportion of Wharton listeners in the MTURK dataset and multiply this by the Sirius radio listeners (51.6 million).

$$\hat{p} \pm z\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

```
p_hat <-
    survey_results_final %>%
    filter(sirius == "Yes") %>%
    summarise(p_hat = sum(wharton == "Yes") / n()) %>%
    unlist()

ci <- c(p_hat - qt(0.975, nrow(survey_results_final)) * sqrt(p_hat*(1-p_hat) / nrow(survey_results_fina
        , p_hat + qt(0.975, nrow(survey_results_final)) * sqrt(p_hat*(1-p_hat) / nrow(survey_results_fi

pop_p <- p_hat * 51.6
pop_ci <- round(ci * 51.6,2)
```

We estimate the sample proportion to be **0.0501** and the 95% confidence interval to be

$$(0.0399, 0.0603)$$

Thus we estimate the size of the Wharton listeners in the US to be **2.58** million and the 95% confidence interval to be (in millions)

$$(2.06, 3.11)$$

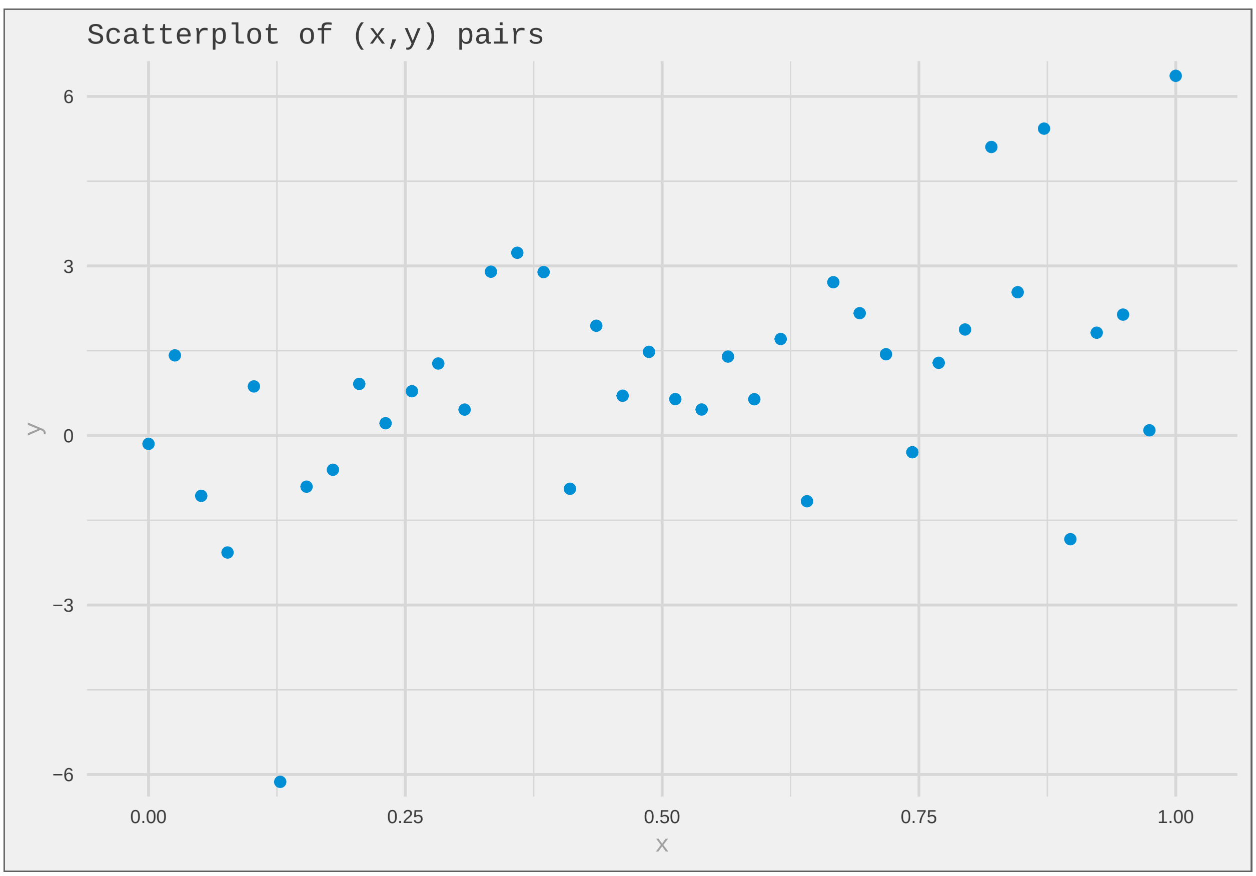
2.5 Brief Report

We have reviewed the survy of 1764 respondents of the MTURK survey. We have evidence that the survey respondents do not represent that population of the US based on the proportion of Sirius listeners (0.1605 vs 0.7729) and age, gender, income, and education characteristics. However, assuming that the sample represents the population, we estimate that there are between 2.06 and 3.11 million listeners of "Business Radio Powered by the Wharton School".

3 Question 3

3.1 Part A

```
x <- seq(0, 1, length = 40)
y <- 1 + 1.2*x + rnorm(40, mean = 0, sd = 2)

ggplot(data_frame(x, y), aes(x = x, y = y)) + geom_point(colour = pal538['blue']) +
    theme_jrf() +
    scale_x_continuous(expand = c(0.05, 0.01)) + scale_y_continuous(expand = c(0.02, 0.01)) +
    labs(title = "Scatterplot of (x,y) pairs", y = "y", x = "x")
```



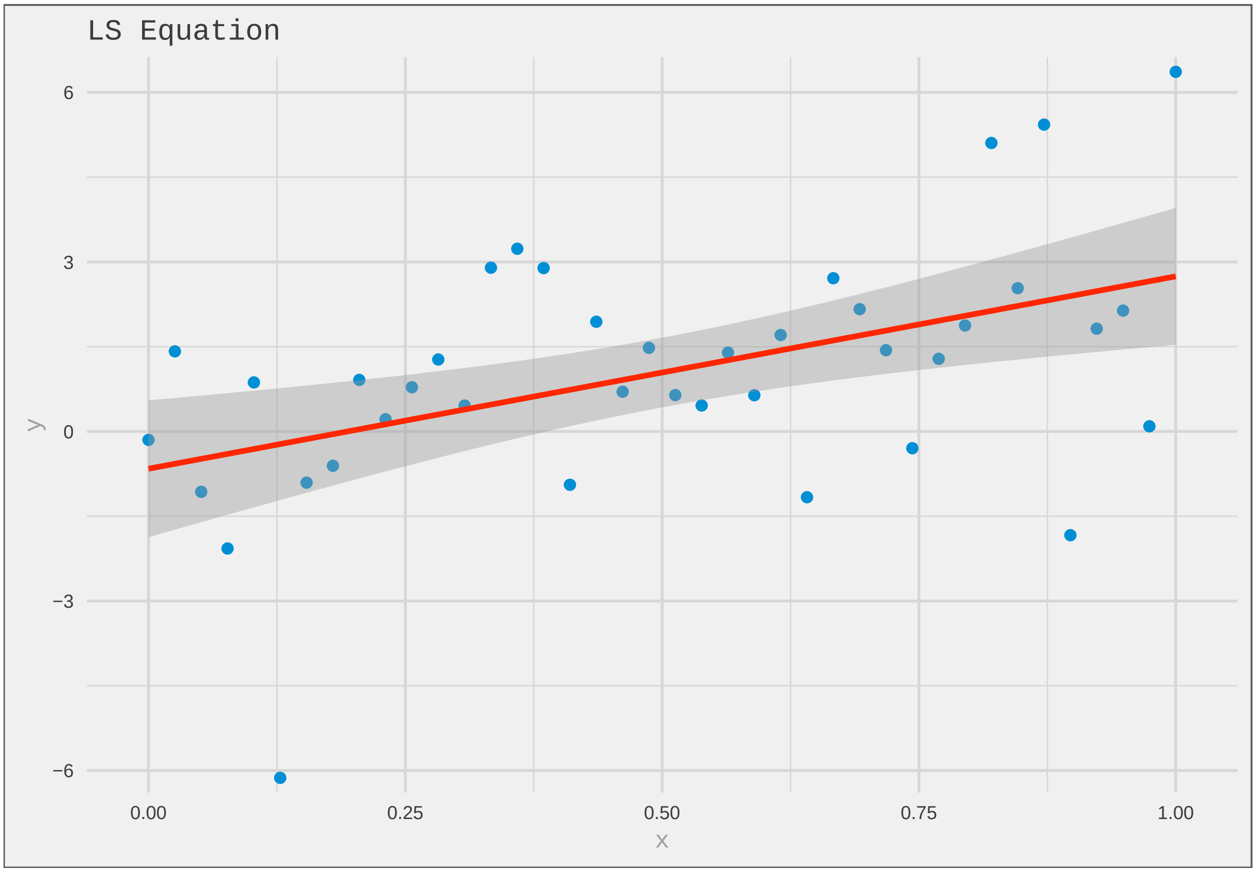
We use the the lm fuction to create a linear model.

```r
fit1 <- lm(y ~ x)
summary(fit1)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -5.907 -0.682  0.080  1.005  3.619
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.659      0.598   -1.10   0.2772
## x              3.404      1.029    3.31   0.0021 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.93 on 38 degrees of freedom
## Multiple R-squared:  0.224,  Adjusted R-squared:  0.203
## F-statistic: 10.9 on 1 and 38 DF,  p-value: 0.00207
```

We find that $\beta_0 = -0.6594$ and $\beta_1 = 3.4043$. Next we overlay LS equation on the scatterplot.

```r
ggplot(data = fit1$model, aes(x = x, y = y)) + geom_point(colour = pal538['blue']) +
    geom_smooth(method="lm", se = TRUE, colour = pal538['red']) +
    theme_jrf() +
    scale_x_continuous(expand = c(0.05, 0.01)) + scale_y_continuous(expand = c(0.02, 0.01)) +
    labs(title = "LS Equation", y = "y", x = "x")
```



The 95% confidence interval for β_1 is

$$3.4043 \pm 1.96 \times 1.0293$$

or

$$(1.3205, 5.4881)$$

This 95% confidence interval does indeed contain the true β_1 which is 1.2.

The RSE is 1.9269 which is very close to the true standard deviation of the error of $\sigma = 2$.

3.2 Part B

We begin with the given simulation code chunk:

```
x <- seq(0, 1, length = 40)
n_sim <- 100
b1 <- numeric(n_sim) # nsim many LS estimates of beta1 (=1.2)
upper_ci <- numeric(n_sim) # lower bound
lower_ci <- numeric(n_sim) # upper bound
t_star <- qt(0.975, 38)

# Carry out the simulation
for (i in 1:n_sim){
    y <- 1 + 1.2 * x + rnorm(40, sd = 2)
    lse <- lm(y ~ x)
```

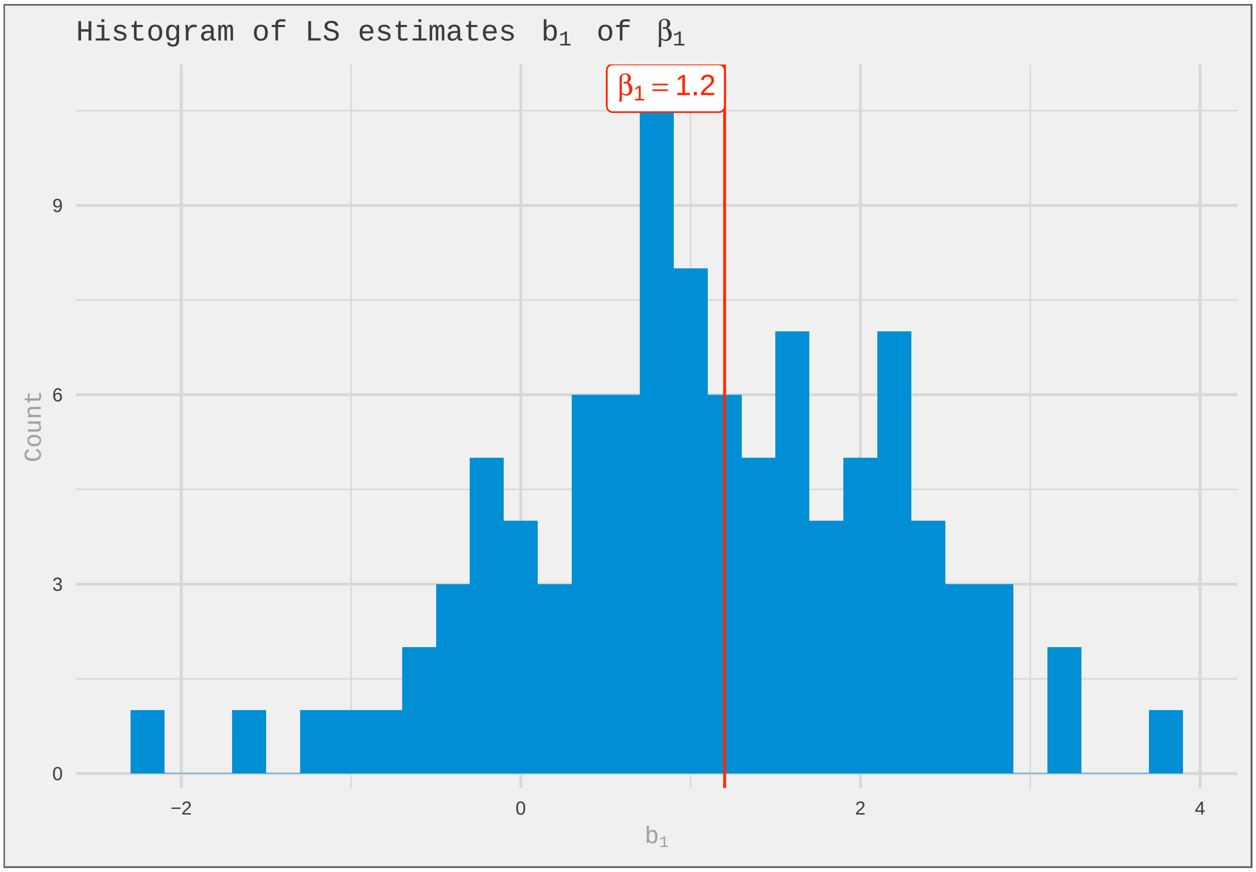
```
    lse_out <- summary(lse)$coefficients
    se <- lse_out[2, 2]
    b1[i] <- lse_out[2, 1]
    upper_ci[i] <- b1[i] + t_star * se
    lower_ci[i] <- b1[i] - t_star * se
}
```

We will summarise β_1.

```
summary(b1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -2.150   0.342   1.030   1.070   1.880   3.790
```

```
ggplot(data = data_frame(b1 = b1), aes(x = b1)) + geom_histogram(binwidth = 0.2, fill = pal538['blue'])
    geom_vline(xintercept = 1.2, colour = pal538['red']) +
    geom_label(aes(x = 1.2, y = Inf, label = 'beta[1] == 1.2'),
               vjust = "inward", hjust = "inward", parse = TRUE, colour = pal538['red']) +
    theme_jrf() +
    scale_x_continuous(expand = c(0.05, 0.01)) + scale_y_continuous(expand = c(0.02, 0.01)) +
    labs(title = expression("Histogram of LS estimates "~b[1]~" of "~beta[1]), y = "Count", x = express
```

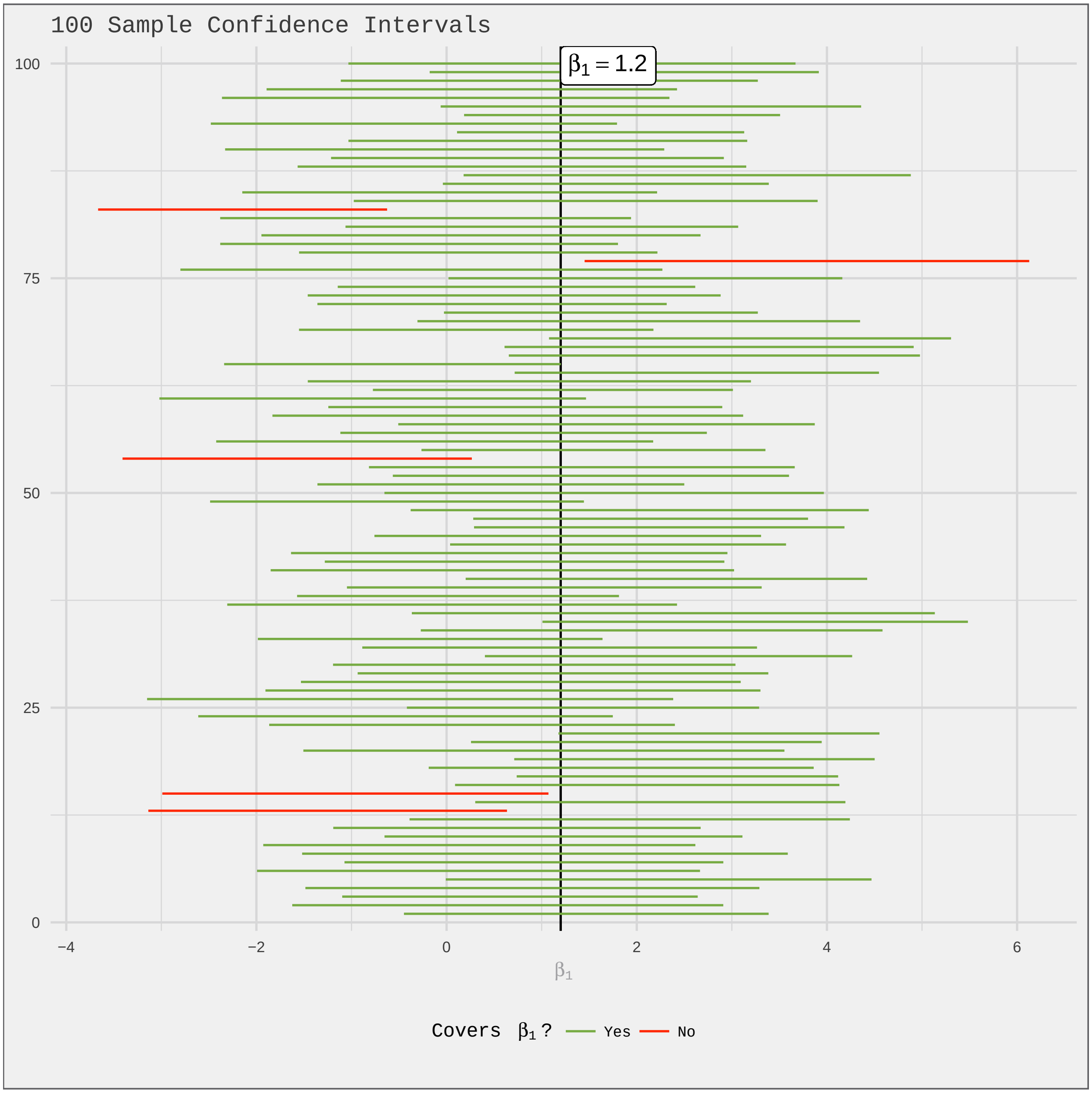


The sampling distribution does agree with the theory as most of the LS estimate of β_1 are close to 1.2.

```r
ci <- data_frame(n = 1:100, b1 = b1 , lower_ci = lower_ci, upper_ci = upper_ci,
                 covers = factor(ifelse(lower_ci < 1.2 & upper_ci > 1.2, "Yes", "No"), levels = c("Yes"
```

We find that 95 out of 100 95% confidence intervals cover the true β_1. We show this graphically below, where the red intervals do not cover the true β_1 and the green intervals do cover the true β_1.

```r
ggplot(data = ci) +
    geom_vline(xintercept = 1.2) +
    geom_segment(aes(x = lower_ci, xend = upper_ci, y = n, yend = n, colour = covers)) +
    labs(title = "100 Sample Confidence Intervals", y = NULL, x = expression(beta[1])) +
    geom_label(aes(x = 1.2, y = Inf, label = 'beta[1] == 1.2'), vjust = "inward", hjust = "inward", pars
    guides(color = guide_legend(title = expression("Covers "~beta[1]~"?"))) +
    theme(legend.position = 'bottom') +
    theme_jrf() +
    scale_x_continuous(expand = c(0.05, 0.01)) + scale_y_continuous(expand = c(0.02, 0.01)) +
    scale_colour_manual(values = c('Yes' = pal538['green'][[1]], 'No' = pal538['red'][[1]]))
```



4 Question 4

4.1 Summary

We begin by loading and tidying the ML Pay dataset.

```
# Read in the ML Pay dataset
ml_pay <- read_csv(paste0(data_dir, "MLPayData_Total.csv"))

# Let's tidy the dataset
ml_pay2 <-
    ml_pay %>%
```

```r
    rename(team = Team.name.2014) %>%
    gather(metric_raw, value, -payroll, -avgwin, -team) %>%
    mutate(
        year = as.factor(str_extract(metric_raw, "(\\d)+"))
        , metric = ifelse(substring(metric_raw,1,1) == "p", "payroll",
                          ifelse(str_detect(metric_raw, ".pct"), "avgwin", "wins"))
    ) %>%
    select(team, year, metric, value, payroll, avgwin)

ml_pay2
```

```
## # A tibble: 1,530 × 6
##                   team   year  metric value payroll avgwin
##                  <chr> <fctr>   <chr> <dbl>   <dbl>  <dbl>
## 1  Arizona Diamondbacks   1998 payroll 31.61  1.1209 0.4903
## 2        Atlanta Braves   1998 payroll 61.71  1.3817 0.5528
## 3     Baltimore Orioles   1998 payroll 71.86  1.1612 0.4538
## 4        Boston Red Sox   1998 payroll 59.50  1.9724 0.5487
## 5          Chicago Cubs   1998 payroll 49.82  1.4598 0.4737
## 6     Chicago White Sox   1998 payroll 35.18  1.3154 0.5111
## 7       Cincinnati Reds   1998 payroll 20.71  1.0248 0.4862
## 8     Cleveland Indians   1998 payroll 59.54  0.9992 0.4959
## 9      Colorado Rockies   1998 payroll 47.71  1.0262 0.4634
## 10       Detroit Tigers   1998 payroll 19.24  1.4297 0.4822
## # ... with 1,520 more rows
```

Let's do a few data quality checks, where we ensure there are 30 teams per year and there are no missing values.

```r
# Show there are 30 unique teams per year
ml_pay2 %>%
    group_by(year) %>%
    summarise(
        n = n()
        , n_distinct = n_distinct(team)
    )
```

```
## # A tibble: 17 × 3
##      year     n n_distinct
##    <fctr> <int>      <int>
## 1    1998    90         30
## 2    1999    90         30
## 3    2000    90         30
## 4    2001    90         30
## 5    2002    90         30
## 6    2003    90         30
## 7    2004    90         30
## 8    2005    90         30
## 9    2006    90         30
## 10   2007    90         30
## 11   2008    90         30
## 12   2009    90         30
```

```
## 13   2010     90          30
## 14   2011     90          30
## 15   2012     90          30
## 16   2013     90          30
## 17   2014     90          30
```

```
# Show that there are no missing values
ml_pay2 %>%
    summarise(
        na = sum(is.na(value))
        , nan = sum(is.nan(value))
    )
```

```
## # A tibble: 1 × 2
##       na   nan
##    <int> <int>
## 1      0     0
```

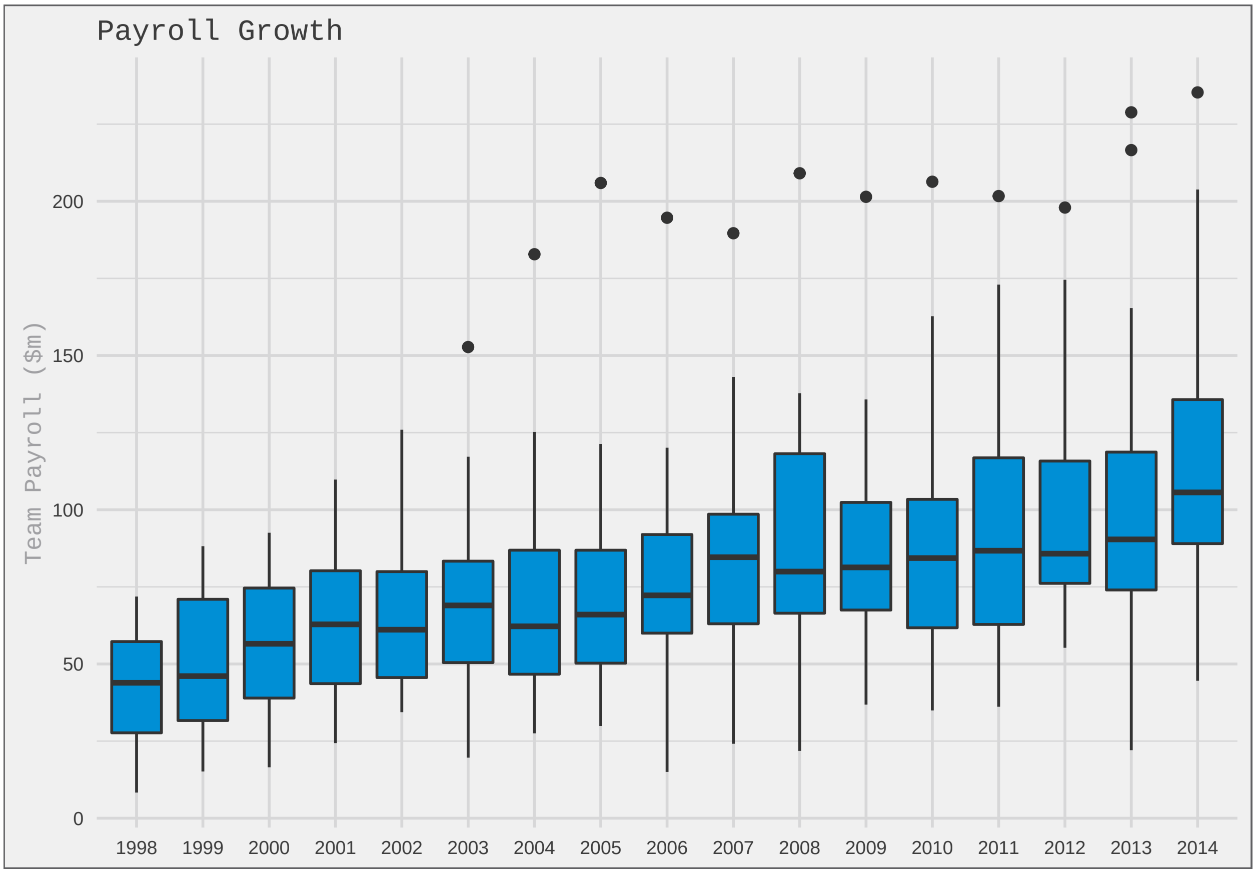
For the 17 years between 1998 and 2014, we summarise the payroll of the 30 teams.

```
ml_pay2 %>%
    filter(metric == "payroll") %>%
    group_by(year) %>%
    summarise(
          min = min(value)
        , p25 = quantile(value, .25)
        , p50 = quantile(value, .5)
        , mean = mean(value)
        , p75 = quantile(value, .75)
        , max = max(value)
    )
```

```
## # A tibble: 17 × 7
##       year    min   p25    p50   mean    p75    max
##     <fctr>  <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1     1998  8.317 27.68  43.89  41.08  57.26  71.86
## 2     1999 15.150 31.67  46.07  48.19  70.96  88.18
## 3     2000 16.520 38.94  56.54  55.66  74.61  92.54
## 4     2001 24.350 43.62  62.85  64.46  80.22 109.79
## 5     2002 34.380 45.60  61.11  67.45  79.94 125.93
## 6     2003 19.630 50.45  68.98  71.03  83.33 152.75
## 7     2004 27.518 46.67  62.21  68.55  86.89 182.84
## 8     2005 29.894 50.26  66.01  72.75  86.88 205.94
## 9     2006 14.998 59.99  72.25  77.56  91.95 194.66
## 10    2007 24.123 63.03  84.62  82.63  98.55 189.64
## 11    2008 21.811 66.45  79.95  89.55 118.18 209.08
## 12    2009 36.814 67.49  81.31  88.35 102.36 201.45
## 13    2010 34.943 61.74  84.33  91.02 103.35 206.33
## 14    2011 36.126 62.81  86.71  92.99 116.85 201.69
## 15    2012 55.245 76.13  85.75  98.02 115.79 197.96
## 16    2013 22.063 74.00  90.39 103.29 118.69 228.84
## 17    2014 44.544 89.02 105.63 115.13 135.70 235.30
```

The boxplot belows show there was a general growth in payroll spending over the 17 years in the MLB. The outlier at the high end of the payroll scale is the New York Yankees.

```
ml_pay2 %>%
    filter(metric == "payroll") %>%
    ggplot(aes(year, value)) + geom_boxplot(fill = pal538['blue']) +
    theme_jrf() +
    labs(title = "Payroll Growth", y = "Team Payroll ($m)", x = NULL)
```



Let's identify what the year-over-year (yoy) growth in payroll has been by team.

```
ml_pay2 %>%
    filter(metric == "payroll") %>%
    arrange(team, year) %>%
    group_by(team) %>%
    mutate(
        yoy_growth = (value - lag(value)) / lag(value)
    ) %>%
    group_by(team) %>%
    summarise(
        avg_yoy_growth = mean(yoy_growth, na.rm = TRUE)
    ) %>%
    ungroup() %>%
    arrange(desc(avg_yoy_growth)) %>%
    print(n = 30)
```

```
## # A tibble: 30 × 2
##                     team avg_yoy_growth
##                    <chr>          <dbl>
## 1   Washington Nationals        0.24248
## 2          Miami Marlins        0.22797
## 3        Cincinnati Reds        0.20362
## 4         Detroit Tigers        0.17466
## 5     Pittsburgh Pirates        0.16347
## 6         Tampa Bay Rays        0.13989
## 7  Philadelphia Phillies        0.13476
## 8     Kansas City Royals        0.12458
## 9      Toronto Blue Jays        0.12416
## 10  Arizona Diamondbacks        0.12251
## 11   Los Angeles Dodgers        0.12002
## 12       Minnesota Twins        0.11355
## 13     Oakland Athletics        0.10831
## 14     Chicago White Sox        0.10042
## 15     Milwaukee Brewers        0.09967
## 16    Los Angeles Angels        0.08662
## 17         Texas Rangers        0.08357
## 18  San Francisco Giants        0.08039
## 19      New York Yankees        0.07984
## 20        Boston Red Sox        0.07538
## 21   St. Louis Cardinals        0.06621
## 22      Colorado Rockies        0.05971
## 23      San Diego Padres        0.05860
## 24      Seattle Mariners        0.05759
## 25     Cleveland Indians        0.05441
## 26          Chicago Cubs        0.05143
## 27        Houston Astros        0.04524
## 28     Baltimore Orioles        0.04505
## 29        Atlanta Braves        0.04355
## 30         New York Mets        0.03761
```

Let's summarise this as the yoy growth overall.

```
avg_yoy_growth <-
    ml_pay2 %>%
        filter(metric == "payroll") %>%
        arrange(team, year) %>%
        group_by(team) %>%
        mutate(
            yoy_growth = (value - lag(value)) / lag(value)
        ) %>%
        group_by(team) %>%
        summarise(
            avg_yoy_growth = mean(yoy_growth, na.rm = TRUE)
        ) %>%
        ungroup() %>%
        summarise(
            avg_yoy_growth = mean(avg_yoy_growth)
        ) %>%
        unlist()
```

```
avg_yoy_growth
```

```
## avg_yoy_growth
##         0.1042
```

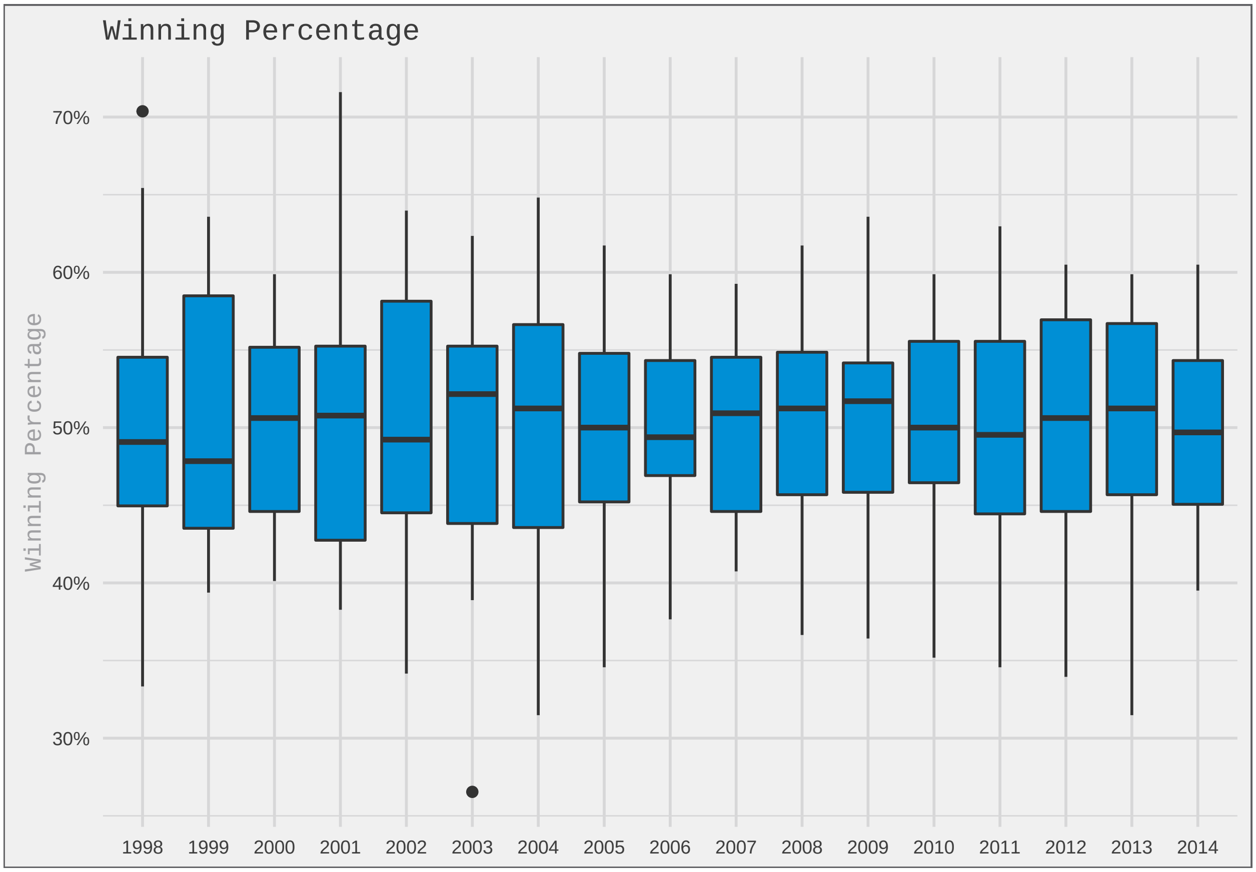
Next, we summarise the winning percentage for the 17 years. This is not particularly meaningful, but it is a way to identify any errant values.

```
ml_pay2 %>%
    filter(metric == "avgwin") %>%
    group_by(year) %>%
    summarise(
          min = min(value)
        , p25 = quantile(value, .25)
        , p50 = quantile(value, .5)
        , mean = mean(value)
        , p75 = quantile(value, .75)
        , max = max(value)
    )
```

```
## # A tibble: 17 × 7
##      year    min    p25    p50   mean    p75    max
##    <fctr>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1    1998 0.3333 0.4496 0.4907 0.5000 0.5453 0.7037
## 2    1999 0.3937 0.4352 0.4784 0.4999 0.5849 0.6358
## 3    2000 0.4012 0.4460 0.5062 0.5000 0.5518 0.5988
## 4    2001 0.3827 0.4275 0.5077 0.5000 0.5525 0.7160
## 5    2002 0.3416 0.4451 0.4923 0.5000 0.5814 0.6398
## 6    2003 0.2654 0.4383 0.5216 0.5000 0.5525 0.6235
## 7    2004 0.3148 0.4357 0.5123 0.4999 0.5664 0.6481
## 8    2005 0.3457 0.4522 0.5000 0.5000 0.5478 0.6173
## 9    2006 0.3765 0.4691 0.4938 0.5000 0.5432 0.5988
## 10   2007 0.4074 0.4460 0.5093 0.5000 0.5453 0.5926
## 11   2008 0.3665 0.4568 0.5123 0.5000 0.5485 0.6173
## 12   2009 0.3642 0.4583 0.5170 0.5000 0.5417 0.6358
## 13   2010 0.3519 0.4645 0.5000 0.5000 0.5556 0.5988
## 14   2011 0.3457 0.4444 0.4954 0.5000 0.5556 0.6296
## 15   2012 0.3395 0.4460 0.5062 0.5000 0.5694 0.6049
## 16   2013 0.3148 0.4568 0.5123 0.5000 0.5670 0.5988
## 17   2014 0.3951 0.4506 0.4969 0.5000 0.5432 0.6049
```

The boxplot below shows the dispersion of winning percentage overtime. You can see the dot in 2003 is the Detriot Tigers who lost more games than any American League team in history (43-119).

```
ml_pay2 %>%
    filter(metric == "avgwin") %>%
    ggplot(aes(year, value)) + geom_boxplot(fill = pal538['blue']) +
    scale_y_continuous(labels = scales::percent) +
    theme_jrf() +
    labs(title = "Winning Percentage", y = "Winning Percentage", x = NULL)
```



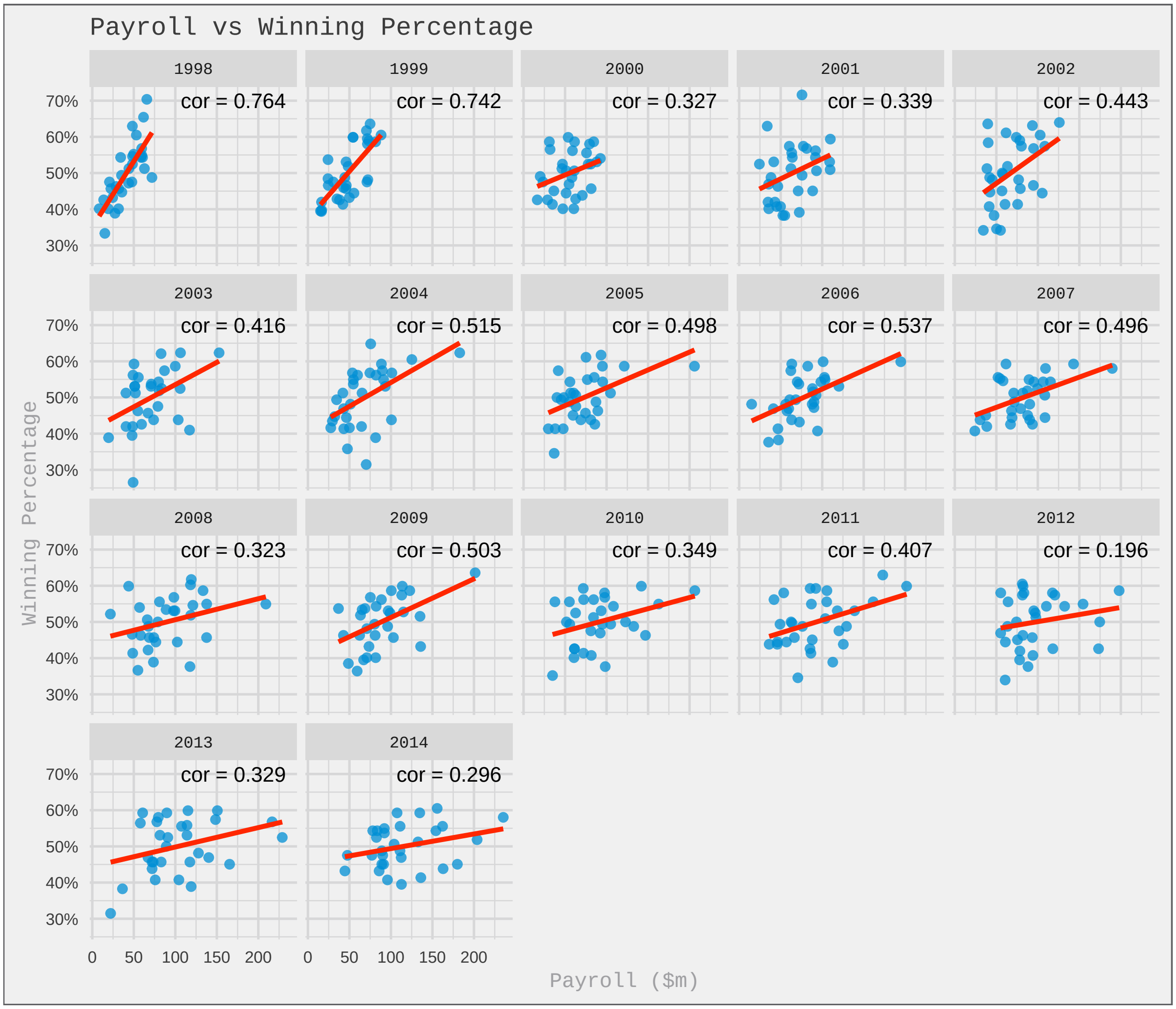
Let's summarise the two variables across time to get an idea of where the values fall.

```
ml_pay2 %>%
    filter(metric %in% c("payroll","avgwin")) %>%
    group_by(metric) %>%
    summarise(
          min = min(value)
        , p25 = quantile(value, .25)
        , p50 = quantile(value, .5)
        , mean = mean(value)
        , p75 = quantile(value, .75)
        , max = max(value)
    ) %>%
    kable()
```

metric	min	p25	p50	mean	p75	max
avgwin	0.2654	0.4444	0.50	0.5	0.5556	0.716
payroll	8.3170	51.3329	73.34	78.1	94.9997	235.295

Next, let's look at scatter plots of payroll vs winning percentage over the 17 years. This plot helps highlight the fact that average payroll increases over time.

```r
ml_pay2 %>%
    filter(metric %in% c("payroll","avgwin")) %>%
    select(-payroll, -avgwin) %>%
    spread(metric, value) %>%
    ggplot(aes(x = payroll, y = avgwin)) + facet_wrap(~ year) +
    geom_point(colour = pal538['blue'], alpha = 0.75) +
    geom_smooth(method = "lm", se = FALSE, colour = pal538['red']) +
    scale_y_continuous(labels = scales::percent) +
    labs(title = "Payroll vs Winning Percentage", y = "Winning Percentage", x = "Payroll ($m)") +
    theme_jrf() +
    geom_text(data =
                  . %>%
                  group_by(year) %>%
                  summarise(
                      cor = cor(payroll, avgwin)
                  ),
        aes(x = 170, y = .7, label = paste0("cor = ", round(cor, 3))),
            size = 3
        )
```



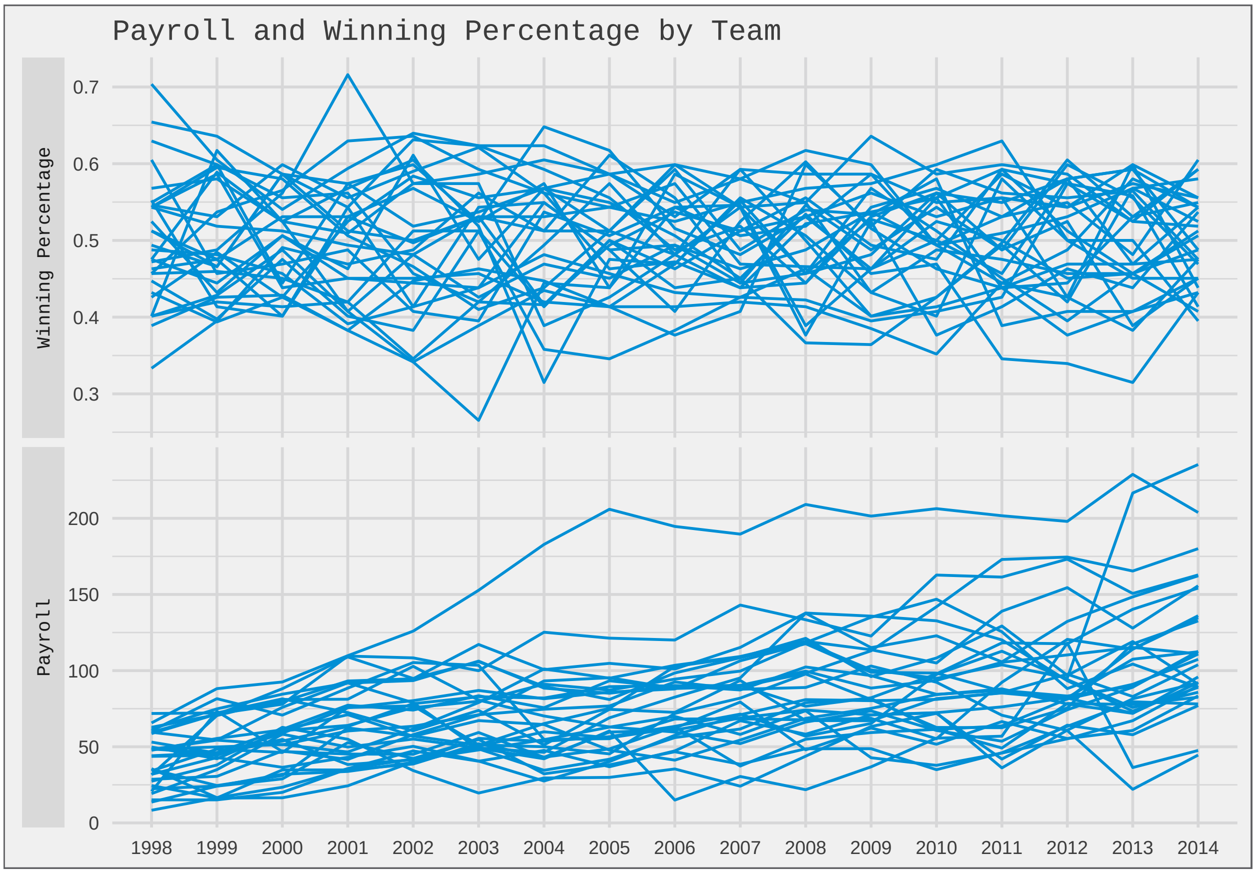

```r
avg_person_cor <-
    ml_pay2 %>%
        filter(metric %in% c("payroll","avgwin")) %>%
        select(-payroll, -avgwin) %>%
        spread(metric, value) %>%
        group_by(year) %>%
        summarise(
            cor = cor(payroll, avgwin)
        ) %>%
        ungroup() %>%
        summarise(
            cor = mean(cor)
        ) %>%
        unlist()

# Avg Person Correlation
avg_person_cor
```

```
##     cor
```

```
## 0.4402
```

Let's show the trends in the two variable by team.

```
ml_pay2 %>%
    filter(metric %in% c("payroll","avgwin")) %>%
    ggplot(aes(x = year, y = value, group = team)) +
    facet_grid(metric ~ ., scales = "free_y", switch = "y",
               labeller = ggplot2::labeller(metric = c(avgwin = "Winning Percentage", payroll = "Payrol
    geom_line(colour = pal538['blue']) +
    guides(color = FALSE) +
    theme_jrf() +
    labs(title = "Payroll and Winning Percentage by Team", y = NULL, x = NULL)
```



Summary of Exploratory Analysis:

1. Payroll has generally been increasing - the yoy average growth is **10.42%**.
2. There does appear to be a linear relationship between payroll and winning percentage, in a given year. The average Person coorrlation coefficient is **0.44**.

4.2 Prediction

Let's build a linear model to predict winning percentage for each of the 17 years. The best way to do this is using nested data frames (tidyr), purrr, and broom.

```r
lm_by_year <-
  ml_pay2 %>%
  filter(metric %in% c("payroll","avgwin")) %>%
  select(-payroll, -avgwin) %>%
  spread(metric, value) %>%
  group_by(year) %>%
  nest() %>%
  mutate(
    model = purrr::map(data, ~ lm(avgwin ~ payroll, data = .))
  )
```

Below is a summary of each model. It appears that some of the models are significant at 95% confidence level, but a number of models are not, noteably 2012, 2014, and 2000. If we look back at the Payroll vs Winning Percentage plot, we can see that correlation for these years are lower than others.

```r
lm_by_year %>%
    unnest(model %>% purrr::map(broom::glance)) %>%
    select(year, r.squared, adj.r.squared, sigma, statistic, p.value)
```

```
## # A tibble: 17 × 6
##      year r.squared adj.r.squared   sigma statistic      p.value
##    <fctr>     <dbl>         <dbl>   <dbl>     <dbl>        <dbl>
## 1    1998   0.58437       0.56953 0.05464    39.368 0.0000008775
## 2    1999   0.55070       0.53465 0.05208    34.319 0.0000026815
## 3    2000   0.10711       0.07522 0.05935     3.359 0.0774960126
## 4    2001   0.11481       0.08319 0.07705     3.631 0.0670117508
## 5    2002   0.19645       0.16775 0.08351     6.845 0.0141629385
## 6    2003   0.17330       0.14378 0.07643     5.870 0.0221235136
## 7    2004   0.26524       0.23900 0.07267    10.108 0.0035884916
## 8    2005   0.24842       0.22157 0.05901     9.255 0.0050598002
## 9    2006   0.28861       0.26320 0.05341    11.360 0.0022044097
## 10   2007   0.24578       0.21885 0.05052     9.125 0.0053365405
## 11   2008   0.10431       0.07232 0.06573     3.261 0.0817202932
## 12   2009   0.25340       0.22674 0.06188     9.503 0.0045725869
## 13   2010   0.12183       0.09046 0.06478     3.884 0.0586948418
## 14   2011   0.16570       0.13590 0.06551     5.561 0.0255815499
## 15   2012   0.03848       0.00414 0.07351     1.121 0.2988433933
## 16   2013   0.10846       0.07662 0.07250     3.406 0.0755396367
## 17   2014   0.08765       0.05507 0.05760     2.690 0.1121612901
```

Below is the full summary of the model for 1998 and note that the results match the 1998 record above.

```r
summary(lm_by_year$model[[1]])
```

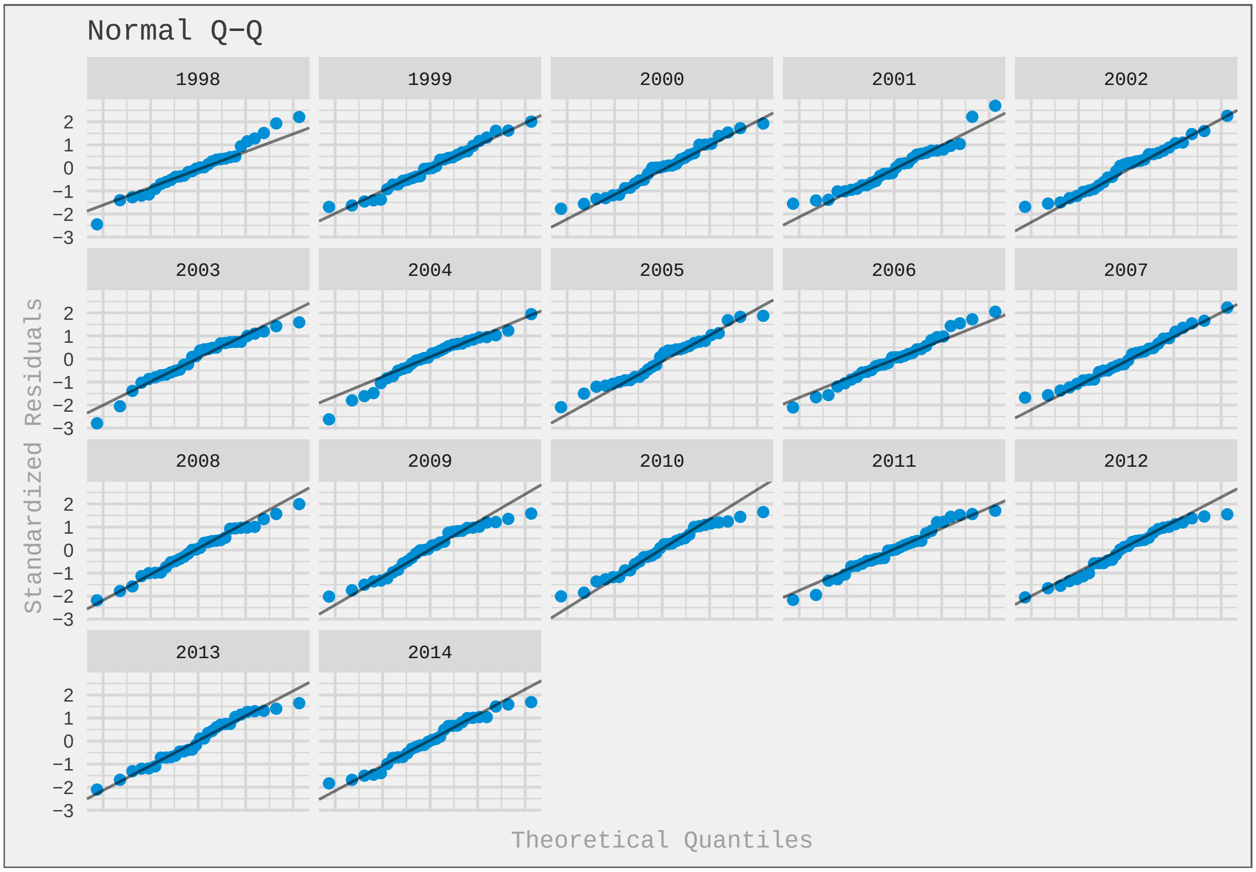
```
## 
## Call:
## lm(formula = avgwin ~ payroll, data = .)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.12418 -0.03151 -0.00042  0.02347  0.11439 
## 
```

```
## Coefficients:
##             Estimate Std. Error t value          Pr(>|t|)
## (Intercept) 0.350656   0.025803   13.59 0.000000000000075 ***
## payroll     0.003635   0.000579    6.27 0.000000877530631 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0546 on 28 degrees of freedom
## Multiple R-squared:  0.584,  Adjusted R-squared:  0.57
## F-statistic: 39.4 on 1 and 28 DF,  p-value: 0.000000878
```

However, before we interpret these models let's check our model assumptions, aside from linearity, we need to check (1) normality and (2) equal variance of the residuals.

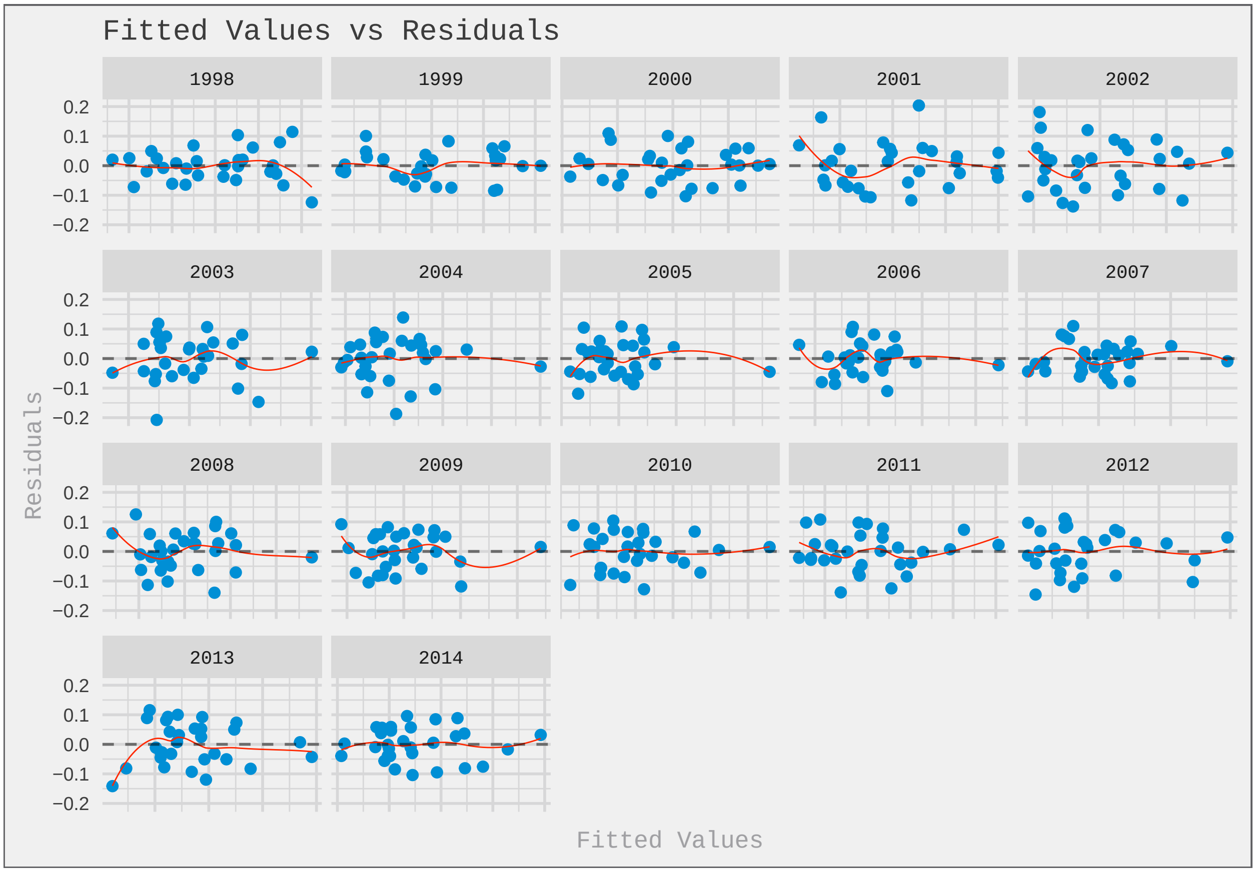
THe normal Q-Q plots below show that the residuals are approximately normal.

```
lm_by_year %>%
    unnest(model %>% purrr::map(broom::augment)) %>%
    ggplot() +
    facet_wrap(~ year) +
    stat_qq(aes(sample = .std.resid), colour = pal538['blue']) +
    geom_abline(data =
        . %>%
        group_by(year) %>%
        summarise(
            slope = diff(quantile(.std.resid, c(0.25, 0.75))) / diff(qnorm(c(0.25, 0.75)))
            , int = quantile(.std.resid, c(0.25, 0.75))[1L] -
               (diff(quantile(.std.resid, c(0.25, 0.75))) /
                    diff(qnorm(c(0.25, 0.75)))) * qnorm(c(0.25, 0.75))[1L]
        ),
        aes(slope = slope, intercept = int), alpha = 0.5
    ) +
    theme_jrf() +
    scale_x_continuous(labels = NULL) +
    labs(title = "Normal Q-Q", y = "Standardized Residuals", x = "Theoretical Quantiles")
```



The fitted values vs residuals plots show approximately equal variance of the residuals (i.e. no heteroscedasticity).

```
lm_by_year %>%
    unnest(model %>% purrr::map(broom::augment)) %>%
    ggplot(aes(x = .fitted, y = .resid)) +
    facet_wrap(~ year, scale = "free_x") +
    geom_point(colour = pal538['blue']) +
    geom_smooth(method = "loess", colour = pal538['red'], se = FALSE, size = .25, alpha = 0.5) +
    geom_hline(yintercept = 0, alpha = 0.5, linetype = 'dashed', color = 'black') +
    theme_jrf() +
    scale_x_continuous(labels = NULL) +
    labs(title = "Fitted Values vs Residuals", y = "Residuals", x = "Fitted Values")
```



Having checked the model assumptions, we can look at the β that have been estimated by the models. Below are the 34 coefficients (17 models with an intercept term and a coefficient for payroll). The p-values show that for many of the estimated coefficients we do not have enough evidence to reject the null hypothesis that the coefficients differ from 0.

```
lm_by_year %>%
    unnest(model %>% purrr::map(broom::tidy)) %>%
    print(n = 34)
```

```
## # A tibble: 34 × 6
##      year        term  estimate std.error statistic               p.value
##    <fctr>       <chr>     <dbl>     <dbl>     <dbl>                 <dbl>
## 1    1998 (Intercept) 0.3506556 0.0258026    13.590 0.0000000000000749604
## 2    1998     payroll 0.0036346 0.0005793     6.274 0.0000008775306309740
## 3    1999 (Intercept) 0.3728181 0.0236785    15.745 0.0000000000000019304
## 4    1999     payroll 0.0026359 0.0004500     5.858 0.0000026814798618320
## 5    2000 (Intercept) 0.4473224 0.0307196    14.561 0.0000000000000136732
## 6    2000     payroll 0.0009464 0.0005164     1.833 0.0774960126417996303
## 7    2001 (Intercept) 0.4288775 0.0398939    10.750 0.0000000000190275152
## 8    2001     payroll 0.0011036 0.0005791     1.906 0.0670117508225681613
## 9    2002 (Intercept) 0.3893265 0.0449764     8.656 0.0000000021026331511
## 10   2002     payroll 0.0016414 0.0006274     2.616 0.0141629384856042442
## 11   2003 (Intercept) 0.4126698 0.0386554    10.676 0.0000000000222959931
## 12   2003     payroll 0.0012296 0.0005075     2.423 0.0221235135867171792
## 13   2004 (Intercept) 0.4095794 0.0313668    13.058 0.0000000000001979586
```

```
## 14   2004     payroll 0.0013182 0.0004146     3.179 0.0035884916201666703
## 15   2005 (Intercept) 0.4282620 0.0259255    16.519 0.0000000000000005715
## 16   2005     payroll 0.0009861 0.0003242     3.042 0.0050598001802222899
## 17   2006 (Intercept) 0.4196590 0.0257541    16.295 0.0000000000000008091
## 18   2006     payroll 0.0010359 0.0003073     3.370 0.0022044097000841812
## 19   2007 (Intercept) 0.4309460 0.0246449    17.486 0.0000000000000001331
## 20   2007     payroll 0.0008354 0.0002766     3.021 0.0053365404542587511
## 21   2008 (Intercept) 0.4479322 0.0312141    14.350 0.0000000000000196417
## 22   2008     payroll 0.0005811 0.0003218     1.806 0.0817202932435047297
## 23   2009 (Intercept) 0.4057345 0.0325884    12.450 0.0000000000006216563
## 24   2009     payroll 0.0010666 0.0003460     3.083 0.0045725869170165955
## 25   2010 (Intercept) 0.4435907 0.0309690    14.324 0.0000000000000205654
## 26   2010     payroll 0.0006197 0.0003144     1.971 0.0586948418081610773
## 27   2011 (Intercept) 0.4345011 0.0302412    14.368 0.0000000000000190576
## 28   2011     payroll 0.0007044 0.0002987     2.358 0.0255815498958609985
## 29   2012 (Intercept) 0.4615409 0.0387310    11.917 0.0000000000017559780
## 30   2012     payroll 0.0003924 0.0003706     1.059 0.2988433933285373212
## 31   2013 (Intercept) 0.4444975 0.0328433    13.534 0.0000000000000829223
## 32   2013     payroll 0.0005371 0.0002910     1.846 0.0755396367276400110
## 33   2014 (Intercept) 0.4535563 0.0302062    15.015 0.0000000000000063621
## 34   2014     payroll 0.0004034 0.0002459     1.640 0.1121612901459839856
```

We find that in some years, payroll is a significant variable in predicting winning percentage while in others it is not. We might consider using previous years payroll to predict winning percentage.

4.3 Aggregated Information

Using the aggregated data provided in *MLPayData_Total.csv* we create linear regression to predict average winning percentage.

```r
fit1 <- lm(avgwin ~ payroll, data = ml_pay2 %>% select(team, payroll, avgwin) %>% distinct())
summary(fit1)
```

```
##
## Call:
## lm(formula = avgwin ~ payroll, data = ml_pay2 %>% select(team,
##     payroll, avgwin) %>% distinct())
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.04003 -0.01749  0.00094  0.01095  0.07030
##
## Coefficients:
##             Estimate Std. Error t value             Pr(>|t|)
## (Intercept)   0.4226     0.0153   27.56 < 0.0000000000000002 ***
## payroll       0.0614     0.0117    5.23             0.000015 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.027 on 28 degrees of freedom
## Multiple R-squared:  0.494,  Adjusted R-squared:  0.476
## F-statistic: 27.4 on 1 and 28 DF,  p-value: 0.0000147
```

We find that model is significant with an F-statistic of 27.38.

4.3.1 Red Sox

```
red_sox <- ml_pay2 %>% select(team, payroll, avgwin) %>% distinct() %>% filter(team == "Boston Red Sox")
(red_sox_interval <- predict(fit1, red_sox, interval = "prediction", level = .95))
```

```
##      fit    lwr    upr
## 1 0.5436 0.4848 0.6025
```

The 95% prediction interval for the Boston Red Sox is $(0.4848, 0.6025)$ and their winning percentage is 0.5487. In other words, the model does quite well in predicting the Boston Red Sox's winning percentage over the 17 year period.

4.3.2 Oakland A's

```
oakland <- ml_pay2 %>% select(team, payroll, avgwin) %>% distinct() %>% filter(team == "Oakland Athleti
(oakland_interval <- predict(fit1, oakland, interval = "prediction", level = .95))
```

```
##      fit    lwr    upr
## 1 0.4742 0.4172 0.5312
```

The 95% prediction interval for the Oakland Athletics is $(0.4172, 0.5312)$ and their winning percentage is 0.5445. In other words, the model under-predicting the Oakland Athletic's winning percentage over the 17 year period as it's outside the prediction interval. This was an expected result as Billy Beane was the general manager for the A's during this period.

4.4 Best Model with Historicals

To build a model to best predict the winning percentage in 2014, we'll use the payroll and winning percentage from previous years. We will use the last 5 years of winning percentages and payroll figures. We use our domain knowledge to assume that data more than 5 years back will not have an influence on the current season.

```
last_5years <-
    ml_pay2 %>%
        filter(metric %in% c("payroll","avgwin")) %>%
        select(-payroll, -avgwin) %>%
        arrange(team, year) %>%
        spread(metric, value) %>%
        group_by(team) %>%
        mutate(
              payroll_lag1 = lag(payroll, 1)
            , payroll_lag2 = lag(payroll, 2)
            , payroll_lag3 = lag(payroll, 3)
            , payroll_lag4 = lag(payroll, 4)
            , payroll_lag5 = lag(payroll, 5)
            , avgwin_lag1 = lag(avgwin, 1)
            , avgwin_lag2 = lag(avgwin, 2)
            , avgwin_lag3 = lag(avgwin, 3)
```

```r
           , avgwin_lag4 = lag(avgwin, 4)
           , avgwin_lag5 = lag(avgwin, 5)
       ) %>%
       ungroup() %>%
       filter(year == 2014) %>%
       select(-payroll, -year)

summary(lm(avgwin ~ . -team, data = last_5years))
```

```
##
## Call:
## lm(formula = avgwin ~ . - team, data = last_5years)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.09422 -0.01651  0.00883  0.02237  0.06299
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.525603   0.115091    4.57  0.00021 ***
## payroll_lag1  0.000450   0.000339    1.33  0.20019
## payroll_lag2  0.000828   0.000626    1.32  0.20133
## payroll_lag3 -0.000859   0.000772   -1.11  0.27985
## payroll_lag4  0.000035   0.000898    0.04  0.96933
## payroll_lag5  0.000340   0.000770    0.44  0.66390
## avgwin_lag1   0.088470   0.162141    0.55  0.59167
## avgwin_lag2   0.515698   0.179432    2.87  0.00972 **
## avgwin_lag3  -0.528842   0.216923   -2.44  0.02477 *
## avgwin_lag4  -0.338486   0.190392   -1.78  0.09144 .
## avgwin_lag5   0.050099   0.214129    0.23  0.81751
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0463 on 19 degrees of freedom
## Multiple R-squared:   0.6,   Adjusted R-squared:  0.39
## F-statistic: 2.85 on 10 and 19 DF,  p-value: 0.0237
```

We will iteratively remove the explanatory variable that has the largest p-value for the coefficient estimate.

```r
summary(lm(avgwin ~ . -team -payroll_lag4, data = last_5years))
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4, data = last_5years)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.09415 -0.01685  0.00889  0.02227  0.06232
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.526261   0.110967    4.74  0.00012 ***
```

```
## payroll_lag1  0.000449   0.000329    1.36  0.18831
## payroll_lag2  0.000832   0.000601    1.39  0.18127
## payroll_lag3 -0.000845   0.000669   -1.26  0.22096
## payroll_lag5  0.000363   0.000488    0.74  0.46594
## avgwin_lag1   0.089320   0.156605    0.57  0.57479
## avgwin_lag2   0.513650   0.167223    3.07  0.00602 **
## avgwin_lag3  -0.530114   0.209032   -2.54  0.01966 *
## avgwin_lag4  -0.337122   0.182412   -1.85  0.07943 .
## avgwin_lag5   0.049045   0.207041    0.24  0.81516
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0451 on 20 degrees of freedom
## Multiple R-squared:   0.6,   Adjusted R-squared:  0.42
## F-statistic: 3.33 on 9 and 20 DF,  p-value: 0.0119
```

```r
summary(lm(avgwin ~ . -team -payroll_lag4 -avgwin_lag5, data = last_5years))
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4 - avgwin_lag5,
##     data = last_5years)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.09380 -0.01754  0.00645  0.01984  0.06502
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.538249   0.096510    5.58 0.000016 ***
## payroll_lag1  0.000457   0.000320    1.43   0.1682
## payroll_lag2  0.000870   0.000566    1.54   0.1390
## payroll_lag3 -0.000837   0.000653   -1.28   0.2137
## payroll_lag5  0.000364   0.000477    0.76   0.4536
## avgwin_lag1   0.091051   0.152878    0.60   0.5578
## avgwin_lag2   0.506139   0.160457    3.15   0.0048 **
## avgwin_lag3  -0.532122   0.204112   -2.61   0.0165 *
## avgwin_lag4  -0.315147   0.153493   -2.05   0.0527 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0441 on 21 degrees of freedom
## Multiple R-squared:  0.599,  Adjusted R-squared:  0.446
## F-statistic: 3.92 on 8 and 21 DF,  p-value: 0.00566
```

```r
summary(lm(avgwin ~ . -team -payroll_lag4 -avgwin_lag5 -avgwin_lag1, data = last_5years))
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4 - avgwin_lag5 -
##     avgwin_lag1, data = last_5years)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -0.09901 -0.01594  0.00531  0.02218  0.06506
##
## Coefficients:
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept)   0.558474   0.089004    6.27 0.0000026 ***
## payroll_lag1  0.000509   0.000304    1.67    0.1082
## payroll_lag2  0.000883   0.000557    1.58    0.1272
## payroll_lag3 -0.000940   0.000621   -1.51    0.1442
## payroll_lag5  0.000390   0.000468    0.83    0.4136
## avgwin_lag2   0.540367   0.147599    3.66    0.0014 **
## avgwin_lag3  -0.507848   0.197046   -2.58    0.0172 *
## avgwin_lag4  -0.321684   0.150838   -2.13    0.0444 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0434 on 22 degrees of freedom
## Multiple R-squared:  0.592,  Adjusted R-squared:  0.462
## F-statistic: 4.56 on 7 and 22 DF,  p-value: 0.00282
```

```
summary(lm(avgwin ~ . -team -payroll_lag4 -avgwin_lag5 -avgwin_lag1 -payroll_lag5, data = last_5years))
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4 - avgwin_lag5 -
##     avgwin_lag1 - payroll_lag5, data = last_5years)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.10148 -0.01826  0.00261  0.02746  0.06409
##
## Coefficients:
##               Estimate Std. Error t value   Pr(>|t|)
## (Intercept)   0.586540   0.081836    7.17 0.00000027 ***
## payroll_lag1  0.000552   0.000297    1.86     0.0764 .
## payroll_lag2  0.000748   0.000530    1.41     0.1711
## payroll_lag3 -0.000604   0.000469   -1.29     0.2104
## avgwin_lag2   0.504779   0.140343    3.60     0.0015 **
## avgwin_lag3  -0.464948   0.188934   -2.46     0.0218 *
## avgwin_lag4  -0.361271   0.142207   -2.54     0.0183 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0432 on 23 degrees of freedom
## Multiple R-squared:  0.579,  Adjusted R-squared:  0.47
## F-statistic: 5.28 on 6 and 23 DF,  p-value: 0.00151
```

```
summary(lm(avgwin ~ . -team -payroll_lag4 -avgwin_lag5 -avgwin_lag1 -payroll_lag5 -payroll_lag3, data =
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4 - avgwin_lag5 -
##     avgwin_lag1 - payroll_lag5 - payroll_lag3, data = last_5years)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.10542 -0.01801  0.00232  0.02926  0.06465
##
## Coefficients:
##               Estimate Std. Error t value   Pr(>|t|)
## (Intercept)   0.570319   0.081966    6.96 0.00000034 ***
## payroll_lag1  0.000392   0.000274    1.43     0.1653
## payroll_lag2  0.000244   0.000362    0.67     0.5063
## avgwin_lag2   0.519764   0.141771    3.67     0.0012 **
## avgwin_lag3  -0.369298   0.176113   -2.10     0.0467 *
## avgwin_lag4  -0.419934   0.136562   -3.08     0.0052 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0437 on 24 degrees of freedom
## Multiple R-squared:  0.549,  Adjusted R-squared:  0.455
## F-statistic: 5.84 on 5 and 24 DF,  p-value: 0.00115
```

```r
summary(lm(avgwin ~ . -team -payroll_lag4 -avgwin_lag5 -avgwin_lag1 -payroll_lag5 -payroll_lag3 -payrol
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4 - avgwin_lag5 -
##     avgwin_lag1 - payroll_lag5 - payroll_lag3 - payroll_lag2,
##     data = last_5years)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.11094 -0.01508  0.00388  0.02677  0.07620
##
## Coefficients:
##               Estimate Std. Error t value   Pr(>|t|)
## (Intercept)   0.565351   0.080740    7.00 0.00000024 ***
## payroll_lag1  0.000499   0.000221    2.25     0.0333 *
## avgwin_lag2   0.486773   0.131613    3.70     0.0011 **
## avgwin_lag3  -0.324439   0.161292   -2.01     0.0552 .
## avgwin_lag4  -0.396055   0.130451   -3.04     0.0055 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0433 on 25 degrees of freedom
## Multiple R-squared:  0.54,   Adjusted R-squared:  0.467
## F-statistic: 7.35 on 4 and 25 DF,  p-value: 0.000467
```

```r
summary(lm(avgwin ~ . -team -payroll_lag4 -avgwin_lag5 -avgwin_lag1 -payroll_lag5 -payroll_lag3 -payrol
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4 - avgwin_lag5 -
##     avgwin_lag1 - payroll_lag5 - payroll_lag3 - payroll_lag2 -
##     avgwin_lag3, data = last_5years)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.14646 -0.01807  0.00922  0.02539  0.07564
##
## Coefficients:
##              Estimate Std. Error t value   Pr(>|t|)
## (Intercept)   0.508958   0.080029    6.36 0.00000098 ***
## payroll_lag1  0.000305   0.000211    1.45     0.1598
## avgwin_lag2   0.383337   0.128052    2.99     0.0060 **
## avgwin_lag4  -0.464237   0.133145   -3.49     0.0018 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0457 on 26 degrees of freedom
## Multiple R-squared:  0.466,  Adjusted R-squared:  0.404
## F-statistic: 7.56 on 3 and 26 DF,  p-value: 0.000853
```

```
summary(lm(avgwin ~ . -team -payroll_lag4 -avgwin_lag5 -avgwin_lag1 -payroll_lag5 -payroll_lag3 -payroll
```

```
##
## Call:
## lm(formula = avgwin ~ . - team - payroll_lag4 - avgwin_lag5 -
##     avgwin_lag1 - payroll_lag5 - payroll_lag3 - payroll_lag2 -
##     avgwin_lag3 - payroll_lag1, data = last_5years)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.14569 -0.01120  0.00467  0.02812  0.07996
##
## Coefficients:
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)   0.4791     0.0789    6.07 0.0000017 ***
## avgwin_lag2   0.4543     0.1207    3.77   0.00082 ***
## avgwin_lag4  -0.4126     0.1308   -3.15   0.00393 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0466 on 27 degrees of freedom
## Multiple R-squared:  0.423,  Adjusted R-squared:  0.38
## F-statistic:  9.9 on 2 and 27 DF,  p-value: 0.000597
```

Using this process, we would select a model with the two explanatory variables of the average winning percentage from 2 years and 4 years ago. This model seems rather arbitary because there is not something significant about 2 years or 4 years ago.

Perhaps a better solutions is to build features from the previous years. Below we attempt to using simple exponential smoothing ($\alpha = 0.6$ to weight recent values more) of payroll and winning percentage over 3 years.

```
alpha <- 0.6
nyears <- 3

fn_ses_forecast <- function(x) {
```

```r
  if (sum(!is.na(x)) < nyears) {
    fore <- as.double(NA)
  } else {
    fore <- data.frame(ses(x, alpha = alpha, initial = 'simple'))[,1][1]
  }
  return(fore)
}

last_2years <-
    ml_pay2 %>%
        filter(metric %in% c("payroll","avgwin")) %>%
        select(-payroll, -avgwin) %>%
        arrange(team, year) %>%
        spread(metric, value) %>%
        group_by(team) %>%
        mutate(
            payroll_ses = rollapply(payroll, FUN = fn_ses_forecast,
                            width = list(-nyears:-1), fill = NA, by.column = TRUE, align = "right")
            , avgwin_ses = rollapply(avgwin, FUN = fn_ses_forecast,
                            width = list(-nyears:-1), fill = NA, by.column = TRUE, align = "right")
        ) %>%
        ungroup() %>%
        group_by(team) %>%
        mutate(
            payroll_ses_lag1 = lag(payroll_ses,1)
            , avgwin_ses_lag1 = lag(avgwin_ses,1)
        ) %>%
        ungroup() %>%
        filter(year == 2014) %>%
        select(team, avgwin, payroll_ses, avgwin_ses, payroll_ses_lag1, avgwin_ses_lag1)

last_2years
```

```
## # A tibble: 30 × 6
##                    team avgwin payroll_ses avgwin_ses payroll_ses_lag1
##                   <chr>  <dbl>       <dbl>      <dbl>            <dbl>
## 1  Arizona Diamondbacks 0.3951       79.87     0.5128            67.16
## 2        Atlanta Braves 0.4877       87.78     0.5827            84.37
## 3     Baltimore Orioles 0.5926       87.79     0.5207            82.39
## 4        Boston Red Sox 0.4383      157.78     0.5504           168.69
## 5          Chicago Cubs 0.4506      103.83     0.4049           106.53
## 6     Chicago White Sox 0.4506      115.39     0.4373           106.50
## 7       Cincinnati Reds 0.4691       96.41     0.5551            79.19
## 8     Cleveland Indians 0.5247       73.36     0.5205            68.66
## 9      Colorado Rockies 0.4074       75.97     0.4410            81.44
## 10       Detroit Tigers 0.5556      137.71     0.5686           124.41
## # ... with 20 more rows, and 1 more variables: avgwin_ses_lag1 <dbl>
```

Despite our efforts above, we find that this is not very helpful. We settle on a model that contains the 3-year smoothed payroll figures.

```r
(ses_fit <- step(lm(avgwin ~ . -team, data = last_2years)))
```

```
## Start:  AIC=-167.8
## avgwin ~ (team + payroll_ses + avgwin_ses + payroll_ses_lag1 +
##     avgwin_ses_lag1) - team
##
##                    Df Sum of Sq    RSS  AIC
## - avgwin_ses_lag1   1   0.00030 0.0804 -170
## - avgwin_ses        1   0.00354 0.0837 -168
## - payroll_ses_lag1  1   0.00532 0.0855 -168
## - payroll_ses       1   0.00542 0.0855 -168
## <none>                          0.0801 -168
##
## Step:  AIC=-169.7
## avgwin ~ payroll_ses + avgwin_ses + payroll_ses_lag1
##
##                    Df Sum of Sq    RSS  AIC
## - avgwin_ses        1   0.00474 0.0852 -170
## - payroll_ses       1   0.00521 0.0856 -170
## <none>                          0.0804 -170
## - payroll_ses_lag1  1   0.00561 0.0860 -170
##
## Step:  AIC=-169.9
## avgwin ~ payroll_ses + payroll_ses_lag1
##
##                    Df Sum of Sq    RSS  AIC
## <none>                          0.0852 -170
## - payroll_ses_lag1  1    0.0134 0.0986 -168
## - payroll_ses       1    0.0166 0.1018 -167
```

```
##
## Call:
## lm(formula = avgwin ~ payroll_ses + payroll_ses_lag1, data = last_2years)
##
## Coefficients:
##      (Intercept)       payroll_ses  payroll_ses_lag1
##          0.49336           0.00124          -0.00123
```

Using this model, we can predict the winning precentage for the teams in 2015. To do this we just need to change the width of our rolling simple exponential smoothing function to include 2014 data as this was not being used in the previous prediction of 2014.

```r
last_2years_2015 <-
    ml_pay2 %>%
        filter(metric %in% c("payroll","avgwin")) %>%
        select(-payroll, -avgwin) %>%
        arrange(team, year) %>%
        spread(metric, value) %>%
        group_by(team) %>%
        mutate(
            payroll_ses = rollapply(payroll, FUN = fn_ses_forecast,
                            width = list((-nyears+1):0), fill = NA, by.column = TRUE, align = "right")
```

```r
            , avgwin_ses = rollapply(avgwin, FUN = fn_ses_forecast,
                             width = list((-nyears+1):0), fill = NA, by.column = TRUE, align = "right")
        ) %>%
        ungroup() %>%
        group_by(team) %>%
        mutate(
            payroll_ses_lag1 = lag(payroll_ses,1)
            , avgwin_ses_lag1 = lag(avgwin_ses,1)
        ) %>%
        ungroup() %>%
        filter(year == 2014) %>%
        select(team, avgwin, payroll_ses, avgwin_ses, payroll_ses_lag1, avgwin_ses_lag1)

cbind(
    last_2years_2015 %>% select(team),
    prediction_2015 = predict(ses_fit, last_2years_2015)
) %>% tbl_df %>%
    print(n = 30)
```

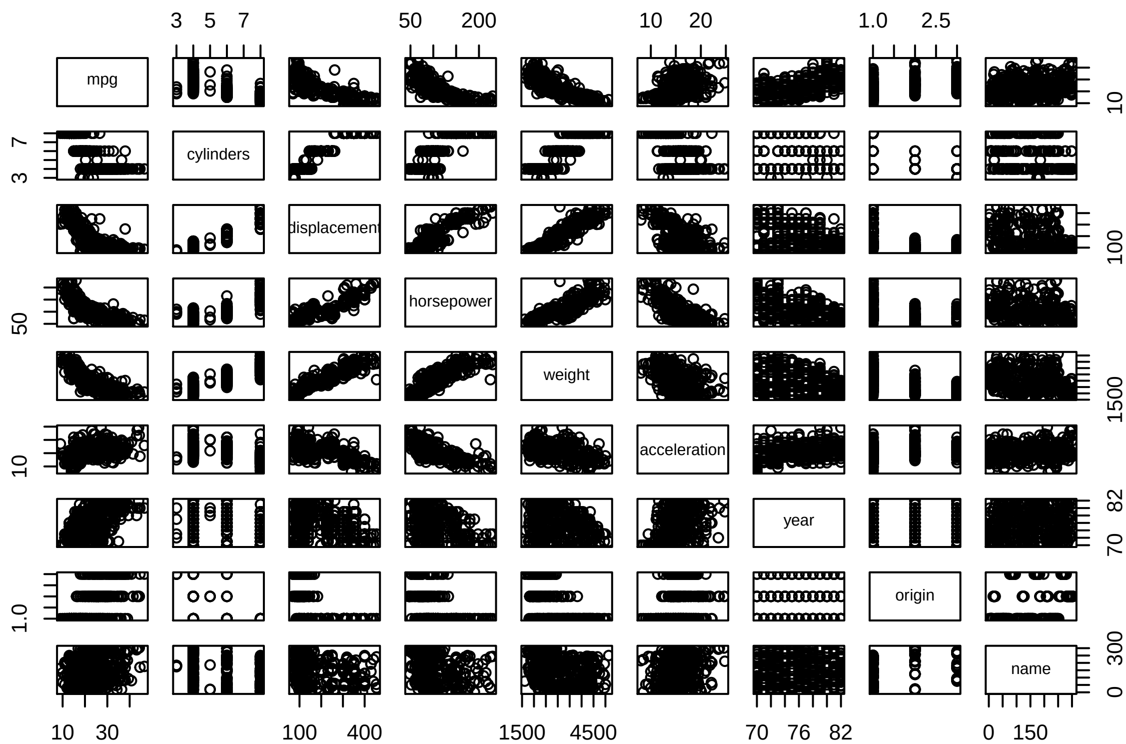
```
## # A tibble: 30 × 2
##                     team prediction_2015
## *                  <chr>           <dbl>
## 1   Arizona Diamondbacks          0.5201
## 2         Atlanta Braves          0.5110
## 3      Baltimore Orioles          0.5084
## 4         Boston Red Sox          0.4994
## 5           Chicago Cubs          0.4803
## 6      Chicago White Sox          0.4738
## 7        Cincinnati Reds          0.5066
## 8      Cleveland Indians          0.5031
## 9       Colorado Rockies          0.5080
## 10        Detroit Tigers          0.5149
## 11        Houston Astros          0.4970
## 12    Kansas City Royals          0.5128
## 13    Los Angeles Angels          0.5104
## 14   Los Angeles Dodgers          0.5432
## 15         Miami Marlins          0.4900
## 16     Milwaukee Brewers          0.5077
## 17       Minnesota Twins          0.4924
## 18         New York Mets          0.4945
## 19      New York Yankees          0.4851
## 20     Oakland Athletics          0.5102
## 21 Philadelphia Phillies          0.5033
## 22    Pittsburgh Pirates          0.5011
## 23       San Diego Padres          0.5164
## 24  San Francisco Giants          0.5115
## 25      Seattle Mariners          0.5051
## 26   St. Louis Cardinals          0.4936
## 27        Tampa Bay Rays          0.5107
## 28         Texas Rangers          0.5143
## 29      Toronto Blue Jays          0.5205
## 30  Washington Nationals          0.5228
```

5 Question 5

5.1 Exploratory Analysis

We can start of with basic pairs plot, but it's difficult to read.

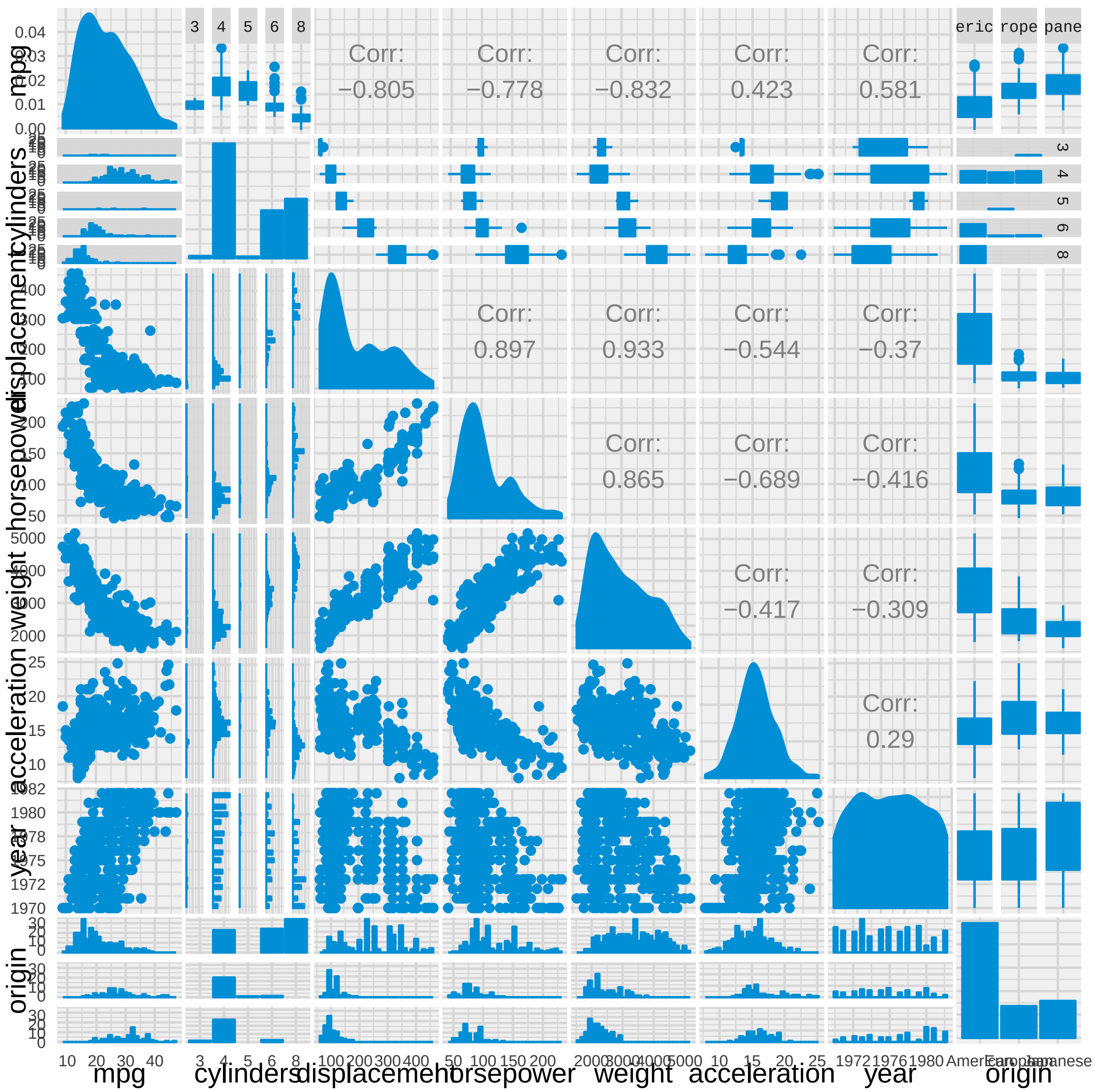
```r
pairs(Auto)
```



But, we can do much better than that using ggpairs. Here we can learn much more about our dataset.

```r
Auto_proper <-
    Auto %>% tbl_df %>%
    mutate(
          cylinders = as.factor(cylinders)
        , year = as.integer(paste0("19", year))
        , year2 = as.factor(paste0("19", year))
        , origin = factor(origin, labels = c('American', 'European', 'Japanese'))
        , name = as.character(name)
    )


ggpairs(Auto_proper %>% select(-name, -year2),
  upper = list(continuous = wrap("cor"),
               combo = wrap("box", colour = pal538['blue'], fill = pal538['blue']),
```

```
                discrete =  wrap("facetbar", colour = pal538['blue'], fill = pal538['blue']))
, lower = list(continuous = wrap("points", colour = pal538['blue'], fill = pal538['blue']),
               combo = wrap("facethist", bins = 30, colour = pal538['blue'], fill = pal538['blue']),
               discrete = wrap("facetbar", colour = pal538['blue'], fill = pal538['blue']))
, diag = list(continuous = wrap("densityDiag", colour = pal538['blue'], fill = pal538['blue']),
              discrete = wrap("barDiag", colour = pal538['blue'], fill = pal538['blue']))
) + theme_jrf()
```



From this plot alone, we can glean a lot information about the Auto dataset.

1. Cars with fewer cylinders generally have higher MPG

2. There are negative relationships between displacement, horsepower, and weight and MPG
3. In general, newer cars have better MPG
4. American made cars have much lower MPGs than European or Japanese cars
5. Most of the cars in the dataset are from America
6. Cars with 6 and 8 cylinders almost exclusively come from America
7. Each year in the range of the dataset has a nearly equal number of cars
8. Generally cars with fewer cyclinders are lighter

More points can be made but this is a strong starting point.

Before going much futher, let's check to ensure we have no missing data.

```r
# Complete.cases shows there is no missing values
sum(!complete.cases(Auto_proper))
```

```
## [1] 0
```

We can do some summary statistics to get a feel of the bounds of the variables

```r
sapply(Auto_proper, summary)
```

```
## $mpg
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     9.0    17.0    22.8    23.4    29.0    46.6
##
## $cylinders
##   3   4   5   6   8
##   4 199   3  83 103
##
## $displacement
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      68     105     151     194     276     455
##
## $horsepower
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    46.0    75.0    93.5   104.0   126.0   230.0
##
## $weight
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1610    2230    2800    2980    3610    5140
##
## $acceleration
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     8.0    13.8    15.5    15.5    17.0    24.8
##
## $year
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1970    1970    1980    1980    1980    1980
##
## $origin
## American European Japanese
##      245       68       79
##
```

```
## $name
##    Length     Class      Mode 
##       392 character character 
## 
## $year2
## 191970 191971 191972 191973 191974 191975 191976 191977 191978 191979 
##     29     27     28     40     26     30     34     28     36     29 
## 191980 191981 191982 
##     27     28     30
```

5.2 Year

5.2.1 MPG vs Year

```r
auto_fit1 <- lm(mpg ~ year, data = Auto_proper)
summary(auto_fit1)
```

```
## 
## Call:
## lm(formula = mpg ~ year, data = Auto_proper)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -12.021  -5.441  -0.441   4.974  18.209 
## 
## Coefficients:
##              Estimate Std. Error t value            Pr(>|t|)    
## (Intercept) -2407.0791   172.6169   -13.9 <0.0000000000000002 ***
## year            1.2300     0.0874    14.1 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 6.36 on 390 degrees of freedom
## Multiple R-squared:  0.337,  Adjusted R-squared:  0.335 
## F-statistic:  198 on 1 and 390 DF,  p-value: <0.0000000000000002
```

We find that model year is a significant variable at the 0.05 level. We have strong evidence against the hypothesis that the coefficient associated with year is equal to 0 (*P-value = 1.076e-36*).

We estimate that for each additional year (car being newer) a cars MPG increases by **1.23**. For example, for a car with model year 1980 we estimate 28.3912 mpg and a car with model year 1981 we estimate 29.6212. The difference a year makes in the estimate is $29.6212 - 28.3912 = 0$.

5.2.2 Add Horsepower

```r
auto_fit2 <- lm(mpg ~ horsepower + year, data = Auto_proper)
summary(auto_fit2)
```

```
## 
```

```
## Call:
## lm(formula = mpg ~ horsepower + year, data = Auto_proper)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.077  -3.078  -0.431   2.588  15.315
##
## Coefficients:
##                Estimate  Std. Error t value            Pr(>|t|)
## (Intercept) -1261.54806   131.20940   -9.61 <0.0000000000000002 ***
## horsepower     -0.13165     0.00634  -20.76 <0.0000000000000002 ***
## year            0.65727     0.06626    9.92 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.39 on 389 degrees of freedom
## Multiple R-squared:  0.685,  Adjusted R-squared:  0.684
## F-statistic:  424 on 2 and 389 DF,  p-value: <0.0000000000000002
```

Year is still significant at the 0.05 level. We have strong evidence against the hypothesis that the coefficient associated with year is equal to 0 (*P-value = 7.994e-21*).

For cars with the same horsepower, we estimate that each additional year (car being newer) a cars MPG increases by **0.6573**.

We show the two confidence intervals for the coefficient of year between the two models.

```r
confint(auto_fit1, "year", level = 0.95)
```

```
##      2.5 % 97.5 %
## year 1.058  1.402
```

```r
confint(auto_fit2, "year", level = 0.95)
```

```
##      2.5 % 97.5 %
## year 0.527 0.7875
```

These two confidence intervals are different. To a non-statistician, we would describe this difference as

> In our first model to predict MPG, we only use the model year of the car. In our second model, we include horsepower which explains part of the variation in MPG between cars. In other words, the effect of the model year on MPG is smaller when we include the variation explained by horsepower.

5.2.3 Interaction Term

```r
auto_fit3 <- lm(mpg ~ horsepower * year, data = Auto_proper)
summary(auto_fit3)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower * year, data = Auto_proper)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.349  -2.451  -0.456   2.406  14.444
##
## Coefficients:
##                    Estimate  Std. Error t value            Pr(>|t|)
## (Intercept)     -4291.36393   318.66569   -13.5 <0.0000000000000002 ***
## horsepower         31.36685     3.08307    10.2 <0.0000000000000002 ***
## year                2.19198     0.16135    13.6 <0.0000000000000002 ***
## horsepower:year    -0.01596     0.00156   -10.2 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.9 on 388 degrees of freedom
## Multiple R-squared:  0.752,  Adjusted R-squared:  0.75
## F-statistic:  393 on 3 and 388 DF,  p-value: <0.0000000000000002
```

The interaction term is significant at the 0.05 level. We have strong evidence against the hypothesis that the coefficient associated with the interaction of year and horespower is equal to 0 ($P\text{-}value = 7.367e\text{-}22$)

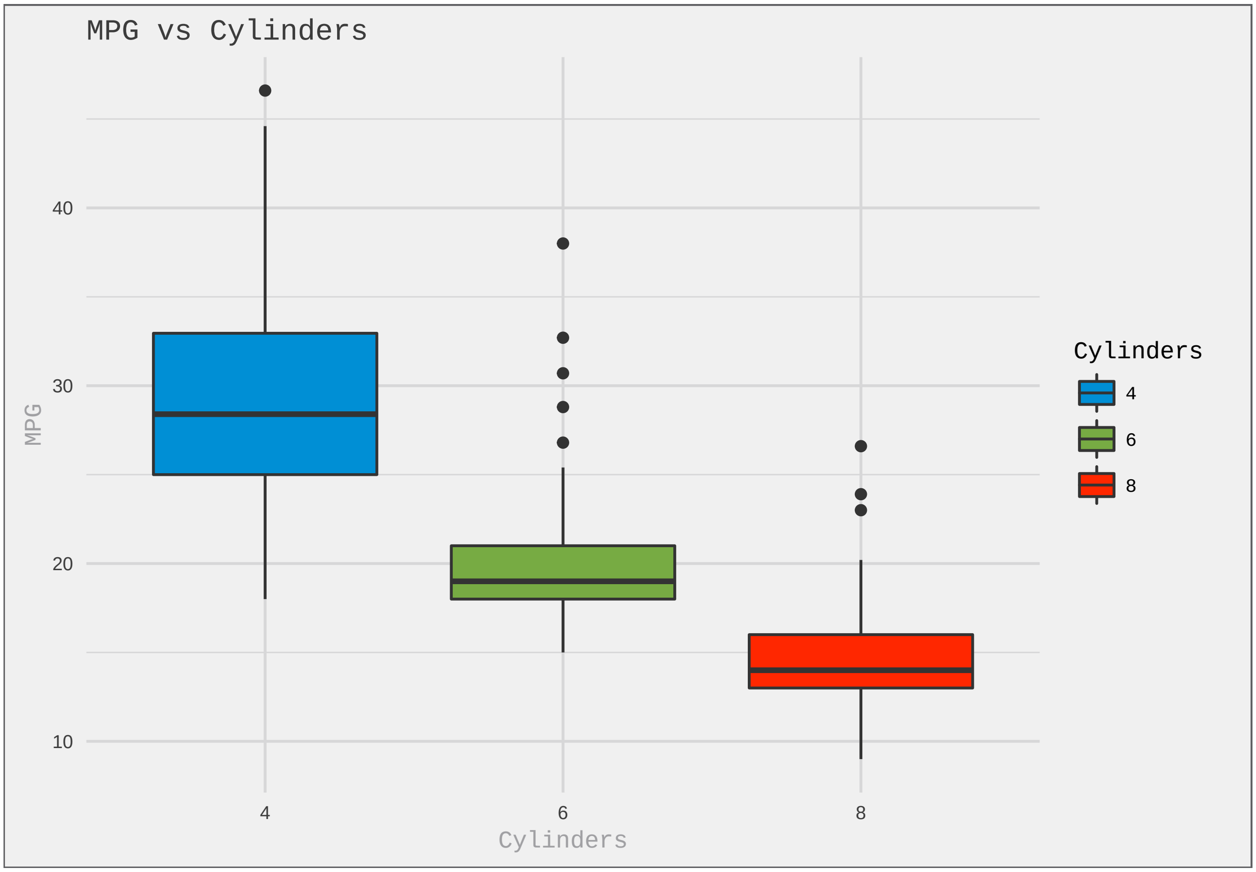
Now the effect of year cannot be interpreted uniformly when holding the other variable horsepower constant as it depends on the value of horsepower. Thus, we show the effect of a 1 year increase in model year (one year newer), on the 25th percentile, median, and 75th percentile horsepower values in our dataset.

	Horsepower = 75	Horsepower = 93.5	Horsepower = 126
Effect of 1 year increase in model year	0.9951	0.6999	0.1812

5.3 Cylinder

We have the cylinder variable coded a categorical variable because the number of cylinders is a characteristic of the car, rather than a feature that can be easily changed. In other words, a 1 unit change in cyclinder is really not meaningful as most engines are made with cylinders with multiples of 2.

```
Auto_proper %>%
    filter(!(cylinders %in% c("3","5"))) %>%
    ggplot(aes(x = cylinders, y = mpg, fill = cylinders)) +
    scale_fill_manual("Cylinders", values = c('4' = pal538['blue'][[1]], '6' = pal538['green'][[1]], '8
    geom_boxplot() +
    theme_jrf() +
    labs(title = "MPG vs Cylinders", x = "Cylinders", y  = "MPG")
```



5.3.1 As Quantitative Variable

Per the question, we will use cylinders as a integer (not continuous).

```
auto_fit4 <- lm(mpg ~ horsepower + as.integer(cylinders), data = Auto_proper)
summary(auto_fit4)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower + as.integer(cylinders), data = Auto_proper)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.392  -2.965  -0.318   2.149  16.634
##
## Coefficients:
##                       Estimate Std. Error t value            Pr(>|t|)
## (Intercept)           40.72614    0.65869   61.83 <0.0000000000000002 ***
## horsepower            -0.08617    0.00985   -8.75 <0.0000000000000002 ***
## as.integer(cylinders) -2.57965    0.28486   -9.06 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.46 on 389 degrees of freedom
## Multiple R-squared:  0.675,  Adjusted R-squared:  0.673
```

```
## F-statistic:  403 on 2 and 389 DF,  p-value: <0.0000000000000002
```

Cylinders is significant at the 0.01 level. We have strong evidence against the hypothesis that the coefficient associated with cylinders is equal to 0 (*P-value = 6.634e-18*).

We estimate that, holding horsepower constant, for each additional cylinder in the car, the car's mpg is -2.5796 lower.

5.3.2 As Categorical Variable

```r
auto_fit5 <- lm(mpg ~ horsepower + cylinders, data = Auto_proper)
summary(auto_fit5)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower + cylinders, data = Auto_proper)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -9.59  -2.71  -0.61   1.90  16.33
##
## Coefficients:
##             Estimate Std. Error t value            Pr(>|t|)
## (Intercept)  30.7761     2.4128   12.76 <0.0000000000000002 ***
## horsepower   -0.1030     0.0113   -9.09 <0.0000000000000002 ***
## cylinders4    6.5734     2.1692    3.03              0.0026 **
## cylinders5    5.0737     3.2666    1.55              0.1212
## cylinders6   -0.3441     2.1858   -0.16              0.8750
## cylinders8    0.4974     2.2764    0.22              0.8272
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.27 on 386 degrees of freedom
## Multiple R-squared:  0.705,  Adjusted R-squared:  0.701
## F-statistic:  184 on 5 and 386 DF,  p-value: <0.0000000000000002
```

Cylinders is significant at the 0.01 level. If one of the coefficients of the factor levels in the model is significant, then the variable as whole is significant. We then use ANOVA to compare the two models.

```r
anova(auto_fit4, auto_fit5)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ horsepower + as.integer(cylinders)
## Model 2: mpg ~ horsepower + cylinders
##   Res.Df  RSS Df Sum of Sq    F      Pr(>F)
## 1    389 7752
## 2    386 7037  3       715 13.1 0.000000038 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

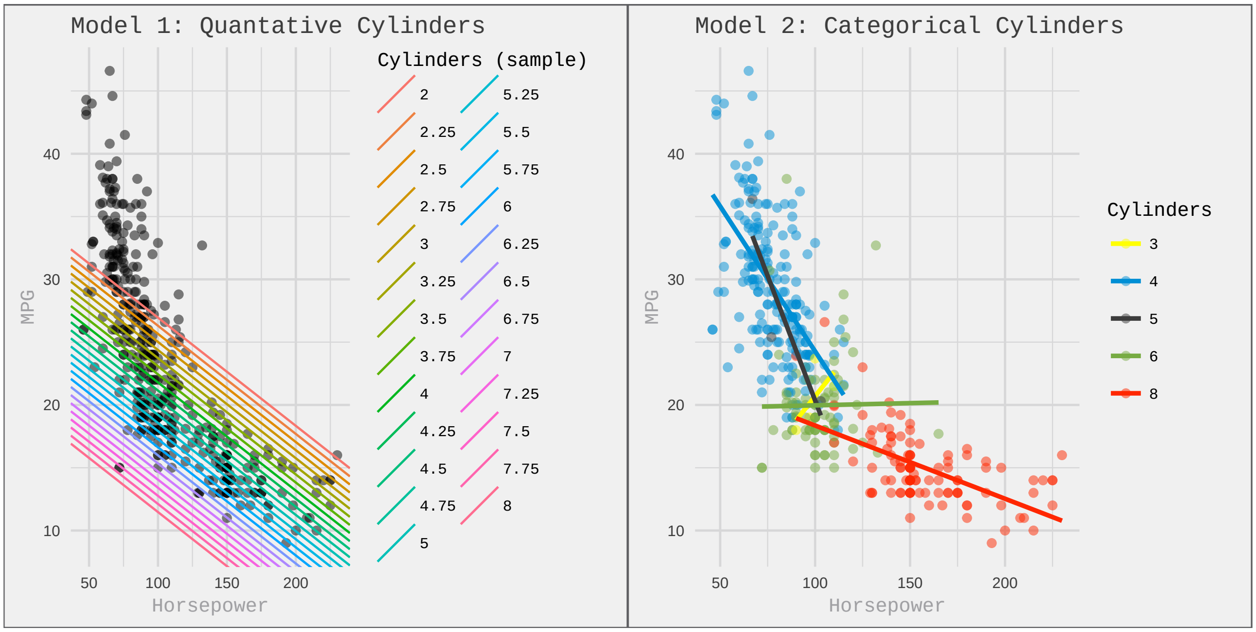
We have strong evidence that the model using categorical variable for cylinder better explains the variation in MPG than the model that uses a quantative variable for cylinder (*P-value = 3.782e-08*)

5.3.3 Difference

The fundemental difference between model 1 and model 2 is that model 1 assumes that there can be incremental increaes in cylinders whereas model 2 assumes that there are different types of cylinders. In reality, you cannot increase a cars cylinders by 0.25 so model 1 is not practically valid. Model 2 recognizes the nature of the variable cylinder and how cars are made, thus being a practically applicable model.

The plots below provide a comparison between the two models.

```
cylinders <- seq(from = 2, to = 8, by = 0.25)
int_line <- data_frame(
                  cylinders = cylinders
                , int = coef(summary(auto_fit4))[,1][[1]] + coef(summary(auto_fit4))[,1][[3]]*cylinders
                , slope = coef(summary(auto_fit4))[,1][[2]]
                )

g1 <-
    ggplot(Auto_proper, aes(x = horsepower, y = mpg)) +
    geom_point(alpha = 0.5) +
    geom_abline(data = int_line, aes(intercept = int, slope = slope, colour = as.factor(cylinders))) +
    theme_jrf() +
    labs(title = "Model 1: Quantative Cylinders", x = "Horsepower", y  = "MPG") +
    guides(colour = guide_legend(title = "Cylinders (sample)"))

g2 <-
    ggplot(Auto_proper, aes(x = horsepower, y = mpg, colour = cylinders)) +
    geom_point(alpha = 0.5) +
    geom_smooth(method = "lm", se = FALSE) +
    theme_jrf() +
    labs(title = "Model 2: Categorical Cylinders", x = "Horsepower", y  = "MPG") +
    scale_colour_manual("Cylinders", values = c('3' = "#ffff00",
                                                 '4' = pal538['blue'][[1]],
                                                 '5' = pal538['dkgray'][[1]],
                                                 '6' = pal538['green'][[1]],
                                                 '8' = pal538['red'][[1]]))


grid.arrange(g1, g2, ncol = 2)
```



5.4 Final Model

First we make a dataframe of the car that we will predict MPG.

```
future_car <- data_frame(
       year = 1983
     , length = 180
     , cylinders = factor(8, levels = c(3,4,5,6,8))
     , displacement = 350
     , horsepower = 260
     , weight = 4000
     )
```

Reviewing the diagonal from the pairs plot in the exploratory analysis, we note that the continuous predictors and the dependent variable MPG are all somewhat normally distribute and we decide not to perform any transformations.

We will use the `leaps` package using each of the 3 methods. With each we will:

1. Show the feature combinations for each value of d (number of predictors)
2. Plot Mallow's Cp, BIC, and Adjusted R^2.

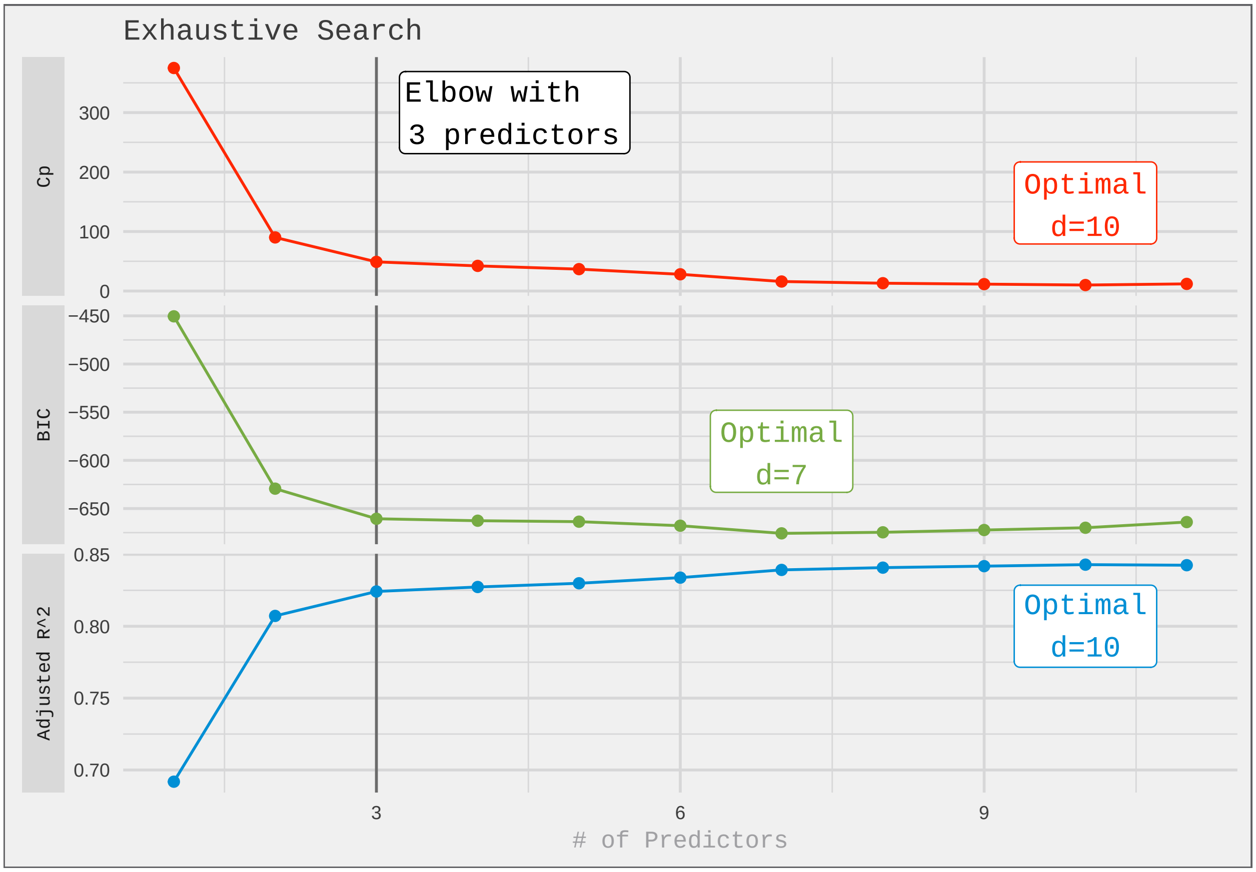
5.4.1 Exhaustive

```
auto_fit6 <- regsubsets(mpg ~ ., data = Auto_proper %>% select(-name, -year2), nvmax = 11, method="exhau
auto_fit6_sum <- summary(auto_fit6)
as_data_frame(auto_fit6_sum$outmat) %>% print(width = Inf)
```

```
## # A tibble: 11 × 11
##    cylinders4 cylinders5 cylinders6 cylinders8 displacement horsepower
```

```
##         <chr>      <chr>      <chr>      <chr>        <chr>      <chr>
## 1
## 2
## 3                              *
## 4           *                                *
## 5           *                                *
## 6           *                                            *
## 7           *                                            *          *
## 8           *          *                                 *          *
## 9           *          *                     *           *          *
## 10          *          *          *          *           *          *
## 11          *          *          *          *           *          *
##    weight acceleration  year originEuropean originJapanese
##     <chr>        <chr> <chr>          <chr>          <chr>
## 1       *
## 2       *                  *
## 3       *                  *
## 4       *                  *
## 5       *                  *                             *
## 6       *                  *              *              *
## 7       *                  *              *              *
## 8       *                  *              *              *
## 9       *                  *              *              *
## 10      *                  *              *              *
## 11      *            *     *              *              *
```

```r
data_frame(
      predictors = 1:length(auto_fit6_sum$cp)
    , cp = auto_fit6_sum$cp
    , bic = auto_fit6_sum$bic
    , adjr2 = auto_fit6_sum$adjr2
) %>%
    gather(metric, value, -predictors) %>%
    mutate(metric = factor(metric, levels = c("cp","bic","adjr2"))) %>%
    ggplot(aes(x = predictors, y = value, colour = metric)) +
    facet_grid(metric ~ ., scale = "free_y", switch = "y",
               labeller = ggplot2::labeller(metric = c(cp = "Cp", bic = "BIC", adjr2 = "Adjusted R^2")))
    geom_vline(xintercept = 3, alpha = 0.5) + geom_line() + geom_point() +
    geom_label(data = data_frame(
        predictors = c(which.min(auto_fit6_sum$cp), which.min(auto_fit6_sum$bic), which.max(auto_fit6_su
        , metric = factor(c("cp","bic","adjr2"), levels = c("cp","bic","adjr2"))
        , value = c(min(auto_fit6_sum$cp), min(auto_fit6_sum$bic), max(auto_fit6_sum$adjr2))
        , label = paste0("Optimal\nd=", c(which.min(auto_fit6_sum$cp), which.min(auto_fit6_sum$bic), whi
        , vjust = c(-.5, -.5, 1.25)
    ), aes(x = predictors, y = value, label = label, vjust = vjust), family = "DecimaMonoPro") +
    theme_jrf() +
    labs(title = "Exhaustive Search", x = "# of Predictors", y = NULL) +
    geom_label(data = data_frame(x = 3, y = 300, metric = factor(c("cp"), levels = c("cp","bic","adjr2")
                label = "Elbow with\n3 predictors"), aes(x=x,y=y,label=label), colour = "black", hjust =
               family = "DecimaMonoPro") +
    scale_colour_manual(guide = FALSE, values = c(pal538['red'][[1]], pal538['green'][[1]], pal538['blue
```



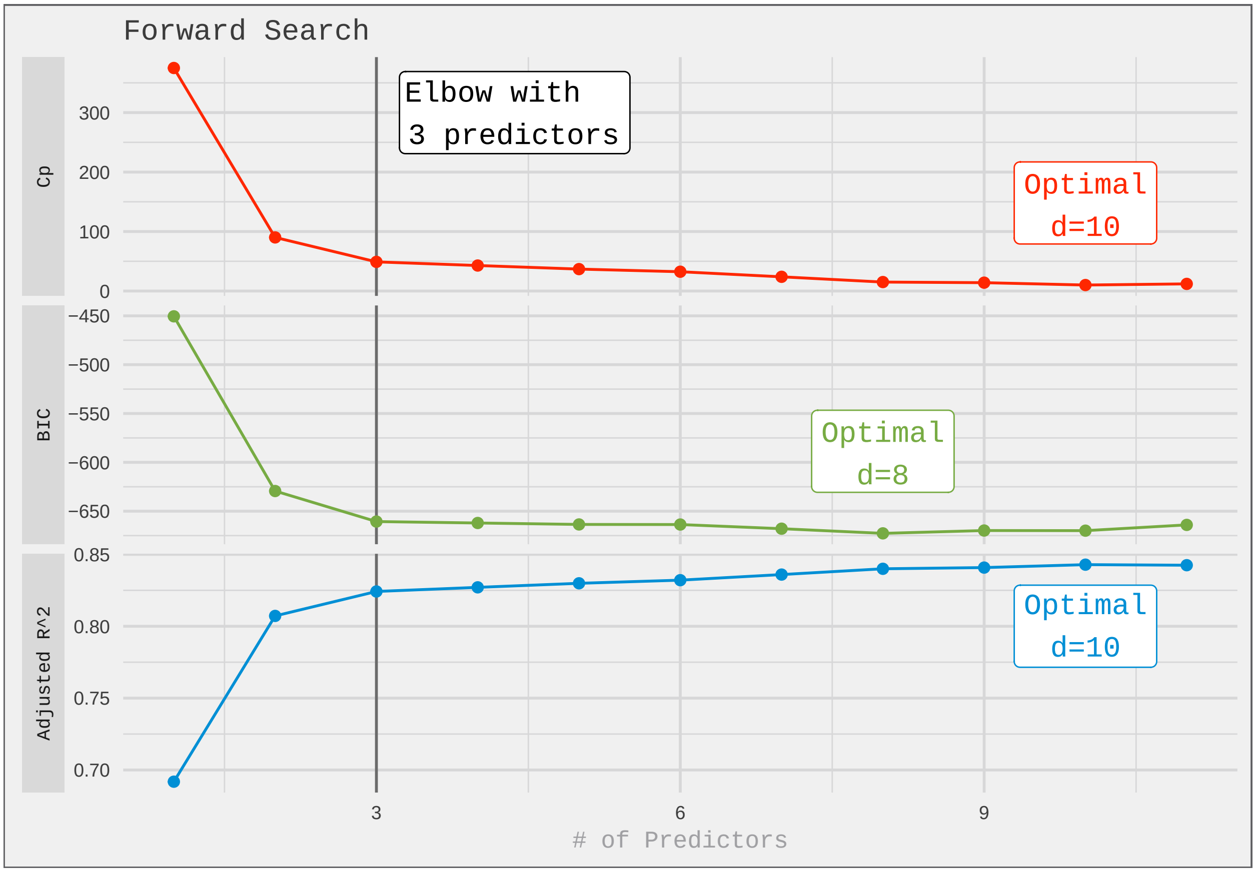
5.4.2 Forward

```
auto_fit7 <- regsubsets(mpg ~ ., data = Auto_proper %>% select(-name, -year2), nvmax = 11, method="forwa
auto_fit7_sum <- summary(auto_fit7)
as_data_frame(auto_fit7_sum$outmat) %>% print(width = Inf)
```

```
## # A tibble: 11 × 11
##    cylinders4 cylinders5 cylinders6 cylinders8 displacement horsepower
##         <chr>      <chr>      <chr>      <chr>        <chr>      <chr>
## 1
## 2
## 3                                 *
## 4                                 *                                  *
## 5                                 *                                  *
## 6                                 *                                  *
## 7                                 *                       *          *
## 8           *                     *                       *          *
## 9           *          *          *                       *          *
## 10          *          *          *          *            *          *
## 11          *          *          *          *            *          *
##    weight acceleration  year originEuropean originJapanese
##     <chr>        <chr> <chr>          <chr>          <chr>
## 1       *
## 2       *                  *
```

```
## 3        *                 *
## 4        *                 *
## 5        *                 *                           *
## 6        *                 *             *             *
## 7        *                 *             *             *
## 8        *                 *             *             *
## 9        *                 *             *             *
## 10       *                 *             *             *
## 11       *           *     *             *             *
```

```
data_frame(
      predictors = 1:length(auto_fit7_sum$cp)
    , cp = auto_fit7_sum$cp
    , bic = auto_fit7_sum$bic
    , adjr2 = auto_fit7_sum$adjr2
) %>%
    gather(metric, value, -predictors) %>%
    mutate(metric = factor(metric, levels = c("cp","bic","adjr2"))) %>%
    ggplot(aes(x = predictors, y = value, colour = metric)) +
    facet_grid(metric ~ ., scale = "free_y", switch = "y",
               labeller = ggplot2::labeller(metric = c(cp = "Cp", bic = "BIC", adjr2 = "Adjusted R^2")))
    geom_vline(xintercept = 3, alpha = 0.5) + geom_line() + geom_point() +
    geom_label(data = data_frame(
        predictors = c(which.min(auto_fit7_sum$cp), which.min(auto_fit7_sum$bic), which.max(auto_fit7_su
        , metric = factor(c("cp","bic","adjr2"), levels = c("cp","bic","adjr2"))
        , value = c(min(auto_fit7_sum$cp), min(auto_fit7_sum$bic), max(auto_fit7_sum$adjr2))
        , label = paste0("Optimal\nd=", c(which.min(auto_fit7_sum$cp), which.min(auto_fit7_sum$bic) ,whi
        , vjust = c(-.5, -.5, 1.25)
    ), aes(x = predictors, y = value, label = label, vjust = vjust), family = "DecimaMonoPro") +
    theme_jrf() +
    labs(title = "Forward Search", x = "# of Predictors", y = NULL) +
    geom_label(data = data_frame(x = 3, y = 300, metric = factor(c("cp"), levels = c("cp","bic","adjr2")
                label = "Elbow with\n3 predictors"), aes(x=x,y=y,label=label), colour = "black", hjust =
               family = "DecimaMonoPro") +
    scale_colour_manual(guide = FALSE, values = c(pal538['red'][[1]], pal538['green'][[1]], pal538['blu
```



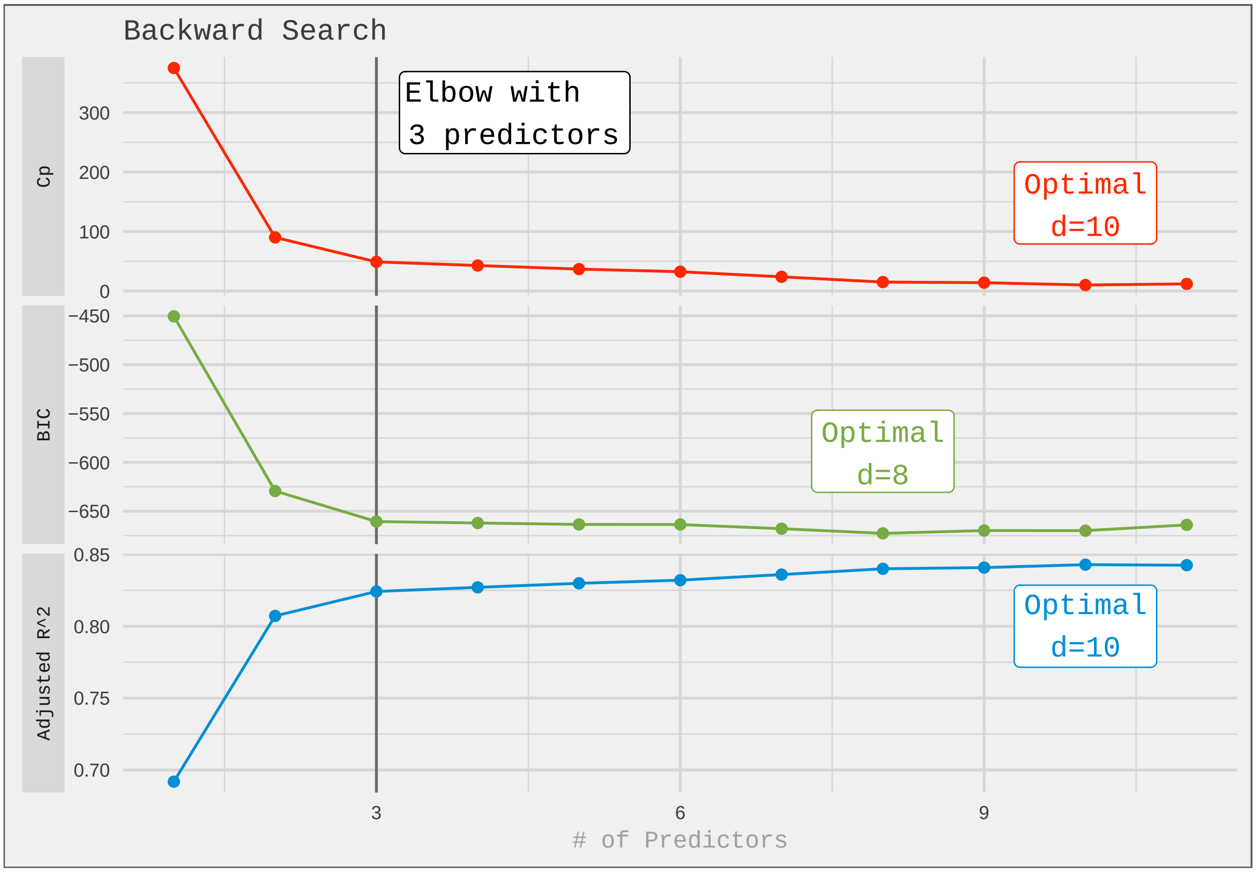
5.4.3 Backward

```
auto_fit8 <- regsubsets(mpg ~ ., data = Auto_proper %>% select(-name, -year2), nvmax = 11, method="backw
auto_fit8_sum <- summary(auto_fit7)
as_data_frame(auto_fit8_sum$outmat) %>% print(width = Inf)
```

```
## # A tibble: 11 × 11
##    cylinders4 cylinders5 cylinders6 cylinders8 displacement horsepower
##         <chr>      <chr>      <chr>      <chr>        <chr>      <chr>
## 1
## 2
## 3                                 *
## 4                                 *                                  *
## 5                                 *                                  *
## 6                                 *                                  *
## 7                                 *                       *          *
## 8           *                     *                       *          *
## 9           *          *          *                       *          *
## 10          *          *          *          *            *          *
## 11          *          *          *          *            *          *
##    weight acceleration  year originEuropean originJapanese
##     <chr>        <chr> <chr>          <chr>          <chr>
## 1       *
## 2       *                  *
```

```
## 3       *                 *
## 4       *                 *
## 5       *                 *                           *
## 6       *                 *             *             *
## 7       *                 *             *             *
## 8       *                 *             *             *
## 9       *                 *             *             *
## 10      *                 *             *             *
## 11      *           *     *             *             *
```

```
data_frame(
      predictors = 1:length(auto_fit8_sum$cp)
    , cp = auto_fit8_sum$cp
    , bic = auto_fit8_sum$bic
    , adjr2 = auto_fit8_sum$adjr2
) %>%
    gather(metric, value, -predictors) %>%
    mutate(metric = factor(metric, levels = c("cp","bic","adjr2"))) %>%
    ggplot(aes(x = predictors, y = value, colour = metric)) +
    facet_grid(metric ~ ., scale = "free_y", switch = "y",
               labeller = ggplot2::labeller(metric = c(cp = "Cp", bic = "BIC", adjr2 = "Adjusted R^2")))
    geom_vline(xintercept = 3, alpha = 0.5) + geom_line() + geom_point() +
    geom_label(data = data_frame(
        predictors = c(which.min(auto_fit8_sum$cp), which.min(auto_fit8_sum$bic), which.max(auto_fit8_su
        , metric = factor(c("cp","bic","adjr2"), levels = c("cp","bic","adjr2"))
        , value = c(min(auto_fit8_sum$cp), min(auto_fit8_sum$bic), max(auto_fit8_sum$adjr2))
        , label = paste0("Optimal\nd=", c(which.min(auto_fit8_sum$cp), which.min(auto_fit8_sum$bic), whi
        , vjust = c(-.5, -.5, 1.25)
    ), aes(x = predictors, y = value, label = label, vjust = vjust), family = "DecimaMonoPro") +
    theme_jrf() +
    labs(title = "Backward Search", x = "# of Predictors", y = NULL) +
    geom_label(data = data_frame(x = 3, y = 300, metric = factor(c("cp"), levels = c("cp","bic","adjr2")
                label = "Elbow with\n3 predictors"), aes(x=x,y=y,label=label), colour = "black", hjust =
               family = "DecimaMonoPro") +
    scale_colour_manual(guide = FALSE, values = c(pal538['red'][[1]], pal538['green'][[1]], pal538['blu
```



5.4.4 Selection

In all 3 methods, we find that there is an elbow in the information criteria at 3 predictors. These three predictors are

1. Weight
2. Year
3. 6 Cylinder Level of Cylinders

Regarding (3), this indicates that we might want to try creating a binary variable, whether or not the car is 6 cylinders. We will create 4 models

1. Model 1: Cylinders - all levels
2. Model 2: Binary 6-cylinder
3. Model 3: Binary 6-cylinder & Horsepower
4. Model 4: Cylinders - all levels & Horsepower

Model 1: Cylinders - all levels

```
Auto_proper2 <-
    Auto_proper %>%
    mutate(
        is_6cylinder = cylinders == 6
    )
```

```r
auto_fit9 <- lm(mpg ~ weight + year + cylinders, Auto_proper2)
summary(auto_fit9)
```

```
##
## Call:
## lm(formula = mpg ~ weight + year + cylinders, data = Auto_proper2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.618 -2.047 -0.129  1.772 13.882
##
## Coefficients:
##                Estimate   Std. Error t value             Pr(>|t|)
## (Intercept) -1448.909862    93.152615  -15.55 < 0.0000000000000002 ***
## weight         -0.006165     0.000438  -14.06 < 0.0000000000000002 ***
## year            0.751328     0.047148   15.94 < 0.0000000000000002 ***
## cylinders4      7.008483     1.619347    4.33             0.000019 ***
## cylinders5      8.532666     2.470665    3.45              0.00061 ***
## cylinders6      4.038757     1.676862    2.41              0.01649 *
## cylinders8      6.194379     1.798940    3.44              0.00064 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.2 on 385 degrees of freedom
## Multiple R-squared:  0.834,  Adjusted R-squared:  0.832
## F-statistic:  323 on 6 and 385 DF,  p-value: <0.0000000000000002
```

Model 2: Binary 6-cylinder

```r
auto_fit10 <- lm(mpg ~ weight + year + is_6cylinder, Auto_proper2)
summary(auto_fit10)
```

```
##
## Call:
## lm(formula = mpg ~ weight + year + is_6cylinder, data = Auto_proper2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.196 -2.005 -0.116  1.824 13.929
##
## Coefficients:
##                     Estimate   Std. Error t value             Pr(>|t|)
## (Intercept)      -1476.992411    93.612621  -15.78 < 0.0000000000000002
## weight              -0.006448     0.000207  -31.15 < 0.0000000000000002
## year                 0.769328     0.047279   16.27 < 0.0000000000000002
## is_6cylinderTRUE    -2.541301     0.408776   -6.22         0.0000000013
##
## (Intercept)      ***
## weight           ***
## year             ***
## is_6cylinderTRUE ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.27 on 388 degrees of freedom
## Multiple R-squared:  0.826,  Adjusted R-squared:  0.824
## F-statistic:  612 on 3 and 388 DF,  p-value: <0.0000000000000002
```

Model 3: Binary 6-cylinder & Horsepower

```r
auto_fit11 <- lm(mpg ~ weight + year + is_6cylinder + horsepower, Auto_proper2)
summary(auto_fit11)
```

```
##
## Call:
## lm(formula = mpg ~ weight + year + is_6cylinder + horsepower,
##     data = Auto_proper2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.942 -2.027 -0.059  1.757 13.851
##
## Coefficients:
##                      Estimate   Std. Error t value             Pr(>|t|)
## (Intercept)      -1393.172022    97.814985  -14.24 < 0.0000000000000002
## weight              -0.005474     0.000413  -13.27 < 0.0000000000000002
## year                 0.726838     0.049419   14.71 < 0.0000000000000002
## is_6cylinderTRUE    -2.916979     0.428256   -6.81       0.000000000037
## horsepower          -0.025695     0.009434   -2.72               0.0067
##
## (Intercept)      ***
## weight           ***
## year             ***
## is_6cylinderTRUE ***
## horsepower       **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.25 on 387 degrees of freedom
## Multiple R-squared:  0.829,  Adjusted R-squared:  0.827
## F-statistic:  469 on 4 and 387 DF,  p-value: <0.0000000000000002
```

Model 4: Cylinders - all levels & Horsepower

```r
auto_fit12 <- lm(mpg ~ weight + year + cylinders + horsepower, Auto_proper2)
summary(auto_fit12)
```

```
##
## Call:
## lm(formula = mpg ~ weight + year + cylinders + horsepower, data = Auto_proper2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.723 -1.996 -0.079  1.773 13.781
```

```
##
## Coefficients:
##                 Estimate   Std. Error t value            Pr(>|t|)
## (Intercept) -1392.602364   96.273881  -14.47 < 0.0000000000000002 ***
## weight         -0.005641    0.000499  -11.30 < 0.0000000000000002 ***
## year            0.723271    0.048673   14.86 < 0.0000000000000002 ***
## cylinders4      6.648429    1.620139    4.10             0.00005 ***
## cylinders5      7.896300    2.476306    3.19             0.00155 **
## cylinders6      3.678307    1.677107    2.19             0.02889 *
## cylinders8      6.521763    1.796698    3.63             0.00032 ***
## horsepower     -0.021556    0.009938   -2.17             0.03070 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.19 on 384 degrees of freedom
## Multiple R-squared:  0.836,  Adjusted R-squared:  0.833
## F-statistic:  280 on 7 and 384 DF,  p-value: <0.0000000000000002
```

Now that we have 4 models, we can compare the AIC (Mallow's Cp in linear regression) and BIC. If there is not a significant difference, we will use the simplest model (model 2).

```
data_frame(
    Model = c("1: Cylinders - all levels", "2: Binary 6-cylinder",
              "3: Binary 6-cylinder & Horsepower","4: Cylinders - all levels & Horsepower")
    , AIC = AIC(auto_fit9, auto_fit10, auto_fit11, auto_fit12)$AIC
    , BIC = BIC(auto_fit9, auto_fit10, auto_fit11, auto_fit12)$BIC
) %>%
    kable()
```

Model	AIC	BIC
1: Cylinders - all levels	2034	2066
2: Binary 6-cylinder	2048	2068
3: Binary 6-cylinder & Horsepower	2042	2066
4: Cylinders - all levels & Horsepower	2031	2067

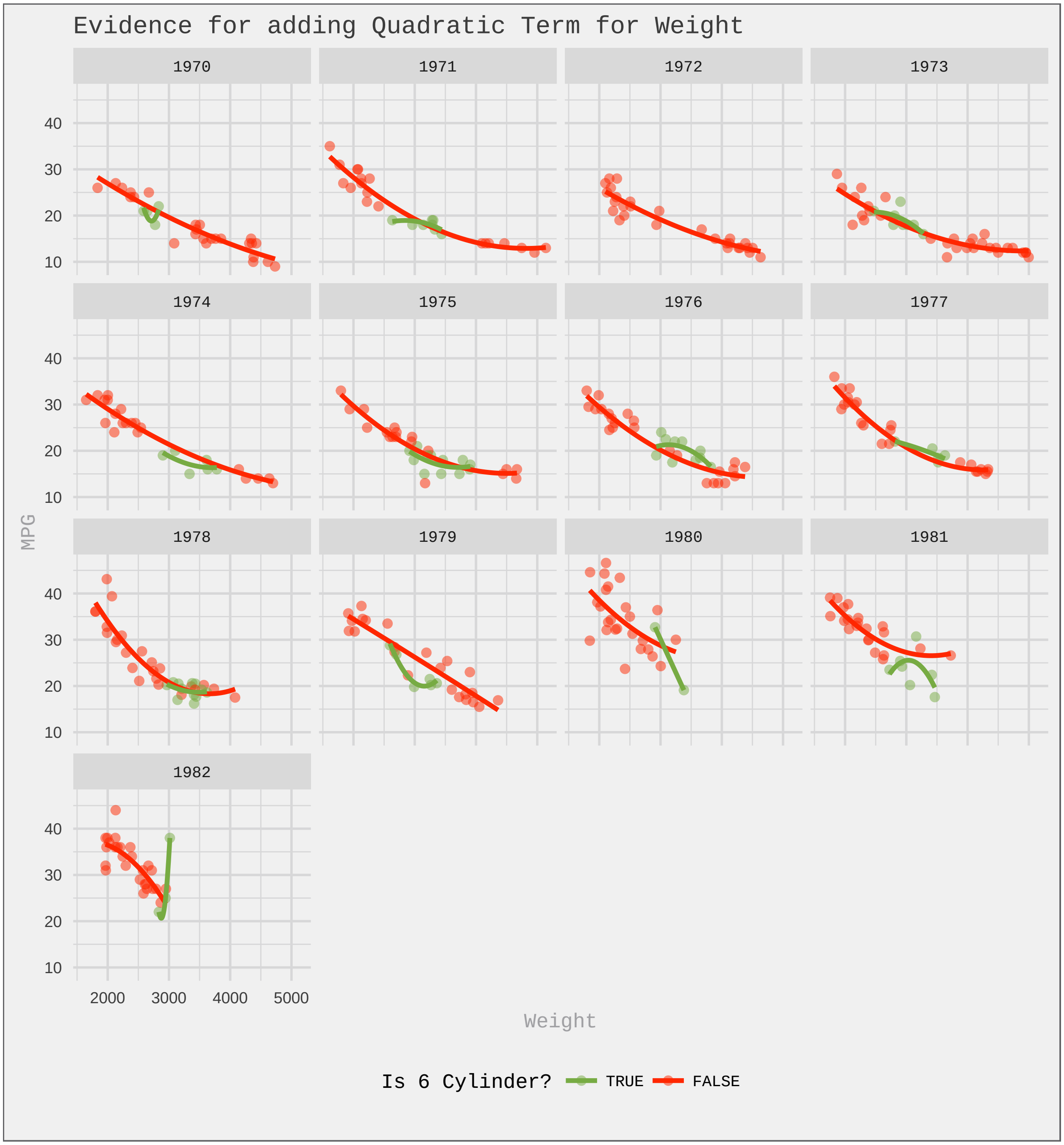
Model 2 has higher AIC and BIC values compared to the other 3 models, indicating it explains less variation in MPG. However, the difference is not large and it is the simplest model. We will use model 2 as our final model.

5.4.4.1 Quadratic Term

We notice that we might want a quadratic term for the predictor weight by looking at the following charts. We may be concerned about overfitting, particulary for 6-cylinder cars (i.e. 1970 and 1982). However, we know that we will be predicting the MPG of a 8-cylinder car.

```
Auto_proper2 %>%
    select(mpg, weight, year, cylinders, is_6cylinder) %>%
    ggplot(aes(x = weight, y = mpg, colour = is_6cylinder)) +
    facet_wrap(~ year) +
    geom_point(alpha = 0.5) +
    theme_jrf(base_size) +
```

```
    geom_smooth(method = "lm", formula = y ~ x + I(x^2), se = FALSE) +
    labs(title = "Evidence for adding Quadratic Term for Weight", x = "Weight", y = "MPG") +
    scale_colour_manual("Is 6 Cylinder?", values = c('TRUE' = pal538['green'][[1]], "FALSE" = pal538['r
    guides(colour = guide_legend(reverse = TRUE)) +
    theme(legend.position = 'bottom')
```



We create a model to add in the quadratic term for weight. We see that the binary variable for whether the car is a 6-cylinder is now only marginally significant.

```r
auto_fit13 <- lm(mpg ~ weight + I(weight^2) + year + is_6cylinder, Auto_proper2)
summary(auto_fit13)
```

```
##
## Call:
## lm(formula = mpg ~ weight + I(weight^2) + year + is_6cylinder,
##     data = Auto_proper2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.501 -1.660 -0.126  1.556 13.164
##
## Coefficients:
##                        Estimate      Std. Error t value
## (Intercept)      -1568.498491872    87.226174961  -17.98
## weight              -0.020075753     0.001671862  -12.01
## I(weight^2)          0.000002125     0.000000259    8.21
## year                 0.825674289     0.044228295   18.67
## is_6cylinderTRUE    -0.751884577     0.436195229   -1.72
##                              Pr(>|t|)
## (Intercept)      < 0.0000000000000002 ***
## weight           < 0.0000000000000002 ***
## I(weight^2)        0.0000000000000035 ***
## year             < 0.0000000000000002 ***
## is_6cylinderTRUE                0.086 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.02 on 387 degrees of freedom
## Multiple R-squared:  0.851,  Adjusted R-squared:  0.85
## F-statistic:  554 on 4 and 387 DF,  p-value: <0.0000000000000002
```

Let's remove the binary predictor.

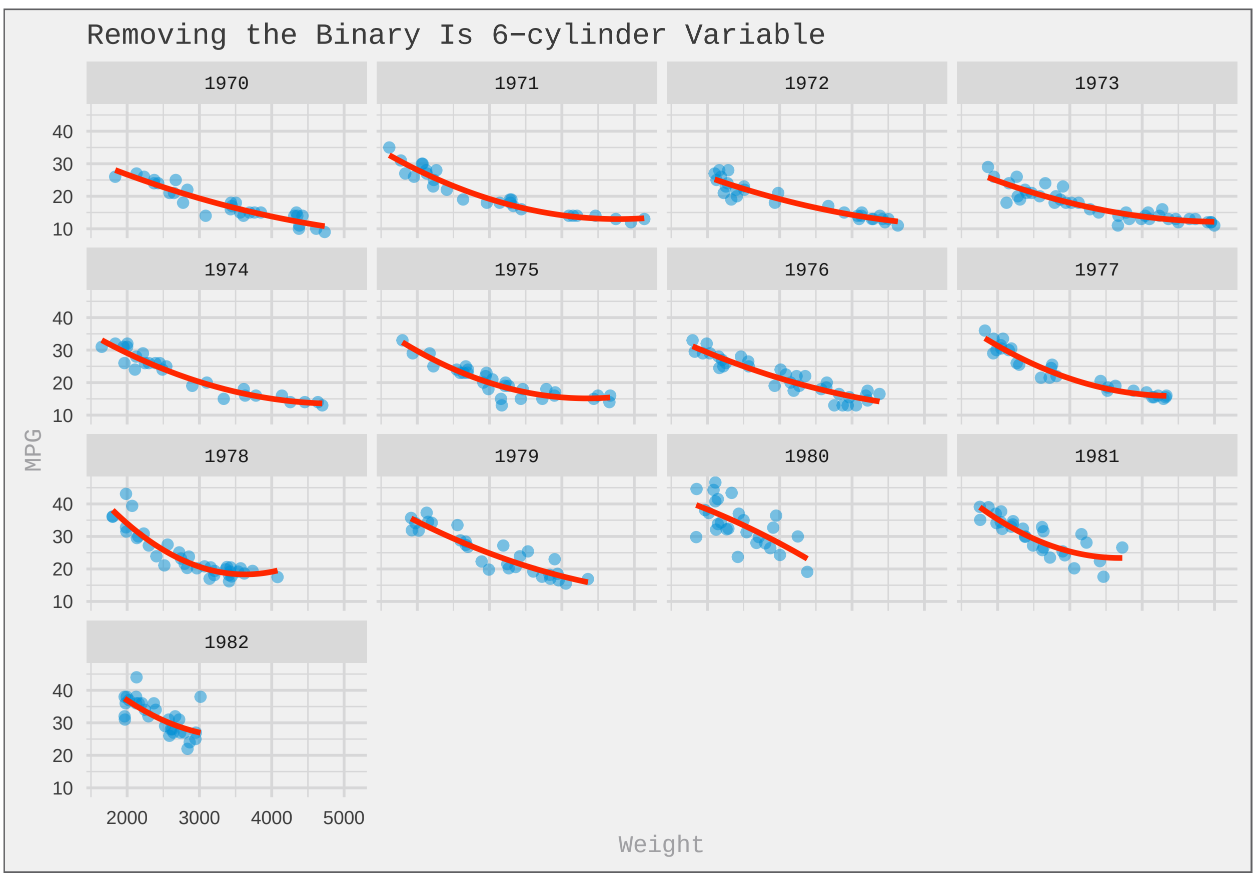
```r
auto_fit14 <- lm(mpg ~ weight + I(weight^2) + year, Auto_proper2)
summary(auto_fit14)
```

```
##
## Call:
## lm(formula = mpg ~ weight + I(weight^2) + year, data = Auto_proper2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.456 -1.708 -0.173  1.519 13.179
##
## Coefficients:
##                   Estimate      Std. Error t value            Pr(>|t|)
## (Intercept) -1572.841690848    87.410981733   -18.0 <0.0000000000000002
## weight         -0.021547967     0.001440887   -14.9 <0.0000000000000002
## I(weight^2)     0.000002348     0.000000225    10.4 <0.0000000000000002
## year            0.828927644     0.044300113    18.7 <0.0000000000000002
##
```

```
## (Intercept) ***
## weight      ***
## I(weight^2) ***
## year        ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.03 on 388 degrees of freedom
## Multiple R-squared:  0.85,   Adjusted R-squared:  0.849
## F-statistic:  734 on 3 and 388 DF,  p-value: <0.0000000000000002
```

When we plot this model the results look great.

```
Auto_proper2 %>%
    select(mpg, weight, year, cylinders, is_6cylinder) %>%
    ggplot(aes(x = weight, y = mpg)) +
    facet_wrap(~ year) +
    geom_point(color = pal538['blue'][[1]], alpha = 0.5) +
    theme_jrf() +
    geom_smooth(method = "lm", formula = y ~ x + I(x^2), se = FALSE, colour = pal538['red'][[1]]) +
    labs(title = "Removing the Binary Is 6-cylinder Variable", x = "Weight", y = "MPG")
```



Let's compare the AIC and BIC values for all of these models.

```r
data_frame(
    Model = c("1: Cylinders - all levels", "2: Binary 6-cylinder",
              "3: Binary 6-cylinder & Horsepower","4: Cylinders - all levels & Horsepower",
              "5: Binary 6-cylinder and Quadratic Weight", "6: Quatric Weight Only")
    , AIC = AIC(auto_fit9, auto_fit10, auto_fit11, auto_fit12, auto_fit13, auto_fit14)$AIC
    , BIC = BIC(auto_fit9, auto_fit10, auto_fit11, auto_fit12, auto_fit13, auto_fit14)$BIC
) %>%
    kable()
```

Model	AIC	BIC
1: Cylinders - all levels	2034	2066
2: Binary 6-cylinder	2048	2068
3: Binary 6-cylinder & Horsepower	2042	2066
4: Cylinders - all levels & Horsepower	2031	2067
5: Binary 6-cylinder and Quadratic Weight	1987	2011
6: Quatric Weight Only	1988	2008

There is only a slight information gain with the binary 6-cylinder variable and we believe this to be overfiting. We will proceed with model 6. Let's check what would happen if we used `regsubsets` with a the quadratic term.

```r
auto_fit15 <- regsubsets(mpg ~ . + I(weight^2), data = Auto_proper2 %>% select(-name, -year2), nvmax = 
```

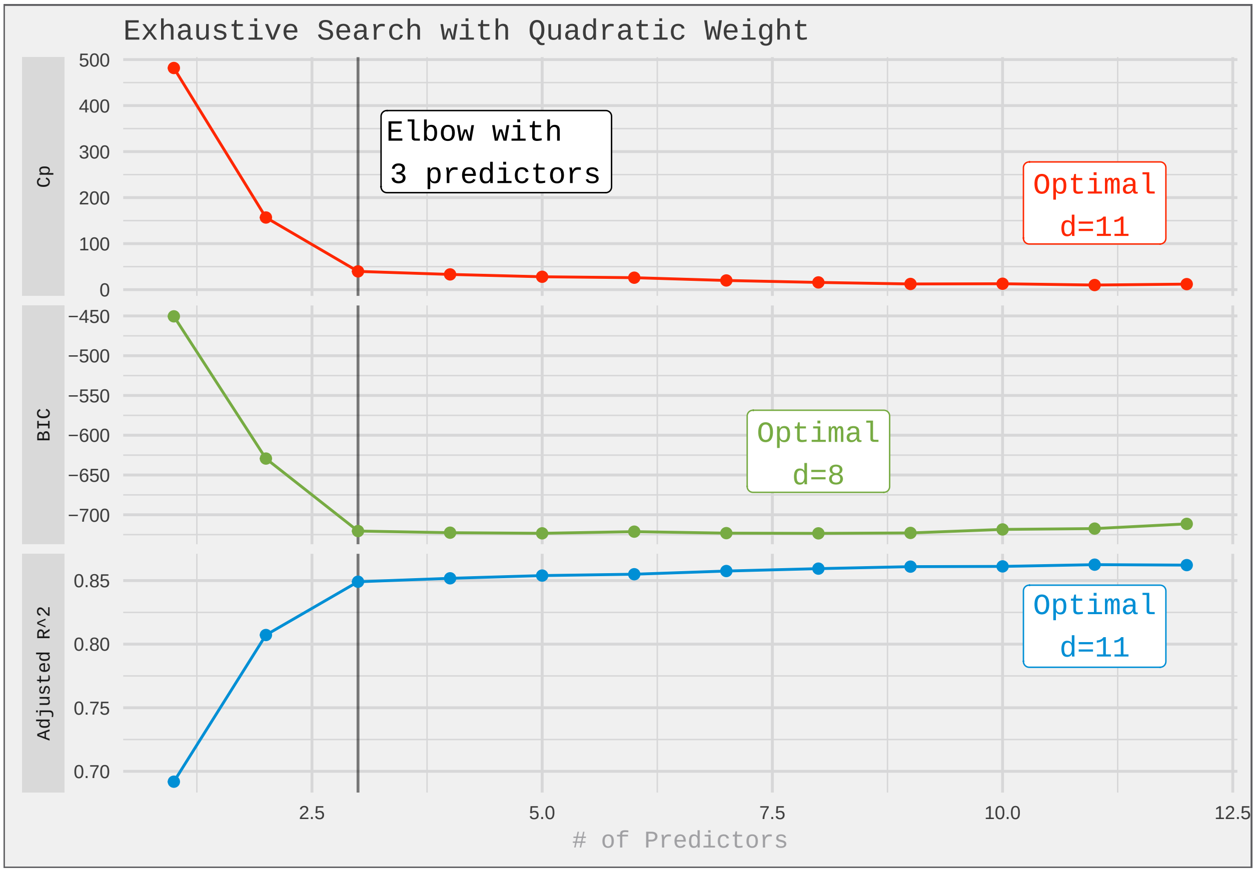
```
## Reordering variables and trying again:
```

```r
auto_fit15_sum <- summary(auto_fit15)
as_data_frame(auto_fit15_sum$outmat) %>% print(width = Inf)
```

```
## # A tibble: 12 × 13
##    cylinders4 cylinders5 cylinders6 cylinders8 displacement horsepower
##         <chr>      <chr>      <chr>      <chr>        <chr>      <chr>
## 1
## 2
## 3
## 4           *
## 5           *          *
## 6           *          *                                             *
## 7           *          *                     *
## 8           *                                             *          *
## 9           *          *                                  *          *
## 10          *          *                     *                       *
## 11          *          *          *          *            *          *
## 12          *          *          *          *            *          *
##    weight acceleration  year originEuropean originJapanese
##     <chr>        <chr> <chr>          <chr>          <chr>
## 1       *
## 2       *                  *
## 3       *                  *
## 4       *                  *
## 5       *                  *
```

```
## 6       *                 *
## 7       *                 *
## 8       *                 *             *             *
## 9       *                 *             *             *
## 10      *                 *             *             *
## 11      *                 *             *             *
## 12      *           *     *             *             *
##    is_6cylinderTRUE `I(weight^2)`
##               <chr>         <chr>
## 1
## 2
## 3                               *
## 4                               *
## 5                               *
## 6                               *
## 7                 *             *
## 8                               *
## 9                               *
## 10                *             *
## 11                              *
## 12                              *
```

```
data_frame(
      predictors = 1:length(auto_fit15_sum$cp)
    , cp = auto_fit15_sum$cp
    , bic = auto_fit15_sum$bic
    , adjr2 = auto_fit15_sum$adjr2
) %>%
    gather(metric, value, -predictors) %>%
    mutate(metric = factor(metric, levels = c("cp","bic","adjr2"))) %>%
    ggplot(aes(x = predictors, y = value, colour = metric)) +
    facet_grid(metric ~ ., scale = "free_y", switch = "y",
               labeller = ggplot2::labeller(metric = c(cp = "Cp", bic = "BIC", adjr2 = "Adjusted R^2"))
    geom_vline(xintercept = 3, alpha = 0.5) + geom_line() + geom_point() +
    geom_label(data = data_frame(
        predictors = c(which.min(auto_fit15_sum$cp), which.min(auto_fit15_sum$bic), which.max(auto_fit1
        , metric = factor(c("cp","bic","adjr2"), levels = c("cp","bic","adjr2"))
        , value = c(min(auto_fit15_sum$cp), min(auto_fit15_sum$bic), max(auto_fit15_sum$adjr2))
        , label = paste0("Optimal\nd=", c(which.min(auto_fit15_sum$cp), which.min(auto_fit15_sum$bic), w
        , vjust = c(-.5, -.5, 1.25)
    ), aes(x = predictors, y = value, label = label, vjust = vjust), family = "DecimaMonoPro") +
    theme_jrf() +
    labs(title = "Exhaustive Search with Quadratic Weight", x = "# of Predictors", y = NULL) +
    geom_label(data = data_frame(x = 3, y = 300, metric = factor(c("cp"), levels = c("cp","bic","adjr2")
                label = "Elbow with\n3 predictors"), aes(x=x,y=y,label=label), colour = "black", hjust =
               family = "DecimaMonoPro") +
    scale_colour_manual(guide = FALSE, values = c(pal538['red'][[1]], pal538['green'][[1]], pal538['blu
```



The result confirms our thinking: a 3 predictor model with a quadratic weight term.

5.4.4.2 Summary

Our final model to predict MPG based on the predictors in the *Auto* dataset is

$$MPG = \beta_0 + \beta_1 Weight + \beta_2 Weight^2 + \beta_3 + year$$

```
(auto_fit_final <- summary(auto_fit14))
```

```
##
## Call:
## lm(formula = mpg ~ weight + I(weight^2) + year, data = Auto_proper2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.456 -1.708 -0.173  1.519 13.179
##
## Coefficients:
##                    Estimate      Std. Error t value             Pr(>|t|)
## (Intercept) -1572.841690848    87.410981733   -18.0 <0.0000000000000002
## weight         -0.021547967     0.001440887   -14.9 <0.0000000000000002
## I(weight^2)     0.000002348     0.000000225    10.4 <0.0000000000000002
## year            0.828927644     0.044300113    18.7 <0.0000000000000002
```

```
##
## (Intercept) ***
## weight      ***
## I(weight^2) ***
## year        ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.03 on 388 degrees of freedom
## Multiple R-squared:  0.85,   Adjusted R-squared:  0.849
## F-statistic:  734 on 3 and 388 DF,  p-value: <0.0000000000000002
```

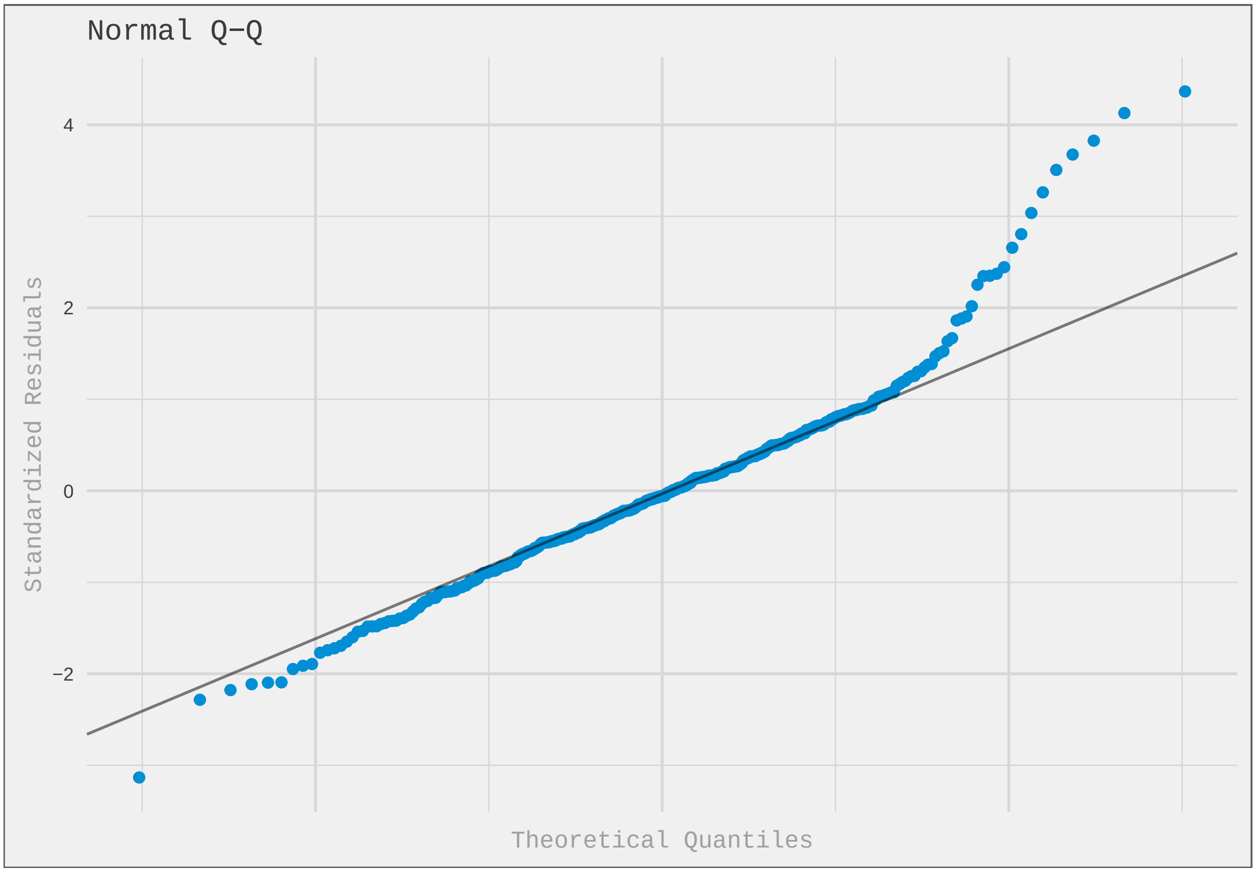
Thus the model is

$$MPG = -1572.84 + -0.02Weight + 0.0000023477159362345Weight^2 + 0.83year$$

5.4.4.2.1 Checking Model Assumptions

The normal Q-Q plot shows that residuals might not come from a normal distribution at the tails, but all together is somewhat normal.

```r
data_frame(std.resid = rstandard(auto_fit14)) %>%
    ggplot() +
    stat_qq(aes(sample = std.resid), colour = pal538['blue']) +
    geom_abline(data =
        . %>%
        summarise(
            slope = diff(quantile(std.resid, c(0.25, 0.75))) / diff(qnorm(c(0.25, 0.75)))
            , int = quantile(std.resid, c(0.25, 0.75))[1L] -
               (diff(quantile(std.resid, c(0.25, 0.75))) /
                    diff(qnorm(c(0.25, 0.75)))) * qnorm(c(0.25, 0.75))[1L]
        ),
        aes(slope = slope, intercept = int), alpha = 0.5
    ) +
    theme_jrf() +
    scale_x_continuous(labels = NULL) +
    labs(title = "Normal Q-Q", y = "Standardized Residuals", x = "Theoretical Quantiles")
```



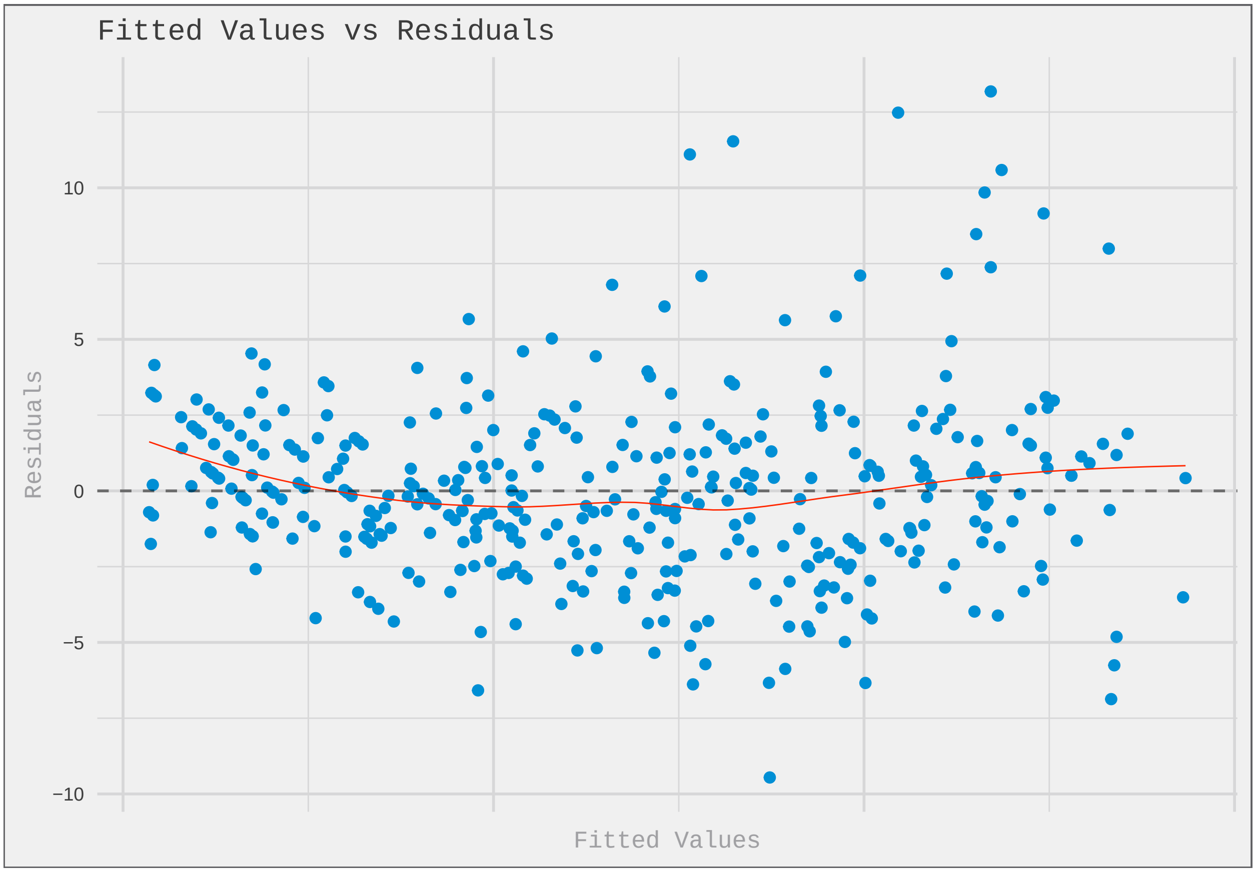
In addition, the Shapiro-Wilks test shows that we have evidence that the residuals do not come from a normal distribution.

```
shapiro.test(rstandard(auto_fit14))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(auto_fit14)
## W = 0.95, p-value = 0.0000000004
```

The fitted values vs residuals plots show approximately equal variance of the residuals (i.e. no heteroscedasticity).

```
data_frame(
    fitted = auto_fit14$fitted.values
    , resid = auto_fit14$residuals
    ) %>%
    ggplot(aes(x = fitted, y = resid)) +
    geom_point(colour = pal538['blue']) +
    geom_smooth(method = "loess", colour = pal538['red'], se = FALSE, size = .25, alpha = 0.5) +
    geom_hline(yintercept = 0, alpha = 0.5, linetype = 'dashed', color = 'black') +
    theme_jrf() +
    scale_x_continuous(labels = NULL) +
    labs(title = "Fitted Values vs Residuals", y = "Residuals", x = "Fitted Values")
```



5.4.4.2.2 Statistical Inference

- The F-test for regression provides extremely strong evidence against the hypothesis that none of the variables are related to the response MPG (*P-value* $\approx$ *0*).
- The Multiple R2 is 0.85 indicating that 90% of the variation in car MPG is explained by the variation in weight and model year, so the model will be good for prediction, however normality is not satisfied so predictions may be unreliable.
- The intercept is not meaningful (*MPG* = *0*).
- We have extremely strong evidence against the hypotheses that the coefficients associated with weight are equal to 0 (*P-values* $\approx$ *0*).
- We have extremely evidence against the hypothesis that the coefficient associated with model year is equal to 0 (*P-value* $\approx$ *0*).

The effects associated with weight are difficult to describe because of the quadratic term. Holding the effect of weight constant, each additional year of car (newer model years), a car's MPG increases by 0.8289.

Finally, we can find a 95% prediction interval for the car built in 1983.

```
future_car_pred <- predict(auto_fit14, future_car, interval = "prediction")
```

The prediction interval for the MPG of this car is

$$(16.2702, 28.3166)$$