STAT 471/571/701 Modern Data Mining - HW 1

Kexin Zhu
Yang Yi
Yifan Jiang

Due: 11:59PM February 3, 2019

Overview / Instructions

- **Homework assignments can be done in a group consisting of up to three members**. Please find your group members as soon as possible and register your group on our canvas site.
- **All work submitted should be completed in the R markdown format.** You can find a cheat sheet for R Markdown here. For those who have never used it before we urge you to start this homework as soon as possible.
- **Submit a zip file containing the (1) Rmd file, (2) a compiled PDF or HTML version, and (3) all necessary data files. Note: Please only upload ONE submission per HW team**. You can directly edit this file to add your answers. If you intend to work on the problems separately within your group, compile your answers into one Rmd file before submitting. We encourage that you at least attempt each problem by yourself before working with your teammates. Additionally, ensure that you can 'knit' or compile your Rmd file. It is also likely that you need to configure Rstudio to properly convert files to PDF. **These instructions** should be helpful.
- In general, be as concise as possible while giving a fully complete answer. All necessary datasets are available in the `Data` folder on Canvas. Make sure to document your code with comments so the teaching fellows can follow along. R Markdown is particularly useful because it follows a 'stream of consciousness' approach: as you write code in a code chunk, make sure to explain what you are doing outside of the chunk.
- A few good submissions will be used as sample solutions. When those are released, make sure to compare your answers and understand the solutions.

Question 0

Review the code and concepts covered during lecture.

Simple Regression

Question 1

This exercise is designed to help you understand the linear model using simulations. In this exercise, we will generate (x_i, y_i) pairs so that all linear model assumptions are met.

Presume that x and y are linearly related with a normal error ϵ , such that $y = 1 + 1.2x + \epsilon$. The standard deviation of the error is $\sigma = 2$.

We can create a sample input vector $(n = 40)$ for x with the following code:

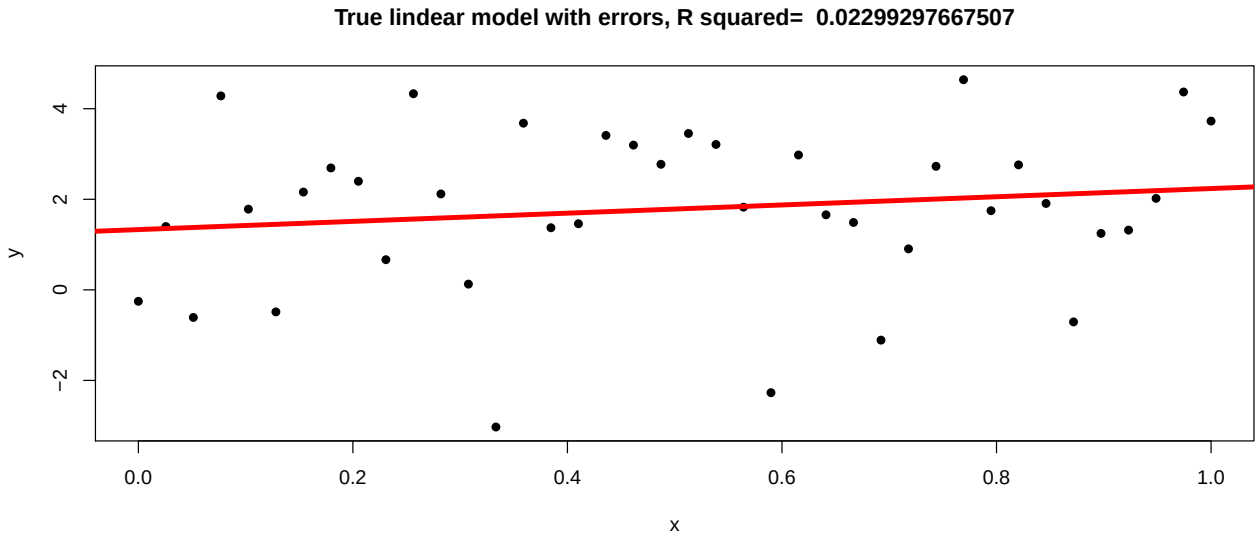
```
# Generates a vector of size 40 with equally spaced values between 0 and 1, inclusive
x <- seq(0, 1, length = 40)
```

Q1.1 Generate data

Create a corresponding output vector for y according to the equation given above. Use `set.seed(1)`. Then, create a scatterplot with (x, y) pairs. Base R plotting is acceptable, but if you can, please attempt to use `ggplot2` to create the plot. Make sure to have clear labels and sensible titles on your plots.

```
set.seed(1)
y = 1 + 1.2*x + rnorm(length(x), sd = 2)

myfit <- lm(y~x)
myfit.out <- summary(myfit)
rsquared <- myfit.out$r.squared
plot(x, y, pch=16, ylab="y",
    xlab = "x",
    main = paste("True lindear model with errors, R squared= ", rsquared))
abline(lm(y~x), col="red", lwd=4)
```



Q1.2 Understand the model

i. Find the LS estimates of β_0 and β_1, using the `lm()` function. What are the true values of β_0 and β_1? Do the estimates look to be good?

Answer: The LS estimates of β_0 is 1.3308, and β_1 is 0.9064. The true values of β_0 is 1, and β_1 is 1.2. Yes, the estimates look to be good.

ii. What is your RSE for this linear model fit? Is it close to $\sigma = 2$?

Answer: RSE for this linear model fit is 1.79, which is close to $\sigma = 2$.

ii. What is the 95% confidence interval for β_1? Does this confidence interval capture the true β_1?

Answer: The 95% confidence interval for β_1 is (-1.01, 2.83), and the true value of $\beta_1 = 1.2$, which is in the confidence interval.

iii. Overlay the LS estimates and the true lines of the mean function onto a copy of the scatterplot you made above.

Answer: Run the following R code for results.

```r
# i
data_x_y <- data.frame("x" = x,"y" = y)
myfit_0 <- lm(y~x, data=data_x_y)
lm_summary <- summary(myfit_0)$coefficients
beta_0 <- lm_summary[1, 1]
beta_1 <- lm_summary[2, 1]
beta_0
```

```
## [1] 1.330834
```

```r
beta_1
```

```
## [1] 0.9064362
```

```r
# ii
RSS <- sum((myfit_0$res)^2)
RSE <- sqrt(RSS/myfit_0$df)
RSE
```

```
## [1] 1.794303
```

```r
# ii
summary(myfit_0)
```

```
##
## Call:
## lm(formula = y ~ x, data = data_x_y)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6624 -0.8797  0.0139  1.2468  2.8823
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.3308     0.5569   2.390   0.0219 *
## x             0.9064     0.9585   0.946   0.3503
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.794 on 38 degrees of freedom
## Multiple R-squared:  0.02299,    Adjusted R-squared:  -0.002718
## F-statistic: 0.8943 on 1 and 38 DF,  p-value: 0.3503
```

```r
CI_HIGH <- 0.9064 + 2 * 0.96
CI_LOW <- 0.9064 - 2 * 0.96
CI_LOW
```

```
## [1] -1.0136
```

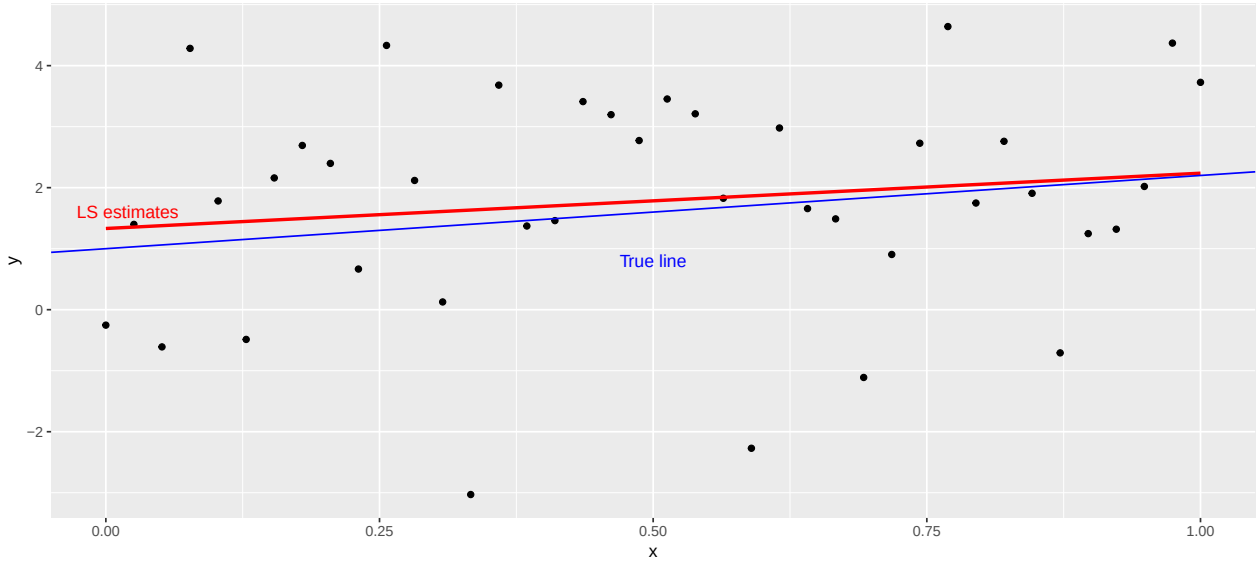
```r
CI_HIGH
```

```
## [1] 2.8264
```

```r
# iii
library(ggplot2)

ggplot(data_x_y,aes(x=x,y = y)) + geom_point() + geom_smooth(method="lm", se = F,color = "red") +
  geom_abline(intercept=1,slope = 1.2 ,color = "blue") +
```

```
  annotate(geom="text", x=0.5, y=0.5, label="True line",color="blue", vjust = -1) +
  annotate(geom="text", x=0.02, y=1.3, label="LS estimates",color="red", vjust = -1)
```



Q1.3 Model diagnoses

i. Provide residual plot of `x = fitted y`, `y = residuals`.

Answer: Run the following R code for results.

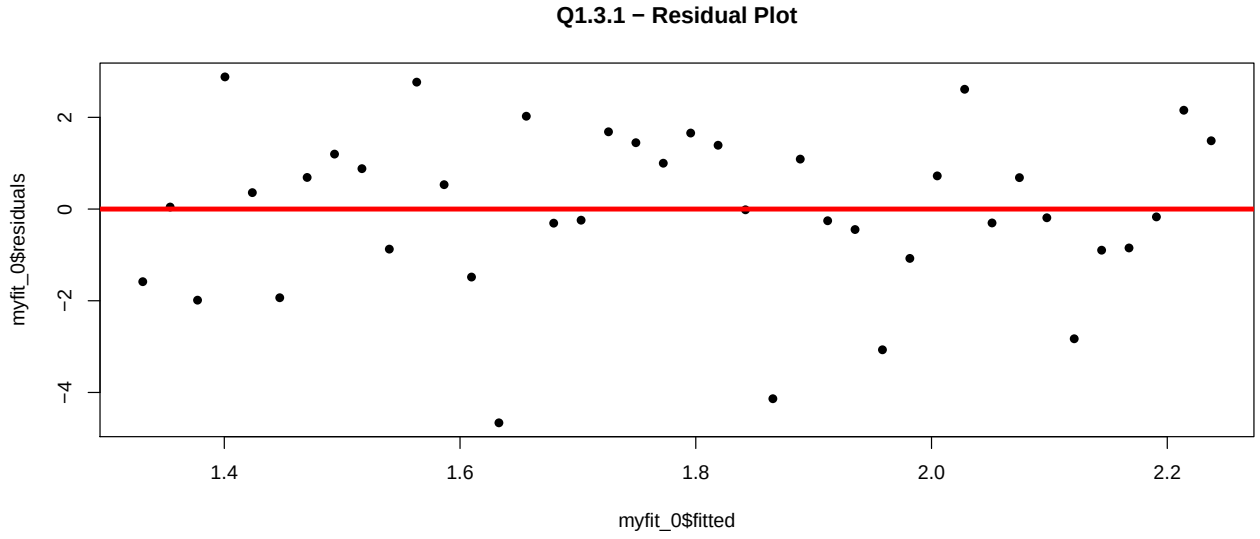
ii. Provide a QQ-Normal plot of the residuals

Answer: Run the following R code for results.

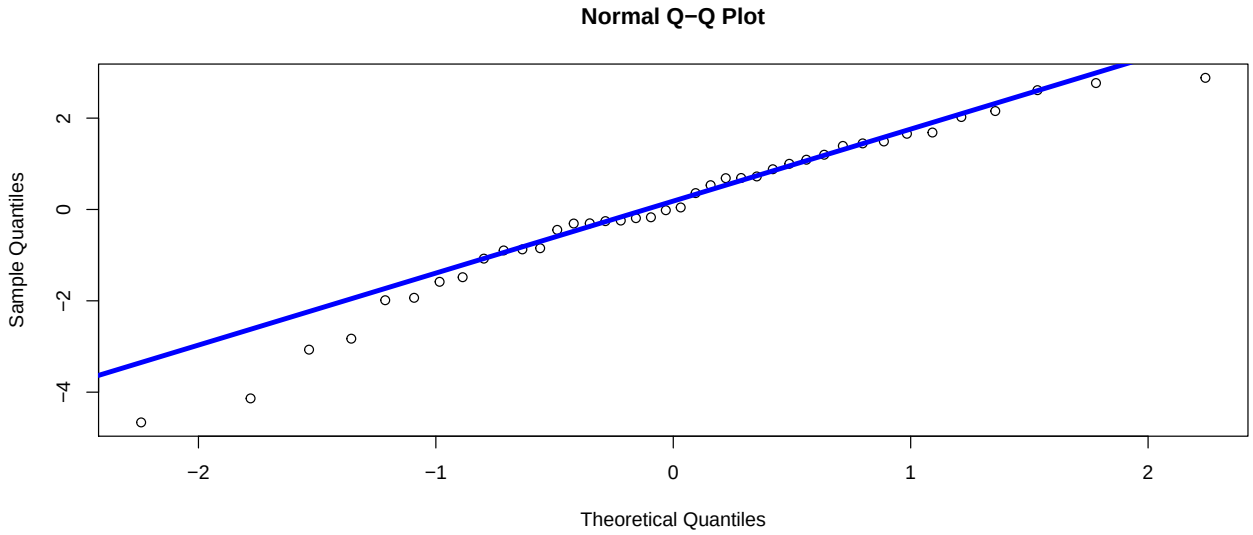
iii. Comment on how well the model assumptions are met for the sample you used.

Answer: The data have been relatively well selected since the residual is small and is fit to the real model.

```
# i
plot(myfit_0$fitted, myfit_0$residuals, pch  = 16, main = "Q1.3.1 - Residual Plot")
abline(h = 0, lwd = 4, col = "red")
```



```r
# ii
qqnorm(myfit_0$residuals)
qqline(myfit_0$residuals, lwd=4, col="blue")
```



Q1.4 Understand sampling distribution and confidence intervals

This part aims to help you understand the notion of sampling statistics and confidence intervals. Let's concentrate on estimating the slope only.

Generate 100 samples of size $n = 40$, and estimate the slope coefficient from each sample. We include some sample code below, which should guide you in setting up the simulation. Note: this code is easier to follow but suboptimal; see the appendix for a more optimal R-like way to run this simulation.

```r
# Inializing variables. Note b_1, upper_ci, lower_ci are vectors
x <- seq(0, 1, length = 40)
n_sim <- 100               # number of simulations
b1 <- 0                    # n_sim many LS estimates of beta_1 (=1.2). Initialize to 0 for now
upper_ci <- 0              # upper bound for beta_1. Initialize to 0 for now.
lower_ci <- 0              # lower bound for beta_1. Initialize to 0 for now.
t_star <- qt(0.975, 38)    # Food for thought: why 38 instead of 40? What is t_star?

# Perform the simulation
for (i in 1:n_sim){
  y <- 1 + 1.2 * x + rnorm(40, sd = 2)
  lse <- lm(y ~ x)
  lse_output <- summary(lse)$coefficients
  se <- lse_output[2, 2]
  b1[i] <- lse_output[2, 1]
  upper_ci[i] <- b1[i] + t_star * se
  lower_ci[i] <- b1[i] - t_star * se
}
results <- as.data.frame(cbind(se, b1, upper_ci, lower_ci))

# remove unecessary variables from our workspace
# rm(se, b1, upper_ci, lower_ci, x, n_sim, b1, t_star, lse, lse_out)
```

i. Summarize the LS estimates of β_1 (stored in `results$b1`). Does the sampling distribution agree with

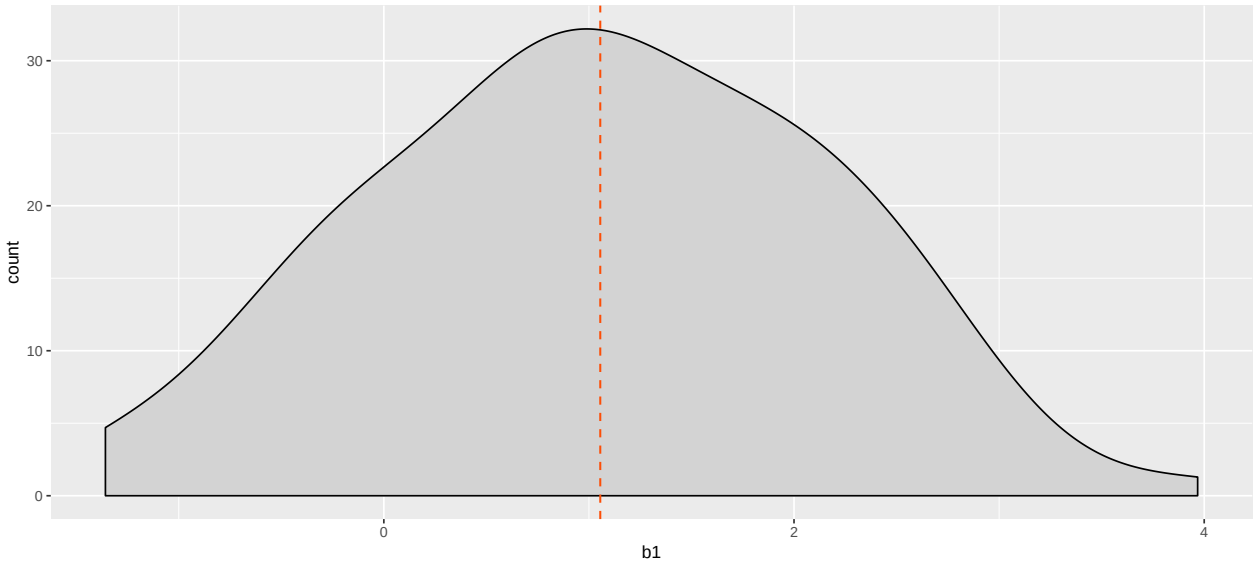
theory?

Answer: Yes, the sampling distribution is similar to normal distribution and $\mu(\beta_1) = 1.3$, which is close to *true* $\beta_1 = 1.2$.

ii. How many of your 95% confidence intervals capture the true β_1? Display your confidence intervals graphically.

Answer: 92 samples 95% confidence intervals capture the true β_1. Run the following R code for the graph.

```
# i
ggplot(results, aes(x = b1)) + geom_density(aes(y = ..count..), fill = "lightgray") +
  geom_vline(aes(xintercept = mean(b1)),linetype = "dashed", size = 0.6,color = "#FC4E07")
```



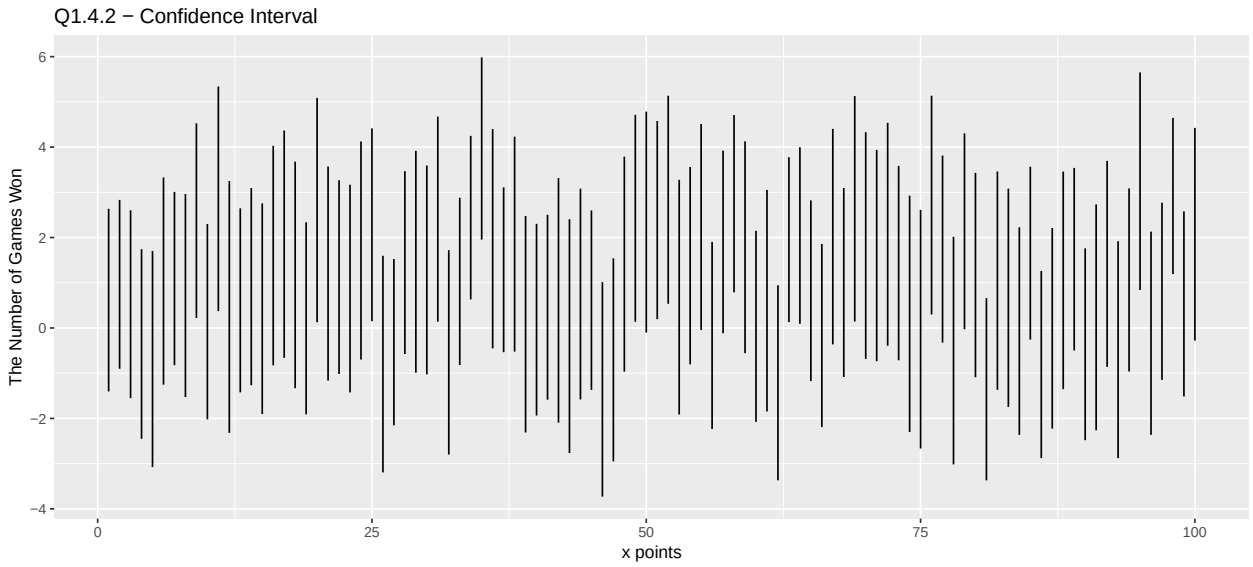

```
mean(b1)
```

```
## [1] 1.055236
```

```
# ii
sum(upper_ci > 1.2 & lower_ci < 1.2)
```

```
## [1] 96
```

```
ggplot(results)+geom_linerange(aes(x = seq(1,length(b1),1), ymin=lower_ci, ymax=upper_ci)) + labs(title
```



Question 2

This question is about Major League Baseball (MLB) and team payrolls. Guiding questions: how do salaries paid to players affect team wins? How could we model win propensity?

We have put together a dataset consisting of the winning records and the payroll data of all 30 MLB teams from 1998 to 2014. There are 54 variables in the dataset, including:

- `payroll`: total team payroll (in $billions) over the 17-year period
- `avgwin`: the aggregated win percentage over the 17-year period
- winning percentage and payroll (in $millions) for each team are also broken down for each year.

The data is stored as `MLPayData_Total.csv` on Canvas.

```
# setwd
salary <- read.csv("MLPayData_Total.csv")
```

Q2.1 Exploratory questions

For each of the following questions, there is a `dplyr` solution that you should try to answer with.

i. Which 5 teams spent the most money in total between years 2000 and 2004, inclusive?

Answer: The 5 teams that spent the most money in total are New York Yankees, Boston Red Sox, Los Angeles Dodgers, New York Mets, Atlanta Braves.

ii. Between 1999 and 2000, inclusive, which team(s) "improved" the most? That is, had the biggest percentage gain in wins?

Answer: Chicago White Sox and St. Louis Cardinals improved the most between 1999 and 2000.

iii. Using `ggplot`, pick a single year, and plot the number of games won vs. `payroll` for that year (`payroll` on x-axis). You may use any 'geom' that makes sense, such as a scatterpoint or a label with the point's corresponding team name.

Answer: Run the following R code for results of year 1998.

```
#i
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
money_select_year <- salary %>% select(Team.name.2014, p2000, p2001, p2002, p2003, p2004)
money_total <- money_select_year %>% mutate(total_money = p2000 + p2001 + p2002 + p2003 + p2004) %>% se
top_five_team <- money_total %>% arrange(desc(total_money)) %>% top_n(5)
```

```
## Selecting by total_money
```

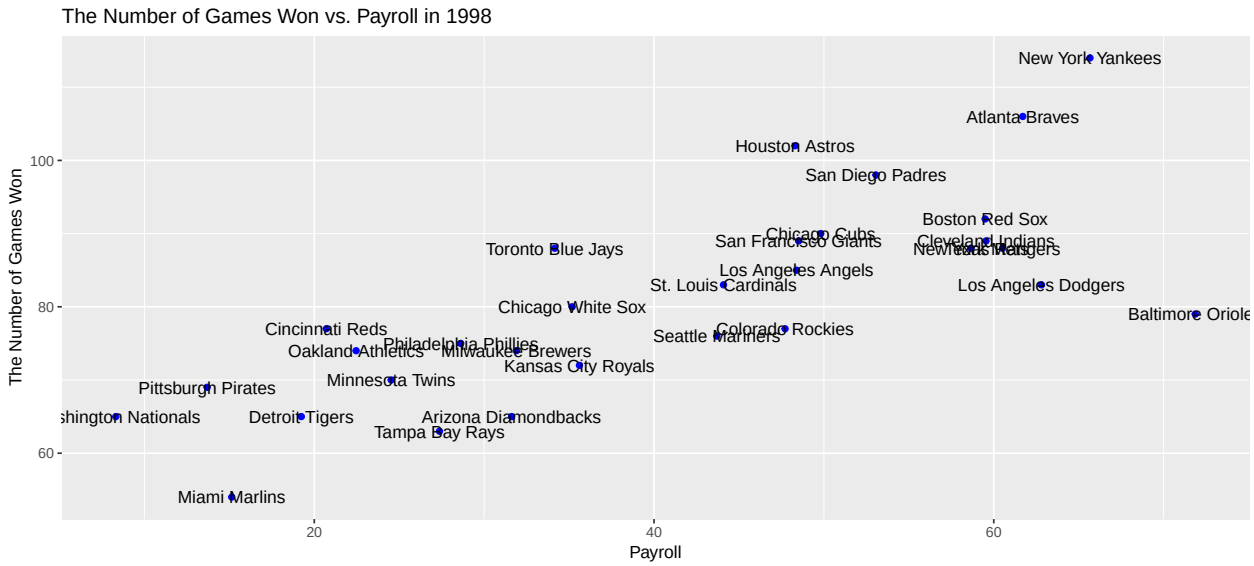
```
top_five_team_name <- top_five_team$Team.name.2014
top_five_team_name
```

```
## [1] New York Yankees    Boston Red Sox      Los Angeles Dodgers
## [4] New York Mets       Atlanta Braves
## 30 Levels: Arizona Diamondbacks Atlanta Braves ... Washington Nationals
```

```
#ii
win_select_year <-salary %>% select(Team.name.2014, X1999, X2000)
win_improve <- win_select_year %>% mutate(improve_rate = (X2000 - X1999) / X1999)
most_improve <- win_improve %>% filter(improve_rate == max(improve_rate))
most_improve_name <- most_improve$Team.name.2014
most_improve_name
```

```
## [1] Chicago White Sox   St. Louis Cardinals
## 30 Levels: Arizona Diamondbacks Atlanta Braves ... Washington Nationals
```

```
#iii
ggplot(salary) +
  geom_point(aes(x = p1998, y = X1998), color = "blue") +
  geom_text(aes(x = p1998, y = X1998, label = Team.name.2014)) +
  labs(title = "The Number of Games Won vs. Payroll in 1998", x = "Payroll", y = "The Number of Games W
```



Q2.2

For a given year, is `payroll` a significant variable in predicting the winning percentage of that year? Choose a single year and run a regression to examine this. You may try this for a few different years. You can do this programmatically (i.e. for every year) if you are interested, but it is not required.

Answer: Payroll is not a significant variable in predicting in year 2000. We can build a null hypothesis: $H_0 : \beta_0 = 0$ $H_1 : \beta_0 \neq 0$ $pvalue = 0.076 > 0.01$, so failing to reject $H_0 : \beta_0 = 0$. Therefore, payroll is not a siginificant variable in year 2000.

```
fit0 = lm(X2000~p2000,data = salary)
summary(fit0)
```

```
##
## Call:
## lm(formula = X2000 ~ p2000, data = salary)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.6836  -7.3406   0.3098   5.8215  17.8425
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 72.36549    4.96779  14.567 1.35e-14 ***
## p2000        0.15392    0.08351   1.843   0.0759 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.597 on 28 degrees of freedom
## Multiple R-squared:  0.1082, Adjusted R-squared:  0.07635
## F-statistic: 3.397 on 1 and 28 DF,  p-value: 0.07592
```

```
fit1 = lm(X2002~p2002,data = salary)
summary(fit1)$coefficients[,4]
```

```
##  (Intercept)        p2002
## 1.937720e-09 1.458372e-02
```

```
fit2 = lm(X2004~p2004,data = salary)
summary(fit2)
```

```
##
## Call:
## lm(formula = X2004 ~ p2004, data = salary)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -30.290  -4.959   1.793   7.698  22.540
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  66.1653     5.0913  12.996 2.22e-13 ***
## p2004         0.2154     0.0673   3.201   0.0034 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.8 on 28 degrees of freedom
```

```
## Multiple R-squared:  0.2679, Adjusted R-squared:  0.2418
## F-statistic: 10.25 on 1 and 28 DF,  p-value: 0.003396
```

```
fit3 = lm(X2010~p2010,data = salary)
summary(fit3)
```

```
##
## Call:
## lm(formula = X2010 ~ p2010, data = salary)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.739  -8.301   1.611   9.448  16.917
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 71.86170    5.01697  14.324 2.06e-14 ***
## p2010        0.10040    0.05094   1.971   0.0587 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.5 on 28 degrees of freedom
## Multiple R-squared:  0.1218, Adjusted R-squared:  0.09046
## F-statistic: 3.884 on 1 and 28 DF,  p-value: 0.05869
```

Q2.3

With the aggregated information, use regression to analyze total payroll and overall winning percentage. Run appropriate model(s) to answer the following questions:

i. In this analysis, do the Boston Red Sox perform reasonably well given their total amount spent on payroll? [Use a 95% interval.]

Answer: The 95% interval for the team is (0.48, 0.60), and the average win is 0.549, which is in the interval. Therefore, Boston Red Sox performs reasonably well in this analysis.

ii. Given their winning percentage, how much would you have expected the Oakland A's to have spent on total payroll? [Use a 95% interval.]

Answer: The 95% interval for the team is (0.95, 2.27). Therefore, Oakland A's could spend on total payroll in the interval.

```
# i
myfit1 <- lm(avgwin ~ payroll, data=salary)
summary(myfit1)
```

```
##
## Call:
## lm(formula = avgwin ~ payroll, data = salary)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.040034 -0.017492  0.000936  0.010954  0.070302
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.42260    0.01534  27.555  < 2e-16 ***
```

```
## payroll      0.06137    0.01173   5.233 1.47e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02697 on 28 degrees of freedom
## Multiple R-squared:  0.4944, Adjusted R-squared:  0.4763
## F-statistic: 27.38 on 1 and 28 DF,  p-value: 1.469e-05
```

```
BRS <- salary %>% filter(Team.name.2014 == "Boston Red Sox")
mean(BRS$avgwin)
```

```
## [1] 0.5487172
```

```
CIpred <- predict(myfit1, BRS, interval="prediction", se.fit=TRUE)
CIpred
```

```
## $fit
##         fit       lwr       upr
## 1 0.5436382 0.4847828 0.6024936
##
## $se.fit
## [1] 0.009916123
##
## $df
## [1] 28
##
## $residual.scale
## [1] 0.02696692
```

```
# ii
myfit2 <- lm(payroll ~ avgwin, data=salary)
myfit2
```

```
##
## Call:
## lm(formula = payroll ~ avgwin, data = salary)
##
## Coefficients:
## (Intercept)       avgwin
##      -2.778        8.056
```

```
Oakland <- salary %>% filter(Team.name.2014 == "Oakland Athletics")
CIpred1 <- predict(myfit2, Oakland, interval="prediction", se.fit=TRUE)
CIpred1
```

```
## $fit
##        fit       lwr      upr
## 1 1.608288 0.9488246 2.267751
##
## $se.fit
## [1] 0.09043301
##
## $df
## [1] 28
##
## $residual.scale
## [1] 0.3089771
```

Multiple Regression

Question 3:

This question utilizes the `Auto` dataset from ISLR. The original dataset contains 408 observations about cars. It is similar to the CARS dataset that we use in our lectures. To get the data, first install the package ISLR. The `Auto` dataset should be loaded automatically. We'll use this dataset to practice the methods learnt so far.

You can access the necessary data with the following code:

```
# Read in the Auto dataset
auto_data <- ISLR::Auto
```

Get familiar with this dataset first. Tip: you can use the command `?ISLR::Auto` to view a description of the dataset.

Q3.1

Explore the data, with particular focus on pairwise plots and summary statistics. Briefly summarize your findings and any peculiarities in the data.

Answer: MPG has significant relationship with horsepower and weight. If taking the factors individually, the correlation between mpg and weight is the most salient, then comes the horsepower.

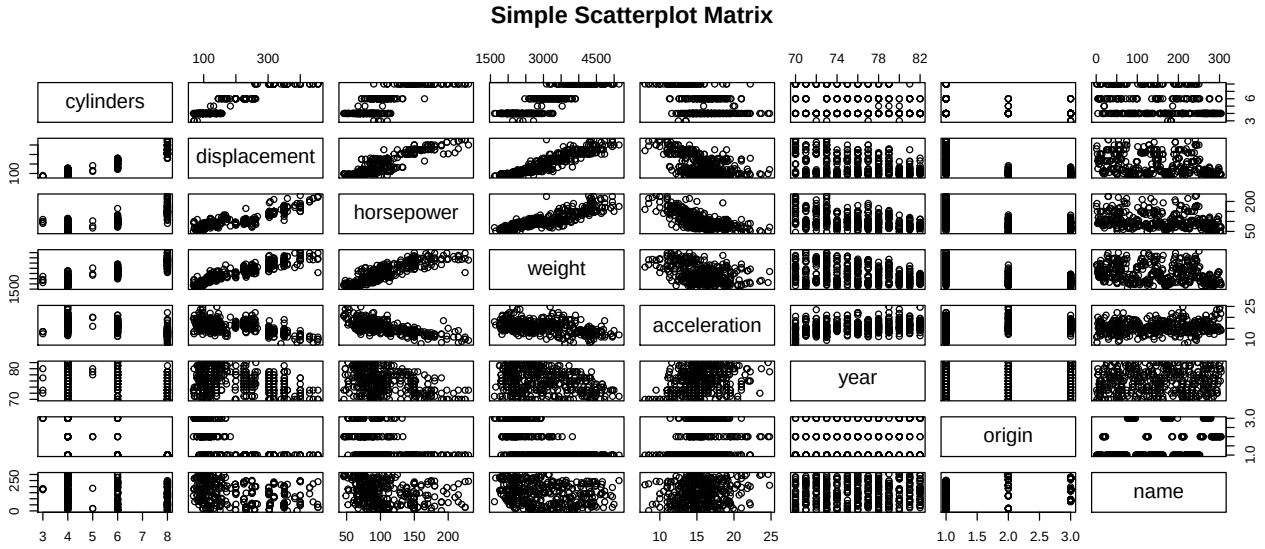
Step1: Inference for each $\beta_i's$, with seven hypothesis, fail to reject cylinders, horsepower and acceleration.

Step2: F-test H_0: at least,$\beta_1, \beta_3, \beta_5$ is 0. H_1: not H_0

Since p_value<0.05, reject the null hypothesis in 95% CI. However, p_value > 0.001, so accept the null hypothesis in 99% CI.

MPG is related with clinders, horsepower, weight, acceleration, year and origin, to be more siginificant, it can relate to displacement, weight, year and origin.

```
library(ggplot2)
?ISLR::Auto
pairs(auto_data[, -1], main="Simple Scatterplot Matrix") # pairwise plots
```



```r
# fit0 is with all B not equal to 0
fit0<-lm(mpg~cylinders+displacement+horsepower+weight+acceleration+year+origin,data = auto_data)
summary(fit0)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + year + origin, data = auto_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5903 -2.1565 -0.1169  1.8690 13.0604
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
## cylinders     -0.493376   0.323282  -1.526  0.12780
## displacement   0.019896   0.007515   2.647  0.00844 **
## horsepower    -0.016951   0.013787  -1.230  0.21963
## weight        -0.006474   0.000652  -9.929  < 2e-16 ***
## acceleration   0.080576   0.098845   0.815  0.41548
## year           0.750773   0.050973  14.729  < 2e-16 ***
## origin         1.426141   0.278136   5.127 4.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.328 on 384 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8182
## F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

```r
fit1 <- lm(mpg~weight+year+origin,data = auto_data) #under H0
RSS0 = sum((summary(fit1)$residuals)^2) # RSS(H_0)
RSS1 = sum((summary(fit0)$residuals)^2) # RSS(H_1)
F_stat = ((RSS0-RSS1)/3)/(RSS1/(392-7))
p_value = pf(F_stat,6,329-7,lower.tail = F)
p_value
```

```
## [1] 0.009221677
```

Q3.2

What effect does `time` have on `MPG`?

i. Start with a simple regression of `mpg` vs. `year` and report R's `summary` output. Is `year` a significant variable at the .05 level? State what effect `year` has on `mpg`, if any, according to this model.

Answer: Use Null hypothese
$H_0 : \beta_1 = 0$
$H_1 : \beta_1 \neq 0$
Since p_value < 0.05, reject H_0. Year is a significant variable at the 0.05 level.
We can say: mpg will increase by 1.23 when year increases by 1 unit. (Car will increase 1.23 in mpg if manufactured in one year later.)

ii. Add `horsepower` on top of the variable `year` to your linear model. Is `year` still a significant variable at the .05 level? Give a precise interpretation of the `year`'s effect found here.

Answer: Yes, year is still a significant variable at the 0.05 level.
We can say: mpg will increase by 0.65 when year increases by 1 unit among the cars with the same horsepower. (Cars with the same horsepower which are manufactured in one year later will have a 0.65 increase in mpg)

iii. The two 95% CI's for the coefficient of year differ among i) and ii). How would you explain the difference to a non-statistician?

Answer:
CI for auto1: (1.055,1.405)
CI for auto2: (0.53,0.79)
There is another variable introduced into the linear regression function, so the influence of the year will be reduced while horsepower will have a stronger influence on mpg.

iv. Create a model with interaction by fitting `lm(mpg ~ year * horsepower)`. Is the interaction effect significant at .05 level? Explain the year effect (if any).

Answer: Yes, the correlation proves to be significant at 0.05 level.
$H_0 : \beta_1 = 0$
$H_1 : \beta_1 \neq 0$
Since p_value < 0.05, reject H_0. Therefore, year*horsepower is a significant variable at the 0.05 level.
The model is: $mpg = -0.012 + 2.19 * year + 1.04 * horsepower - 0.015 * year * horsepower$. After extracting the year factor, we have: $mpg = -0.012 + year * (2.19 - 0.015 * horsepower) + 1.04 * horsepower$.
Thus, for one additional horsepower, cars manufactured in one year later will have an increase in mpg by 2.19-0.015 = 2.175.

```
# i
attach(auto_data)
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
auto1 <- lm(mpg~year)
summary(lm(mpg~year, data = auto_data))
```

```
##
## Call:
## lm(formula = mpg ~ year, data = auto_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.0212  -5.4411  -0.4412   4.9739  18.2088
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -70.01167    6.64516  -10.54   <2e-16 ***
## year          1.23004    0.08736   14.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.363 on 390 degrees of freedom
## Multiple R-squared:  0.337,  Adjusted R-squared:  0.3353
## F-statistic: 198.3 on 1 and 390 DF,  p-value: < 2.2e-16
```

```
#ii
auto2 <- lm(mpg~year)
summary(lm(mpg~year + horsepower, data = auto_data))
```

```
##
## Call:
## lm(formula = mpg ~ year + horsepower, data = auto_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.0768  -3.0783  -0.4308   2.5884  15.3153
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12.739166   5.349027  -2.382   0.0177 *
## year          0.657268   0.066262   9.919   <2e-16 ***
## horsepower   -0.131654   0.006341 -20.761   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.388 on 389 degrees of freedom
## Multiple R-squared:  0.6855, Adjusted R-squared:  0.6839
## F-statistic: 423.9 on 2 and 389 DF,  p-value: < 2.2e-16
```

```r
# iii
upper_ci1 = 1.23 + 2*summary(auto1)$coefficients[2,2]
lower_ci1 = 1.23 - 2*summary(auto1)$coefficients[2,2]

upper_ci2 = 0.66 + 2*summary(auto2)$coefficients[2,2]
lower_ci2 = 0.66 - 2*summary(auto2)$coefficients[2,2]

# iv
auto3 <- lm(mpg ~ year * horsepower)
summary(auto3)
```

```
##
## Call:
## lm(formula = mpg ~ year * horsepower)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.3492  -2.4509  -0.4557   2.4056  14.4437
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1.266e+02  1.212e+01 -10.449   <2e-16 ***
## year             2.192e+00  1.613e-01  13.585   <2e-16 ***
## horsepower       1.046e+00  1.154e-01   9.063   <2e-16 ***
## year:horsepower -1.596e-02  1.562e-03 -10.217   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.901 on 388 degrees of freedom
## Multiple R-squared:  0.7522, Adjusted R-squared:  0.7503
## F-statistic: 392.5 on 3 and 388 DF,  p-value: < 2.2e-16
```

Q3.3

Remember that the same variable can play different roles! Take a quick look at the variable `cylinders`, and try to use this variable in the following analyses wisely. We all agree that a larger number of cylinders will lower mpg. However, we can interpret `cylinders` as either a continuous (numeric) variable or a categorical variable.

i. Fit a model that treats `cylinders` as a continuous/numeric variable: `lm(mpg ~ horsepower + cylinders, ISLR::Auto)`. Is `cylinders` significant at the 0.01 level? What effect does `cylinders` play in this model?

Answer: Yes, p value<0.01. Each time when cylinder incresed by 1 unit, mpg will decrease by 1.9 unit.

ii. Fit a model that treats `cylinders` as a categorical/factor variable: `lm(mpg ~ horsepower + as.factor(cylinders), ISLR::Auto)`. Is `cylinders` significant at the .01 level? What is the effect of `cylinders` in this model? Use `anova(fit1, fit2)` and `Anova(fit2)` to help gauge the effect. Explain the difference between `anova()` and `Anova`.

Answer: 'anova' is a function in base R. 'Anova' is a function in the car package. The former calculates type I tests, that is, each variable is added in sequential order. The latter calculates type II or III tests. Type II tests test each variable after all the others. Generally, Anova() gives a summary table testing each variable, one at a time.

iii. What are the fundamental differences between treating `cylinders` as a continuous and categorical variable in your models?

Answer: Use a continuous variable in the model would relatively generate more accurate answers,since the continuous RSS is larger than the categorical RSS. It is because for the continuous model, there is always a range for the number of cylinders. However, if "cylinders" is treated as a categorical variable, each certain number of cylinders would be a binary variable in the model, which, in essense, introduces more explanatory variables into our model and lowers down the goodness of fit.

```
attach(auto_data)
```

```
## The following objects are masked from auto_data (pos = 3):
##
##     acceleration, cylinders, displacement, horsepower, mpg, name,
##     origin, weight, year
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
# i
auto4 <- lm(mpg ~ horsepower + cylinders, ISLR::Auto)
summary(auto4)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower + cylinders, data = ISLR::Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.4378  -3.2422  -0.3721   2.3532  16.9289
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.94842    0.77880  55.147  < 2e-16 ***
## horsepower  -0.08612    0.01119  -7.693 1.19e-13 ***
```

```
## cylinders   -1.91982    0.25261  -7.600 2.24e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.584 on 389 degrees of freedom
## Multiple R-squared:  0.6569, Adjusted R-squared:  0.6551
## F-statistic: 372.4 on 2 and 389 DF,  p-value: < 2.2e-16
```

```
cylinders <- summary(auto4)$coefficients[3,]

# ii
auto5 <- lm(mpg ~ horsepower + as.factor(cylinders), ISLR::Auto)
summary(auto5)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower + as.factor(cylinders), data = ISLR::Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5917 -2.7067 -0.6102  1.9001 16.3258
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            30.77614    2.41283  12.755  < 2e-16 ***
## horsepower             -0.10303    0.01133  -9.095  < 2e-16 ***
## as.factor(cylinders)4   6.57344    2.16921   3.030  0.00261 **
## as.factor(cylinders)5   5.07367    3.26661   1.553  0.12120
## as.factor(cylinders)6  -0.34406    2.18580  -0.157  0.87501
## as.factor(cylinders)8   0.49738    2.27639   0.218  0.82716
## ---
## 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:  0.7046, Adjusted R-squared:  0.7008
## F-statistic: 184.1 on 5 and 386 DF,  p-value: < 2.2e-16
```

```
anova(auto4,auto5)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ horsepower + cylinders
## Model 2: mpg ~ horsepower + as.factor(cylinders)
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    389 8172.5
## 2    386 7036.7  3    1135.8 20.769 1.705e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
```

```
##
##     recode
```

```
Anova(auto5)
```

```
## Anova Table (Type II tests)
##
## Response: mpg
##                       Sum Sq  Df F value    Pr(>F)
## horsepower            1507.8   1  82.712 < 2.2e-16 ***
## as.factor(cylinders) 2349.2   4  32.217 < 2.2e-16 ***
## Residuals             7036.7 386
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#summary(myfit5)
```

Q3.4

Final modelling question: we want to explore the effects of each feature as best as possible. You may explore interactions, feature transformations, higher order terms, or other strategies within reason. The model(s) should be as parsimonious (simple) as possible unless the gain in accuracy is significant from your point of view.

i. Describe the final model. Include diagnostic plots with particular focus on the model residuals and diagnoses.

Answer: $MPG = -18.61 - 0.006 \times weight + 0.75 \times year + 1.15 \times origin$ Residuals plot is roughly symmetry. In terms of the normal Q-Q plot, most data fit well.

ii. Summarize the effects found.

Answer: Despite of a large number of possible explanatory variables, we found that the influence of "year" and "cylinders" on mpg is the most salient. This means that, if manufactured in the same year, cars that have one fewer cylinder will gain an increase of about 3 in mpg; if with the same number of cylinders, cars that are manufactured one year newer will have an increase of 0.75 in mpg.

iii. Predict the `mpg` of the following car: A red car built in the US in 1983 that is 180 inches long, has 8 cylinders, displaces 350 cu. inches, weighs 4000 pounds, and has a horsepower of 260. Also give a 95% CI for your prediction.

Answer: The mpg of the given car is predicted as 21.79 based on our linear model. 95% confidence interval for our prediction is (14.1,27.5).

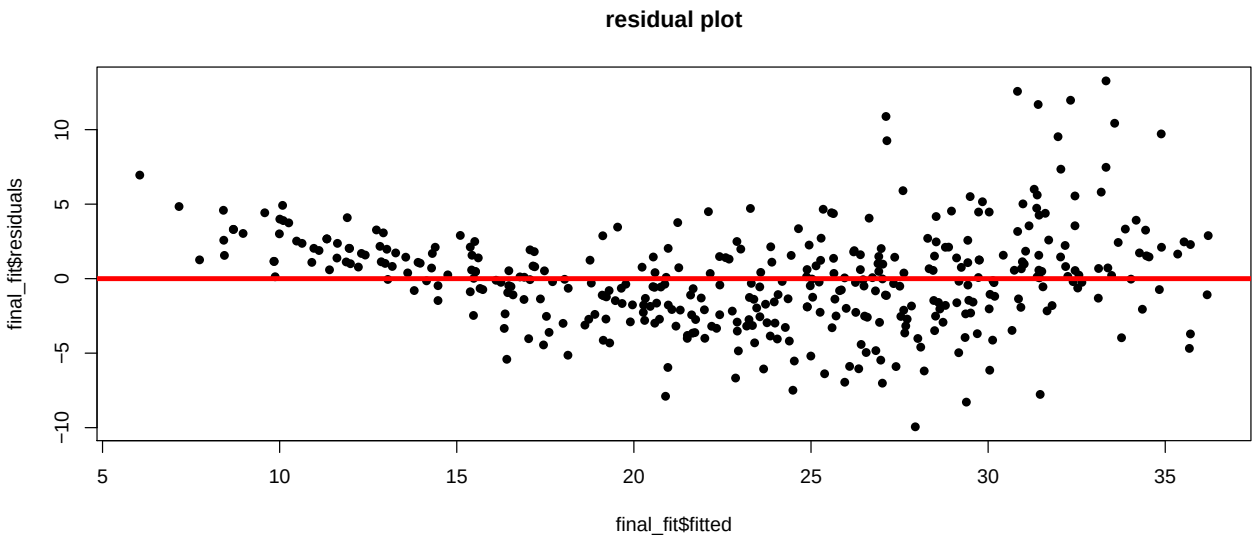
```
cars <- ISLR::Auto
# i

final_fit <- lm(mpg~weight+year+origin,data = cars)
summary(final_fit)
```

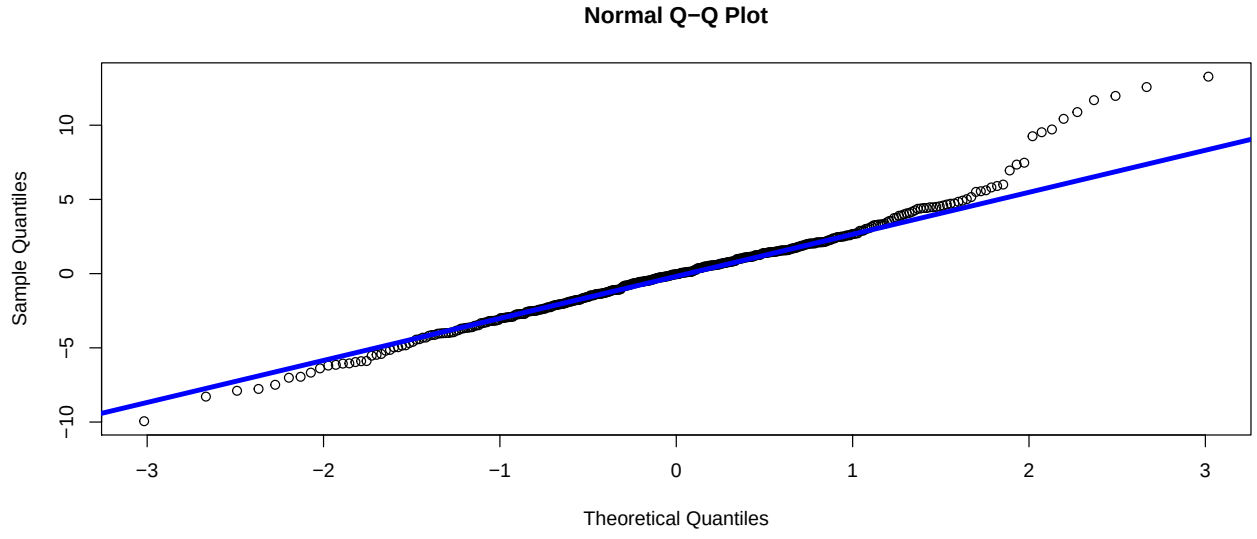
```
##
## Call:
## lm(formula = mpg ~ weight + year + origin, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.9440 -2.0948 -0.0389  1.7255 13.2722
##
```

```
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.805e+01  4.001e+00  -4.510 8.60e-06 ***
## weight      -5.994e-03  2.541e-04 -23.588  < 2e-16 ***
## year         7.571e-01  4.832e-02  15.668  < 2e-16 ***
## origin       1.150e+00  2.591e-01   4.439 1.18e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.348 on 388 degrees of freedom
## Multiple R-squared:  0.8175, Adjusted R-squared:  0.816
## F-statistic: 579.2 on 3 and 388 DF,  p-value: < 2.2e-16
```

```
plot(final_fit$fitted, final_fit$residuals,
     pch  = 16,
     main = "residual plot")
abline(h=0, lwd=4, col="red")
```




```
qqnorm(final_fit$residuals)
qqline(final_fit$residuals, lwd=4, col="blue")
```



```r
#iii
mpg = -18.61 -0.006*4000+ 0.75*83+1.15*1
mpg
```

```
## [1] 20.79
```

```r
upper_ci = mpg+2*summary(final_fit)$sigma
lower_ci = mpg-2*summary(final_fit)$sigma
lower_ci
```

```
## [1] 14.09479
```

```r
upper_ci
```

```
## [1] 27.48521
```

Appendix

This is code that is roughly equivalent to what we provide above in Question 2 but is more streamlined (simulations).

```r
simulate_lm <- function(n) {
  # note: `n` is an input but not used (don't worry about this hack)
  x <- seq(0, 1, length = 40)
  y <- 1 + 1.2 * x + rnorm(40, sd = 2)
  t_star <- qt(0.975, 38)
  lse <- lm(y ~ x)
  lse_out <- summary(lse)$coefficients
  se <- lse_out[2, 2]
  b1 <- lse_out[2, 1]
  upper_CI = b1 + t_star * se
  lower_CI = b1 - t_star * se
  return(data.frame(se, b1, upper_CI, lower_CI))
}

# this step runs the simulation 100 times,
# then matrix transposes the result so rows are observations
sim_results <- data.frame(t(sapply(X = 1:100, FUN = simulate_lm)))
```