STAT 5014: Homework Three

Bobby Soule

9/18/2017

Problem 4

In the lecture, there were two links to programming style guides. What is your takeaway from this and what specifically are *you* going to do to improve your coding style?

I already knew many of these guidelines from previous experience. The one rule I have never followed, however, is the 80 character line limit. I tend to not bother messsing with line length unless I have lines that go beyond the edge of my screen, but this is something I should work on in the future.

Another thing I need to work on is having concise, descriptive object names. I tend to just create names as I write my code and I usually end up with a whole mess of object names when I am finished. To help with this, it will be beneficial for me to think about what objects I will need and what they should be named before I start to writing code.

I found it interesting that one of the style guides advocated for the use of dots in variable names while the other suggested underscores. I tend to use underscores since that is the convention in other languages (ie. Python).

Problem 5

Good programming practices start with this homework. In the last homework, you imported, munged, cleaned and summarized datasets from Wu and Hamada's *Experiments: Planning, Design and Analysis*. In this problem, please using *lintr* to lint your last homework. From the messages, what are some things you need to change in your code?

Based off of the comments, the biggest fix I need to make is adding spaces around infix operators and after all commas. The second most common comment was that I had a few lines that were longer than 80 characters. There were also a couple of spots where I used single quotes rather than double quotes.

Problem 6

A situation you may encounter is a data set where you need to create a summary statistic for each observation type. Sometimes, this type of redundancy is perfect for a function. Here, we need to create a single function to:

1. calculate the mean for dev1
2. calculate the mean for dev2
3. calculate the sd for dev1
4. calculate the sd for dev2
5. calculate the correlation between dev1 and dev2
6. return the above as a single data.frame

The function has only one parameter: the dataset. First, the function stores the unique values of "Observer"" and all values of "dev1" and "dev2". Next, it creates an empty dataframe that will hold the summary statistics for each observer. Using a for loop, the function iterates through each observer and calculates the relevant summary statistics. Before outputing the data, the function uses functions from tidyr to turn the summary dataset into a tidy dataset.

```
summ_func <- function(data) {
  observers <- sort(unique(data[, 1]))
  dev1 <- data[, 2]
  dev2 <- data[, 3]

  summ_data <- data.frame(matrix(NA, nrow = length(observers), ncol = 6))
  colnames(summ_data) <- c("obs", "mean1", "mean2", "sd1", "sd2", "cor")

  for (observer in observers) {
    rows <- dev_data$Observer == observer
    summ_data[observer, 1] <- observer
    summ_data[observer, 2] <- mean(dev1[rows])
    summ_data[observer, 3] <- mean(dev2[rows])
    summ_data[observer, 4] <- sd(dev1[rows])
    summ_data[observer, 5] <- sd(dev2[rows])
    summ_data[observer, 6] <- cor(dev1[rows], dev2[rows])
  }

  means <- summ_data %>%
    select(obs:mean2) %>%
    gather(key = device, value = mean, -obs) %>%
    mutate(device = gsub("mean", "", device)) %>%
    arrange(obs, device) %>%
    mutate(obs = as.factor(obs), device = as.factor(device))

  sds <- summ_data %>%
    select(obs, sd1, sd2, cor) %>%
    gather(key = device, value = sd, sd1:sd2) %>%
    mutate(device = gsub("sd", "", device)) %>%
    select(obs, device, sd, cor) %>%
    arrange(obs, device)

  cbind(means, sds[, 3:4])
}

dev_data <- readRDS("HW3_data.rds")
dev_summ <- summ_func(dev_data)
```

We will use this function to summarize a dataset which has multiple repeated measurements from two devices by thirteen observers. The output of this problem should be:

a. A single table of the means, sd, and correlation for each of the 13 Observers
b. A box plot of all the means to compare the spread of means from dev1 to dev2
c. A violin plot of all the sd to compare the spread of sd from dev1 to dev2

Part A

The summary dataset was formatted to be "tidy", so it is longer than 13 observations. Creating a tidy dateset made it much easier to create the necessary graphics using *ggplot2*.

```
knitr::kable(dev_summ, caption="Summary Statistics for 13 Observers")
```

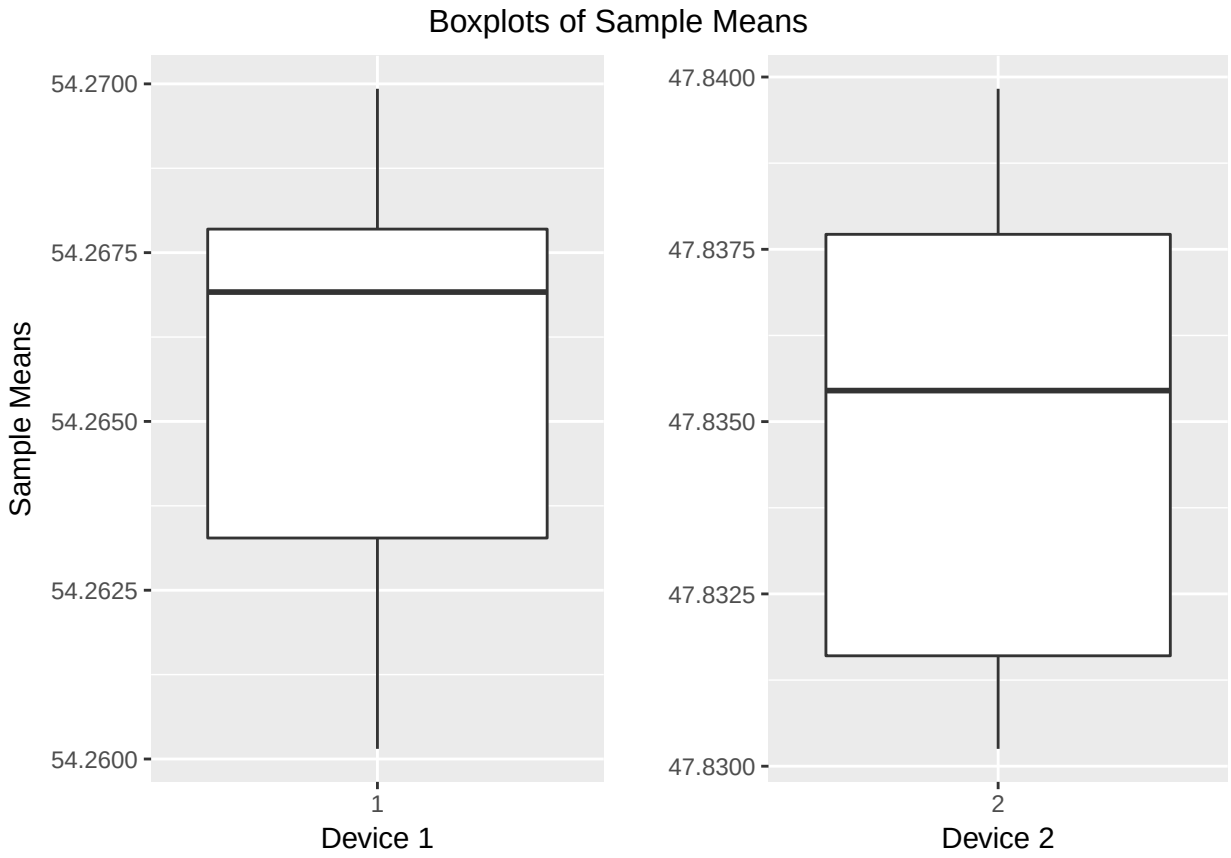
Table 1: Summary Statistics for 13 Observers

obs	device	mean	sd	cor
1	1	54.26610	16.76983	-0.0641284
1	2	47.83472	26.93974	-0.0641284
2	1	54.26873	16.76924	-0.0685864
2	2	47.83082	26.93573	-0.0685864
3	1	54.26732	16.76001	-0.0683434
3	2	47.83772	26.93004	-0.0683434
4	1	54.26327	16.76514	-0.0644719
4	2	47.83225	26.93540	-0.0644719
5	1	54.26030	16.76774	-0.0603414
5	2	47.83983	26.93019	-0.0603414
6	1	54.26144	16.76590	-0.0617148
6	2	47.83025	26.93988	-0.0617148
7	1	54.26881	16.76670	-0.0685042
7	2	47.83545	26.94000	-0.0685042
8	1	54.26785	16.76676	-0.0689797
8	2	47.83590	26.93610	-0.0689797
9	1	54.26588	16.76885	-0.0686092
9	2	47.83150	26.93861	-0.0686092
10	1	54.26734	16.76896	-0.0629611
10	2	47.83955	26.93027	-0.0629611
11	1	54.26993	16.76996	-0.0694456
11	2	47.83699	26.93768	-0.0694456
12	1	54.26692	16.77000	-0.0665752
12	2	47.83160	26.93790	-0.0665752
13	1	54.26015	16.76996	-0.0655833
13	2	47.83972	26.93000	-0.0655833

Part B

Since the sample means have such small spread relative to the distance between them, the boxplots appeared as lines when putting them on the same plot. Instead, each plot was created separately and then the two plots were placed next to each other using *grid.arrange* from the *gridExtra* package. Formatting the graphs this way makes it harder to directly compare the spread of the two samples, but this is the only workaround to the boxplots showing up as lines that I found.

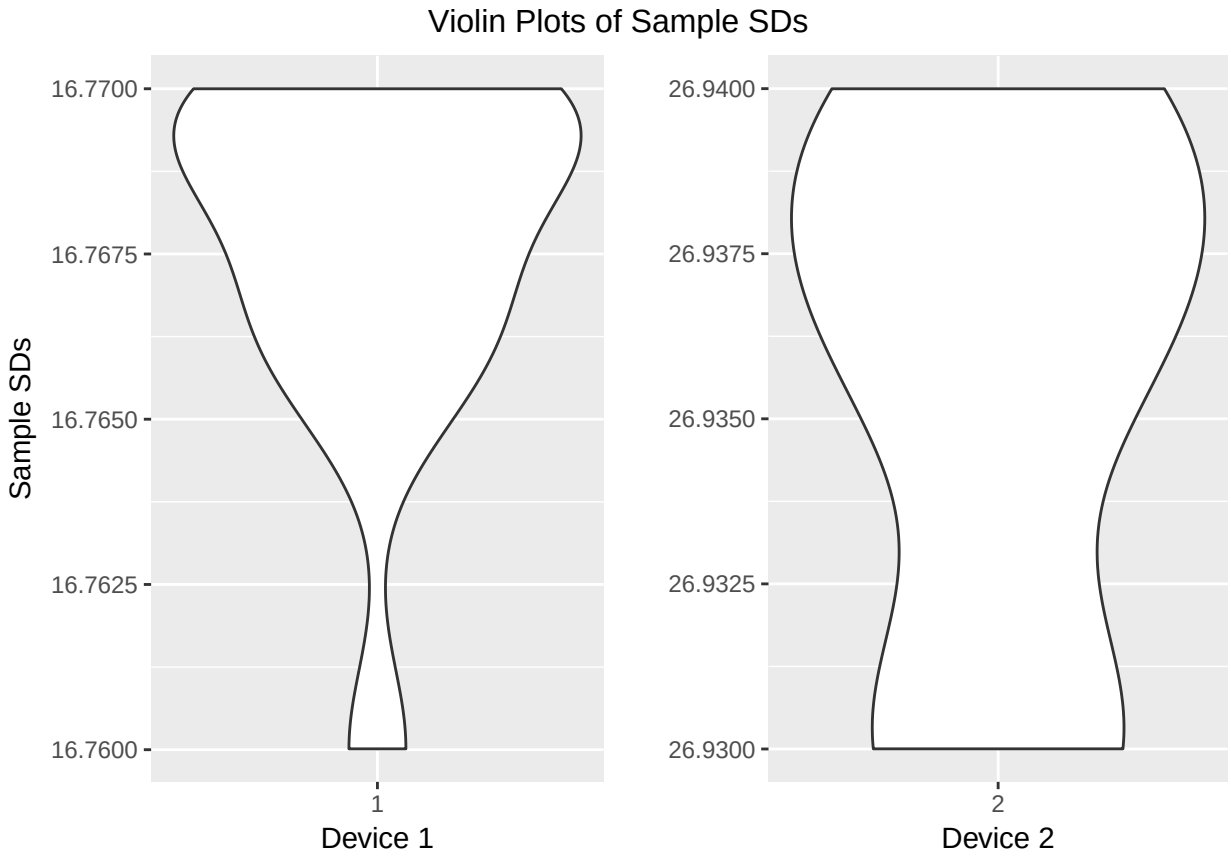
```
box1 <- ggplot(dev_summ[dev_summ$device==1, ], aes(x=device, y=mean)) +
        geom_boxplot() +
        labs(x = "Device 1", y = "Sample Means")
box2 <- ggplot(dev_summ[dev_summ$device==2, ], aes(x=device, y=mean)) +
        geom_boxplot() +
        labs(x = "Device 2", y = "")
grid.arrange(box1, box2, ncol=2, top = "Boxplots of Sample Means")
```



Part C

The same problem from **Part B** was encountered when creating the violin plots of the sample standard deviations, so once again the two violin plots were created on separate plots and then displayed together.

```
viol1 <- ggplot(dev_summ[dev_summ$device==1, ], aes(x=device, y=sd)) +
         geom_violin() +
         labs(x = "Device 1", y = "Sample SDs")
viol2 <- ggplot(dev_summ[dev_summ$device==2, ], aes(x=device, y=sd)) +
         geom_violin() +
         labs(x = "Device 2", y = "")
grid.arrange(viol1, viol2, ncol=2, top = "Violin Plots of Sample SDs")
```



Problem 7 – redo

Same as last time, please create and annotate the process to create a tidy dataset from http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BloodPressure.dat

The first thing done to clean this dataset was removing the repeated "day" variable. The next problem was that the dataset has variables as column names. The categorical "type" variable, which can take on values of "dev" or "doc", and the "reading" variable (1-3) are both being used as column names. The data were gathered, but then the "type" and "reading" variables were in the same column so they were separated. Finally, the needed variables were selected and arranged by "day", "reading", and "type".

```
url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BloodPressure.dat"
bp_data <- read.table(url, header=TRUE, skip=1, fill=TRUE, stringsAsFactors=FALSE) %>%
  tbl_df %>%
  select(-Day.1) %>%
  gather(key = dev_doc, value = BP, -Day) %>%
  mutate(Reading = gsub("Dev|Doc", "", dev_doc)) %>%
  mutate(Type = gsub("1|2|3", "", dev_doc)) %>%
  select(Day, Reading, Type, BP) %>%
  arrange(Day, Reading, Type)

knitr::kable(bp_data, caption="Blood Pressure Readings: Doctors vs Devices")
```

Table 2: Blood Pressure Readings: Doctors vs Devices

Day	Reading	Type	BP
1	1	Dev	133.34
1	1	Doc	126.54
1	2	Dev	133.36
1	2	Doc	127.36
1	3	Dev	133.45
1	3	Doc	131.88
2	1	Dev	110.94
2	1	Doc	124.69
2	2	Dev	110.85
2	2	Doc	128.86
2	3	Dev	110.92
2	3	Doc	132.39
3	1	Dev	118.54
3	1	Doc	125.46
3	2	Dev	118.56
3	2	Doc	129.43
3	3	Dev	118.67
3	3	Doc	134.43
4	1	Dev	137.94
4	1	Doc	125.95
4	2	Dev	137.80
4	2	Doc	130.72
4	3	Dev	137.77
4	3	Doc	134.28
5	1	Dev	139.52
5	1	Doc	125.90
5	2	Dev	139.62
5	2	Doc	130.13
5	3	Dev	139.59
5	3	Doc	134.44
6	1	Dev	139.23
6	1	Doc	127.85
6	2	Dev	139.11
6	2	Doc	132.03
6	3	Dev	139.36
6	3	Doc	137.37
7	1	Dev	117.96
7	1	Doc	125.55
7	2	Dev	117.81
7	2	Doc	132.05
7	3	Dev	117.85
7	3	Doc	132.17
8	1	Dev	119.59
8	1	Doc	125.80
8	2	Dev	119.42
8	2	Doc	129.87
8	3	Dev	119.48
8	3	Doc	134.97
9	1	Dev	116.12
9	1	Doc	125.11

Day	Reading	Type	BP
9	2	Dev	116.00
9	2	Doc	128.09
9	3	Dev	115.93
9	3	Doc	133.97
10	1	Dev	128.38
10	1	Doc	125.75
10	2	Dev	128.48
10	2	Doc	131.94
10	3	Dev	128.41
10	3	Doc	132.68
11	1	Dev	125.17
11	1	Doc	128.77
11	2	Dev	125.25
11	2	Doc	130.05
11	3	Dev	125.34
11	3	Doc	134.75
12	1	Dev	134.62
12	1	Doc	125.26
12	2	Dev	134.41
12	2	Doc	131.13
12	3	Dev	134.55
12	3	Doc	134.29
13	1	Dev	136.14
13	1	Doc	126.26
13	2	Dev	136.07
13	2	Doc	130.91
13	3	Dev	136.22
13	3	Doc	133.38
14	1	Dev	131.21
14	1	Doc	125.68
14	2	Dev	131.03
14	2	Doc	128.83
14	3	Dev	130.96
14	3	Doc	135.67
15	1	Dev	132.51
15	1	Doc	124.47
15	2	Dev	132.86
15	2	Doc	129.46
15	3	Dev	132.65
15	3	Doc	134.39

Problem 8

Create a function to find solutions to (1) using Newton's method. The answer should include the solutions with tolerance used to terminate the loop and a plot showing the iterations on the path to the solution.

$$f(x) = 3^x - sin(x) + cos(5x) \tag{1}$$

The equation for Newton's Method is as follows:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Better approximations are obtained with each successive iteration. Iterations should be continued until the difference between x_n and x_{n-1} is less than a given tolerance, ϵ.

So, the function must have two parameters: an initial point, x_0, and a tolerance, ϵ. The function must also assign $f(x)$ and its derivative. Once all setup is complete, the function will carry out the method within a while loop that continues until the difference is less than the tolerance. At the end of each iteration of the while loop, x_n will be stored in a vector so that the path to the solution can be plotted later.

```
newt_meth <- function(x0, epsilon = 0.01) {
  xn <- x0
  continue <- TRUE

  fx <- function(x) {3^x - sin(x) + cos(5*x)}
  fx_prime <- function(x) {log(3)*(3^x) - cos(x) - 5*sin(5*x)}

  while (continue) {
    curr_x <- xn[length(xn)]
    xn <- c(xn, curr_x - (fx(curr_x) / fx_prime(curr_x)))
    if (abs(xn[length(xn)] - xn[length(xn) - 1]) < epsilon) {continue <- FALSE}
  }
  list(iterations = xn, tolerance = epsilon)
}
```

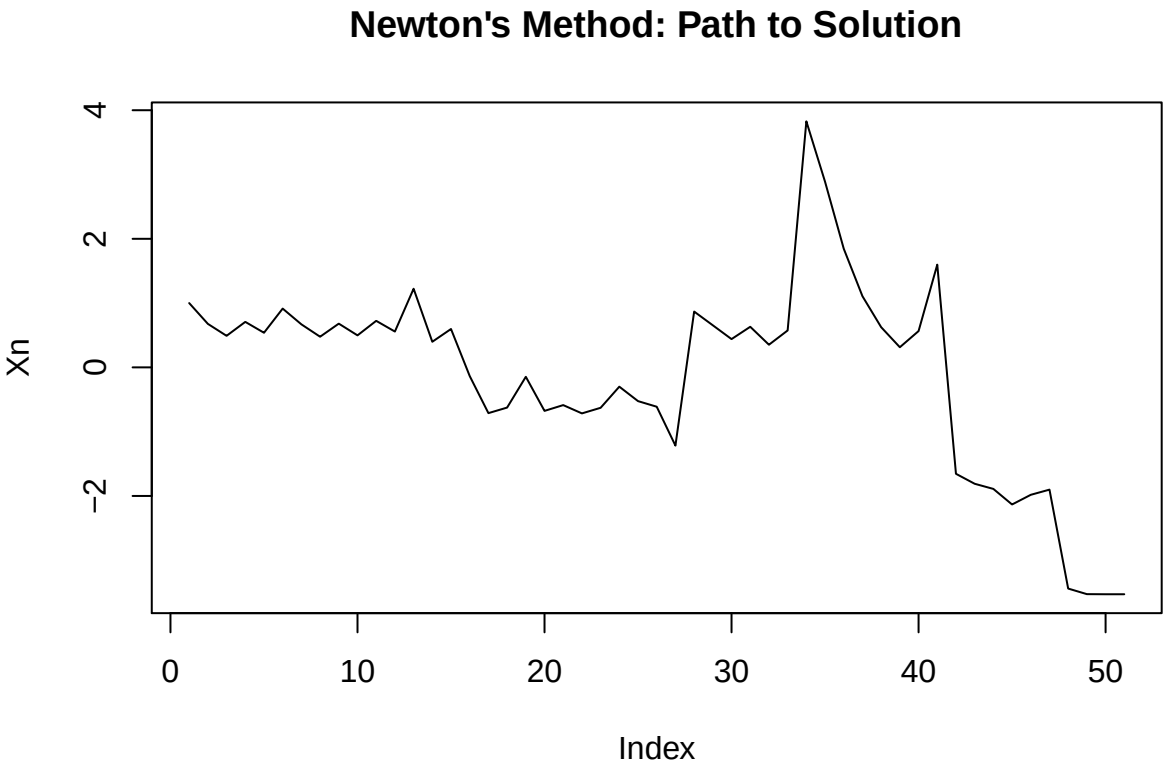
With initial value $x_0 = 1$ and tolerance $\epsilon = 0.001$, Newton's Method finds a solution to the equation at $x = -3.5287$.

```
solution <- newt_meth(x0 = 1, epsilon = 0.001)
solution
```

```
## $iterations
##  [1]  1.0000000  0.6765376  0.4909463  0.7077883  0.5384152  0.9142942
##  [7]  0.6693496  0.4766906  0.6803219  0.4978181  0.7238992  0.5570264
## [13]  1.2232230  0.3982372  0.5971722 -0.1351323 -0.7114928 -0.6253884
## [19] -0.1461600 -0.6766512 -0.5867190 -0.7156108 -0.6288626 -0.3000029
## [25] -0.5250965 -0.6121527 -1.2168610  0.8684422  0.6542222  0.4397971
## [31]  0.6315670  0.3524363  0.5740955  3.8267941  2.8868739  1.8459105
## [37]  1.1066374  0.6250342  0.3133281  0.5648088  1.5979047 -1.6553903
## [43] -1.8105509 -1.8906413 -2.1325650 -1.9819303 -1.9015053 -3.4405071
## [49] -3.5273555 -3.5287207 -3.5287228
##
## $tolerance
## [1] 0.001
```

A plot of the method's path to this solution can be seen below.

```
plot(solution$iterations, type="l",
     main = "Newton's Method: Path to Solution", ylab = "Xn")
```



Problem 9 – redo, make a good honest and professional attempt

One common situation data scientists encounter is when data is spread across many data files. This can be that the data is simply split across data files OR different aspects of the data is in different data files. Here we will look at the second scenario: different aspects of a dataset are contained in different data files that need to be merged. In this case, we are going to munge some open data containing car records, reported defects, and defect descriptions. You should start this problem by looking at the help for variations on SQL like merge functions: merge, join, inner_join, left_join, right_join. As in the last problem, please create a tidy dataset, summarize and annotate the process, and report the indicated statistics.

First, the datasets were loaded into R and then they were filtered so that they only contined rows from 2017. Using full_join, the gebreken and geconstat dataset were merged by license plate and then the result was merged with the personenauto dataset by defect code. Since full_join was used the resulting dataset had many NAs, so any row containing missing data was removed.

```
folder_path <- "/Users/bobbysoule/Documents/College/Graduate/STAT_5014/STAT_5014_homework/02_data_mungi

#defect code, defect comment
Car_Gebreken <- fread(input = paste(folder_path, "Open_Data_RDW__Gebreken.csv", sep = ""),
                      header = T, select=c(1,6), showProgress=F) %>% tbl_df
colnames(Car_Gebreken) <- c("Code", "Comment")
#license plate, inspection date, defect code
Car_Geconstat <- fread(input = paste(folder_path, "Open_Data_RDW__Geconstateerde_Gebreken.csv", sep = "
                       header=T, select=c(1,3,5),showProgress=F) %>% tbl_df
colnames(Car_Geconstat) <- c("Plate", "Inspect_Date", "Code")
Car_Geconstat <- mutate(Car_Geconstat, Inspect_Date = ymd(Inspect_Date)) %>%
                 filter(year(Inspect_Date) == 2017)
#license plate, make, model
```

```
Car_Person <- fread(input = paste(folder_path, "Personenauto_basisdata.csv", sep = ""),
                           header=T, showProgress = F, select = c(1,3,4)) %>% tbl_df
colnames(Car_Person) <- c("Plate", "Make", "Model")

Car_Final <- full_join(x = Car_Geconstat, y = Car_Person, by = "Plate") %>%
             full_join(y = Car_Gebreken, by = "Code")
Car_Final <- Car_Final[complete.cases(Car_Final),]
```

Now we are interested in the number of unique makes and models:

```
unique_make <- n_distinct(Car_Final$Make)
unique_make
```

```
## [1] 503
```

```
unique_model <- n_distinct(Car_Final$Model)
unique_model
```

```
## [1] 33470
```

Next, we would like to create a table of the top five defects and the top makes and models with those defects. First we must extract the top five defects:

```
Car_byCode <- group_by(Car_Final, Code, Comment)
Freq_Defects <- summarize(Car_byCode, Count=n()) %>%
                arrange(desc(Count)) %>%
                head(5)
top5_defects <- Freq_Defects$Code
top5_defects
```

```
## [1] "AC1" "K04" "RA2" "205" "497"
```

Now that we have the top five defects, we will group by defect code, make, and model; and we will use *summarize* to create a table that has the count for each combination of defect, make, and model. We will filter the table so that it only contains the top five make-model combinations within each of the top five defects.

```
Car_byCodeMakeModel <- group_by(Car_Final, Code, Make, Model)
Freq_Defects2 <- summarize(Car_byCodeMakeModel, Count=n()) %>%
                 filter(Code %in% top5_defects) %>%
                 arrange(desc(Count))

for (code in top5_defects) {
  assign(code,
         filter(Freq_Defects2, Code==code) %>%
         arrange(desc(Count)) %>%
         head(5) %>% ungroup %>%
         mutate(Tot_Count = sum(Count))
         )
}

Freq_Defects_Final <- get(top5_defects[1])
for (i in 2:length(top5_defects)) {
  Freq_Defects_Final <- rbind(Freq_Defects_Final, get(top5_defects[i]))
}

tot_counts <- Freq_Defects$Count
```

```
names(tot_counts) <- Freq_Defects$Code
Freq_Defects_Final <- mutate(Freq_Defects_Final, Tot_Count = tot_counts[Code])

knitr::kable(Freq_Defects_Final, caption="Top Five Defects")
```

Table 3: Top Five Defects

Code	Make	Model	Count	Tot_Count
AC1	VOLKSWAGEN	POLO	5744	385621
AC1	VOLKSWAGEN	GOLF	4312	385621
AC1	TOYOTA	TOYOTA AYGO	4103	385621
AC1	PEUGEOT	PEUGEOT 107	3769	385621
AC1	OPEL	CORSA	3516	385621
K04	VOLKSWAGEN	POLO	3389	271259
K04	OPEL	CORSA	3263	271259
K04	RENAULT	TWINGO	2864	271259
K04	RENAULT	CLIO	2656	271259
K04	TOYOTA	TOYOTA AYGO	2613	271259
RA2	PEUGEOT	206; 1.4 3DRS	2913	223142
RA2	FIAT	FIAT PUNTO; 1.2	2518	223142
RA2	OPEL	ASTRA-G-CC; X1.6SZR	2268	223142
RA2	PEUGEOT	206; 1.4 5DRS	2126	223142
RA2	OPEL	CORSA-C; Z1.2XE	2022	223142
205	VOLKSWAGEN	POLO	2474	171244
205	PEUGEOT	PEUGEOT 107	2174	171244
205	TOYOTA	TOYOTA AYGO	2130	171244
205	VOLKSWAGEN	GOLF	1832	171244
205	CITROEN	CITROEN C1	1831	171244
497	OPEL	CORSA	2352	139980
497	VOLKSWAGEN	POLO	1971	139980
497	RENAULT	TWINGO	1800	139980
497	PEUGEOT	PEUGEOT 107	1796	139980
497	RENAULT	MEGANE	1770	139980

The last thing we are interested in is testing for a relationship between the number of defects and the make of the car. Below are the regression coefficients and the ANOVA table.

```
cars_reg <- lm(formula = Count ~ Make, data = Freq_Defects2)
cars_anova <- anova(cars_reg)

knitr::kable(summary(cars_reg)$coef, caption="Regression Coefficients")
```

Table 4: Regression Coefficients

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.0000000	108.3822	0.0092266	0.9926384
MakeA.M.C.	-0.0000001	132.7406	0.0000000	1.0000000
MakeACCUBUILT	0.0000000	132.7406	0.0000000	1.0000000
MakeACM	0.0000000	153.2756	0.0000000	1.0000000
MakeACURA	0.0000000	121.1750	0.0000000	1.0000000
MakeADRIA	0.3870967	110.1165	0.0035153	0.9971952
MakeADRIA MOBIL	1.1444444	108.9827	0.0105012	0.9916215

	Estimate	Std. Error	t value	Pr(>|t|)
MakeAIXAM	0.0000000	132.7406	0.0000000	1.0000000
MakeALFA ROMEO	8.7142857	108.4400	0.0803604	0.9359509
MakeALKO	0.3181818	109.6069	0.0029029	0.9976838
MakeALPINA	0.0000000	111.9367	0.0000000	1.0000000
MakeALPINE-RENAULT	0.3076923	112.4736	0.0027357	0.9978173
MakeALVIS	0.0000000	153.2756	0.0000000	1.0000000
MakeAMC JEEP	0.0000000	125.1490	0.0000000	1.0000000
MakeAMERICAN	0.0000000	153.2756	0.0000000	1.0000000
MakeAMERICAN CUSTOM COACHWORK	0.0000000	153.2756	0.0000000	1.0000000
MakeAMERICAN MOTORS	0.0000000	121.1750	0.0000000	1.0000000
MakeARCA	0.0000000	132.7406	0.0000000	1.0000000
MakeARIEL	0.0000000	132.7406	0.0000000	1.0000000
MakeARMBRUSTER/STAGEWAY	0.0000000	153.2756	0.0000000	1.0000000
MakeASIA	0.0000000	132.7406	0.0000000	1.0000000
MakeASTON-MARTIN	0.7750000	109.7287	0.0070629	0.9943647
MakeAUDI	11.3865058	108.4065	0.1050353	0.9163481
MakeAUDI/PORSCHE	0.0000000	153.2756	0.0000000	1.0000000
MakeAUSTIN	0.2238806	109.1881	0.0020504	0.9983640
MakeAUSTIN-HEALEY	0.0000000	115.8655	0.0000000	1.0000000
MakeAUSTIN-VANDEN PLAS	0.0000000	153.2756	0.0000000	1.0000000
MakeAUTO BIANCHI	0.3846153	112.4736	0.0034196	0.9972716
MakeAUTO UNION	2.0540540	108.5153	0.0189287	0.9848980
MakeAUTOSTAR	0.3000000	113.6723	0.0026392	0.9978943
MakeBALZER/APAL	0.0000000	153.2756	0.0000000	1.0000000
MakeBARTAN	0.0000000	153.2756	0.0000000	1.0000000
MakeBAVARIA	0.0000000	117.0662	0.0000000	1.0000000
MakeBEARDMORE	0.0000000	153.2756	0.0000000	1.0000000
MakeBEAUFORD	0.0000000	153.2756	0.0000000	1.0000000
MakeBEDFORD	0.2222222	110.3711	0.0020134	0.9983935
MakeBEDFORD-HYMERMOBIL	0.5000000	121.1750	0.0041263	0.9967077
MakeBENIMAR	0.1000000	113.6723	0.0008797	0.9992981
MakeBENTLEY	0.3877551	109.4826	0.0035417	0.9971741
MakeBERTONE	0.2000000	118.7268	0.0016845	0.9986559
MakeBINZ	0.7500000	121.1750	0.0061894	0.9950616
MakeBLUE BIRD	0.0000000	153.2756	0.0000000	1.0000000
MakeBMW	12.2687302	108.4013	0.1131788	0.9098892
MakeBMW I	0.0000000	153.2756	0.0000000	1.0000000
MakeBOMBARDIER	0.0000000	153.2756	0.0000000	1.0000000
MakeBOONACKER	0.1428571	115.8655	0.0012330	0.9990162
MakeBORGWARD	0.3333333	125.1490	0.0026635	0.9978749
MakeBOSMAL	1.0000000	125.1490	0.0079905	0.9936246
MakeBOVA	0.0000000	132.7406	0.0000000	1.0000000
MakeBREMACH	0.0000000	132.7406	0.0000000	1.0000000
MakeBUERSTNER	0.8135593	108.8405	0.0074748	0.9940361
MakeBUICK	0.6301369	109.1221	0.0057746	0.9953926
MakeBYD	0.0000000	153.2756	0.0000000	1.0000000
MakeCADILLAC	1.5478723	108.6701	0.0142438	0.9886355
MakeCAPRON	0.4000000	109.9197	0.0036390	0.9970965
MakeCAR CRAFT	0.0000000	153.2756	0.0000000	1.0000000
MakeCARAVANS INTERNAT. SPA	0.0000000	115.8655	0.0000000	1.0000000
MakeCARBODIES	0.0000000	132.7406	0.0000000	1.0000000
MakeCARTHAGO	0.3448275	110.2351	0.0031281	0.9975041

	Estimate	Std. Error	t value	Pr(>|t|)
MakeCATERHAM	0.6250000	114.9567	0.0054368	0.9956621
MakeCBT	0.0000000	115.8655	0.0000000	1.0000000
MakeCHALLANGER	0.0000000	153.2756	0.0000000	1.0000000
MakeCHALLENGER	0.2222222	110.3711	0.0020134	0.9983935
MakeCHAMONIX	0.0000000	153.2756	0.0000000	1.0000000
MakeCHATEAU	0.5000000	121.1750	0.0041263	0.9967077
MakeCHAUSSON	0.4067796	109.2969	0.0037218	0.9970305
MakeCHECKER	0.0000000	153.2756	0.0000000	1.0000000
MakeCHEVROLET	16.4337837	108.4554	0.1515257	0.8795616
MakeCHEVROLET DAEWOO	0.5000000	132.7406	0.0037667	0.9969946
MakeCHEVROLET ITALIA	0.5000000	132.7406	0.0037667	0.9969946
MakeCHEVROLET-CRUISE MASTER	0.0000000	153.2756	0.0000000	1.0000000
MakeCHEVROLET-EDSON	0.0000000	153.2756	0.0000000	1.0000000
MakeCHEVROLET-EL DORADO	0.0000000	153.2756	0.0000000	1.0000000
MakeCHEVROLET-ITASCA	0.3333333	125.1490	0.0026635	0.9978749
MakeCHEVROLET-JAYCO	0.0000000	153.2756	0.0000000	1.0000000
MakeCHEVROLET-LINDY	0.0000000	132.7406	0.0000000	1.0000000
MakeCHEVROLET-SOUTHWIND	0.0000000	153.2756	0.0000000	1.0000000
MakeCHEVROLET-STARCRAFT	0.0000000	132.7406	0.0000000	1.0000000
MakeCHEVROLET-WINNEBAGO	0.0000000	153.2756	0.0000000	1.0000000
MakeCHRYSLER	7.7699805	108.4878	0.0716208	0.9429040
MakeCHRYSLER-SIMCA	0.0000000	153.2756	0.0000000	1.0000000
MakeCITROEN	23.6607284	108.4037	0.2182650	0.8272234
MakeCITY-VAN	0.0000000	153.2756	0.0000000	1.0000000
MakeCMC	0.0000000	153.2756	0.0000000	1.0000000
MakeCOLLINS	0.0000000	153.2756	0.0000000	1.0000000
MakeCOMMER	0.2000000	118.7268	0.0016845	0.9986559
MakeCONCORDE	0.0000000	112.8078	0.0000000	1.0000000
MakeCONCORDE REISEMOBILE GMBH	0.0000000	153.2756	0.0000000	1.0000000
MakeCRISTALL	0.0000000	121.1750	0.0000000	1.0000000
MakeD.K.W.	0.0000000	118.7268	0.0000000	1.0000000
MakeDACIA	85.1081081	109.8371	0.7748575	0.4384270
MakeDAEWOO	19.5009380	108.4839	0.1797589	0.8573425
MakeDAF	0.7205882	109.1763	0.0066002	0.9947338
MakeDAIHATSU	39.9488817	108.5552	0.3680051	0.7128708
MakeDAIMLER	1.0547945	109.1221	0.0096662	0.9922877
MakeDAIMLER BENZ	0.0000000	118.7268	0.0000000	1.0000000
MakeDAIMLERCHRYSLER AG	0.0952381	110.9328	0.0008585	0.9993150
MakeDATSUN	0.1052631	111.1978	0.0009466	0.9992447
MakeDAX	0.0000000	118.7268	0.0000000	1.0000000
MakeDAX COBRA	0.1666666	117.0662	0.0014237	0.9988641
MakeDE LOREAN	0.0000000	153.2756	0.0000000	1.0000000
MakeDE TOMASO	0.0000000	153.2756	0.0000000	1.0000000
MakeDEN OUDSTEN	0.0000000	153.2756	0.0000000	1.0000000
MakeDERCKS	0.7777777	114.2449	0.0068080	0.9945681
MakeDETHLEFFS	0.4456521	108.6764	0.0041007	0.9967281
MakeDODGE	4.1102941	108.7800	0.0377854	0.9698589
MakeDODGE-EDSON	0.0000000	153.2756	0.0000000	1.0000000
MakeDODGE-MIDAS	0.0000000	153.2756	0.0000000	1.0000000
MakeDONKERVOORT(JD)	0.2142857	112.1863	0.0019101	0.9984760
MakeDREAM	0.0000000	132.7406	0.0000000	1.0000000
MakeDUTTON	0.0000000	125.1490	0.0000000	1.0000000

	Estimate	Std. Error	t value	Pr(>|t|)
MakeE.B.S.	0.0000000	153.2756	0.0000000	1.0000000
MakeE.W.B	0.0000000	132.7406	0.0000000	1.0000000
MakeEIGENBOUW	0.0000000	153.2756	0.0000000	1.0000000
MakeELNAGH	0.4545454	110.8181	0.0041017	0.9967273
MakeESTEREL	0.0000000	153.2756	0.0000000	1.0000000
MakeEURA MOBIL	0.0000000	111.7179	0.0000000	1.0000000
MakeEVM	0.0000000	153.2756	0.0000000	1.0000000
MakeEXCALIBUR	0.0000000	121.1750	0.0000000	1.0000000
MakeEXECUTIVE COACH BUILDERS	0.0000000	153.2756	0.0000000	1.0000000
MakeFENDT	0.5000000	132.7406	0.0037667	0.9969946
MakeFERRARI	0.4000000	109.1024	0.0036663	0.9970748
MakeFIAT	27.8287139	108.4123	0.2566934	0.7974164
MakeFIAT/CHALLENGER	1.0000000	153.2756	0.0065242	0.9947945
MakeFISKER	1.5000000	132.7406	0.0113002	0.9909839
MakeFLEETWOOD	0.0000000	125.1490	0.0000000	1.0000000
MakeFLEURETTE	0.0000000	121.1750	0.0000000	1.0000000
MakeFORD	32.8904854	108.4033	0.3034086	0.7615796
MakeFORD-CASUAL	0.0000000	153.2756	0.0000000	1.0000000
MakeFORD-CNG-TECHNIK	3.6666666	114.2449	0.0320948	0.9743966
MakeFORD-RAMBLER	1.0000000	153.2756	0.0065242	0.9947945
MakeFRANKIA	0.1428571	112.1863	0.0012734	0.9989840
MakeFSM	0.3333333	125.1490	0.0026635	0.9978749
MakeG.M.C.	0.3466666	109.1024	0.0031774	0.9974648
MakeGALLOPER	0.0000000	132.7406	0.0000000	1.0000000
MakeGARDNER DOUGLAS	0.0000000	132.7406	0.0000000	1.0000000
MakeGAZ	0.0000000	132.7406	0.0000000	1.0000000
MakeGENERAL MOTORS-GMC	0.0000000	153.2756	0.0000000	1.0000000
MakeGEO	0.5000000	121.1750	0.0041263	0.9967077
MakeGINETTA	0.0000000	153.2756	0.0000000	1.0000000
MakeGIOTTILINE	0.3333333	117.0662	0.0028474	0.9977281
MakeGM DAEWOO	0.0000000	118.7268	0.0000000	1.0000000
MakeGMC	0.0000000	153.2756	0.0000000	1.0000000
MakeGMC-EDSON	0.2000000	118.7268	0.0016845	0.9986559
MakeGMC-MIDAS	0.6000000	113.6723	0.0052783	0.9957885
MakeGMC-TIOGA	0.0000000	153.2756	0.0000000	1.0000000
MakeGOGGOMOBIL	0.0000000	153.2756	0.0000000	1.0000000
MakeGORGUS	0.0000000	153.2756	0.0000000	1.0000000
MakeGURGEL	0.0000000	153.2756	0.0000000	1.0000000
MakeHAMMOND & THIEDE	0.0000000	132.7406	0.0000000	1.0000000
MakeHANOMAG	0.0000000	153.2756	0.0000000	1.0000000
MakeHANOMAG-HENSCHEL	0.0909090	110.8181	0.0008203	0.9993455
MakeHEALEY	0.0000000	153.2756	0.0000000	1.0000000
MakeHESS EN EISENHARDT	0.0000000	132.7406	0.0000000	1.0000000
MakeHILLMAN	0.0000000	153.2756	0.0000000	1.0000000
MakeHINDUSTAN	0.0000000	153.2756	0.0000000	1.0000000
MakeHOBBY	0.5833333	110.6172	0.0052734	0.9957924
MakeHOME CAR	0.0000000	115.8655	0.0000000	1.0000000
MakeHONDA	8.9554140	108.4315	0.0825905	0.9341774
MakeHOTCHKISS	0.0000000	132.7406	0.0000000	1.0000000
MakeHUISKAMP	0.0000000	125.1490	0.0000000	1.0000000
MakeHUMBER	0.0000000	125.1490	0.0000000	1.0000000
MakeHUMMER	0.3750000	111.7179	0.0033567	0.9973218

	Estimate	Std. Error	t value	Pr(>|t|)
MakeHYMER	0.7638888	108.6328	0.0070318	0.9943895
MakeHYUNDAI	39.9568862	108.4471	0.3684458	0.7125422
MakeI.V.R.	0.0000000	125.1490	0.0000000	1.0000000
MakeIAV	1.5000000	121.1750	0.0123788	0.9901235
MakeIMPERIAL	0.0000000	153.2756	0.0000000	1.0000000
MakeINFINITI	0.8863636	109.6069	0.0080867	0.9935478
MakeINNOCENTI	0.0000000	153.2756	0.0000000	1.0000000
MakeINTERNATIONAL	0.1428571	115.8655	0.0012330	0.9990162
MakeISUZU	0.5714285	115.8655	0.0049318	0.9960650
MakeITINEO	0.0000000	125.1490	0.0000000	1.0000000
MakeIVECO	0.2788461	108.9021	0.0025605	0.9979570
MakeIVECO FIAT	0.0000000	153.2756	0.0000000	1.0000000
MakeJAGO	0.0000000	153.2756	0.0000000	1.0000000
MakeJAGUAR	2.5905660	108.4844	0.0238796	0.9809487
MakeJAGUAR CARS	6.0344827	109.0033	0.0553605	0.9558514
MakeJAGUAR LAND ROVER	0.5000000	114.9567	0.0043495	0.9965297
MakeJCSPORTCAR	0.0000000	153.2756	0.0000000	1.0000000
MakeJEEP	3.7251308	108.5240	0.0343254	0.9726178
MakeJENSEN	0.6666666	117.0662	0.0056948	0.9954563
MakeJIANGLING	0.0000000	153.2756	0.0000000	1.0000000
MakeJOINT	0.1052631	111.1978	0.0009466	0.9992447
MakeKAISER	0.3333333	125.1490	0.0026635	0.9978749
MakeKARMANN MOBIL	0.5294117	111.5244	0.0047470	0.9962124
MakeKIA	47.8233995	108.5018	0.4407613	0.6593874
MakeKIA MOTORS	0.1250000	114.9567	0.0010874	0.9991324
MakeKIAMASTER	0.2500000	121.1750	0.0020631	0.9983539
MakeKNAUS	0.8235294	109.1763	0.0075431	0.9939815
MakeKNAUS TABBERT	0.2222222	114.2449	0.0019451	0.9984480
MakeKTM	0.0000000	153.2756	0.0000000	1.0000000
MakeL.T.I.CARBODIES	0.0000000	132.7406	0.0000000	1.0000000
MakeLA STRADA	2.0000000	132.7406	0.0150670	0.9879788
MakeLADA	0.2857142	109.9197	0.0025993	0.9979261
MakeLADA-VAZ	0.6250000	111.7179	0.0055944	0.9955363
MakeLAIKA	0.1111111	114.2449	0.0009726	0.9992240
MakeLAIKA CARAVANS S.P.A.	0.0000000	115.8655	0.0000000	1.0000000
MakeLAMBORGHINI	0.0000000	114.9567	0.0000000	1.0000000
MakeLANCIA	4.2040816	108.5401	0.0387330	0.9691034
MakeLAND ROVER	5.8561643	108.5059	0.0539709	0.9569585
MakeLANDWIND	1.4000000	118.7268	0.0117918	0.9905918
MakeLDV	0.0000000	153.2756	0.0000000	1.0000000
MakeLE VOYAGEUR	0.0000000	132.7406	0.0000000	1.0000000
MakeLEXUS	6.1700680	108.7503	0.0567361	0.9547556
MakeLEYLAND	0.0000000	125.1490	0.0000000	1.0000000
MakeLEYLAND-DEN OUDSTEN	0.0000000	153.2756	0.0000000	1.0000000
MakeLINCOLN	1.1836734	109.4826	0.0108115	0.9913739
MakeLINER	0.0000000	153.2756	0.0000000	1.0000000
MakeLMC	0.1315789	109.7991	0.0011984	0.9990439
MakeLOCOST	0.0000000	153.2756	0.0000000	1.0000000
MakeLOCUST	0.0000000	153.2756	0.0000000	1.0000000
MakeLOMAX	0.0000000	132.7406	0.0000000	1.0000000
MakeLOTUS	0.4509803	109.4397	0.0041208	0.9967121
MakeLTI VEHICLES	0.0000000	153.2756	0.0000000	1.0000000

	Estimate	Std. Error	t value	Pr(>|t|)
MakeM.A.N.	0.0000000	153.2756	0.0000000	1.0000000
MakeMACK	0.0000000	132.7406	0.0000000	1.0000000
MakeMAHINDRA	0.0000000	132.7406	0.0000000	1.0000000
MakeMAN	0.0000000	153.2756	0.0000000	1.0000000
MakeMARLIN	0.0000000	125.1490	0.0000000	1.0000000
MakeMARUTI	0.6666666	125.1490	0.0053270	0.9957497
MakeMASERATI	0.6428571	109.3457	0.0058791	0.9953092
MakeMATRA	0.3333333	114.2449	0.0029177	0.9976720
MakeMATRA SPORTS	0.0000000	153.2756	0.0000000	1.0000000
MakeMAYBACH	0.0000000	153.2756	0.0000000	1.0000000
MakeMAZDA	12.9799857	108.4210	0.1197184	0.9047066
MakeMC LOUIS	0.3333333	117.0662	0.0028474	0.9977281
MakeMC LOUIS SPA	0.3809523	110.9328	0.0034341	0.9972600
MakeMEGA	0.0000000	132.7406	0.0000000	1.0000000
MakeMERCEDES-AMG	0.2666666	111.9367	0.0023823	0.9980992
MakeMERCEDES-BENZ	8.0288888	108.3973	0.0740691	0.9409557
MakeMERCEDES-HYMER	0.0000000	132.7406	0.0000000	1.0000000
MakeMERCEDES-RAPIDO	0.0000000	153.2756	0.0000000	1.0000000
MakeMERCURY	0.3095238	109.6649	0.0028225	0.9977480
MakeMG	3.6591928	108.6250	0.0336865	0.9731273
MakeMICRO COMPACT CAR	38.1666666	117.0662	0.3260263	0.7444056
MakeMICRO COMPACT CAR SMART	224.1666666	117.0662	1.9148709	0.0555138
MakeMIDAS	0.0000000	132.7406	0.0000000	1.0000000
MakeMIDDLEBRIDGE SCIMITAR	0.0000000	153.2756	0.0000000	1.0000000
MakeMIDSTATES	0.0000000	153.2756	0.0000000	1.0000000
MakeMINERVA	0.0000000	132.7406	0.0000000	1.0000000
MakeMINI	20.1645569	108.5193	0.1858153	0.8525902
MakeMITSUBISHI	11.8111195	108.4235	0.1089350	0.9132544
MakeMK INDY	0.0000000	153.2756	0.0000000	1.0000000
MakeMOBIL PARTNER	0.0000000	121.1750	0.0000000	1.0000000
MakeMOBILVETTA	0.2000000	113.6723	0.0017594	0.9985962
MakeMOHR	0.0000000	125.1490	0.0000000	1.0000000
MakeMOKE	0.0000000	153.2756	0.0000000	1.0000000
MakeMOLONEY	0.0000000	153.2756	0.0000000	1.0000000
MakeMONCAYO	0.2857142	110.9328	0.0025756	0.9979450
MakeMOOVEO	0.0000000	153.2756	0.0000000	1.0000000
MakeMORGAN	0.6571428	109.9197	0.0059784	0.9952300
MakeMORRIS	0.6666666	110.9328	0.0060096	0.9952050
MakeMOSKVITCH	0.0000000	132.7406	0.0000000	1.0000000
MakeMOTO GUZZI	0.0000000	153.2756	0.0000000	1.0000000
MakeMOWAG	0.0000000	153.2756	0.0000000	1.0000000
MakeN.S.U.	0.4285714	115.8655	0.0036989	0.9970487
MakeNEKAF	0.4000000	111.9367	0.0035734	0.9971488
MakeNIESMANN-BISCHOFF	0.3076923	112.4736	0.0027357	0.9978173
MakeNILSSON	0.0000000	125.1490	0.0000000	1.0000000
MakeNISSAN	21.6637168	108.4302	0.1997941	0.8416423
MakeOLDSMOBILE	0.3500000	109.7287	0.0031897	0.9974550
MakeOPEL	30.7930944	108.3964	0.2840785	0.7763512
MakeOVERLAND	0.0000000	153.2756	0.0000000	1.0000000
MakePANHARD	0.0000000	132.7406	0.0000000	1.0000000
MakePANTHER	0.0000000	132.7406	0.0000000	1.0000000
MakePEUGEOT	34.8020710	108.3983	0.3210574	0.7481680

	Estimate	Std. Error	t value	Pr(>|t|)
MakePIAGGIO	0.0000000	153.2756	0.0000000	1.0000000
MakePILGRIM	0.0000000	125.1490	0.0000000	1.0000000
MakePILOTE	0.0000000	111.9367	0.0000000	1.0000000
MakePININFARINA	1.0000000	153.2756	0.0065242	0.9947945
MakePLA	0.0000000	153.2756	0.0000000	1.0000000
MakePLYMOUTH	0.2000000	111.9367	0.0017867	0.9985744
MakePOESSL	1.0344827	110.2351	0.0093843	0.9925125
MakePOLSKI FIAT	0.0000000	118.7268	0.0000000	1.0000000
MakePONTIAC	0.6380952	108.8971	0.0058596	0.9953248
MakePORSCHE	2.3759750	108.4668	0.0219051	0.9825237
MakePROTON	0.0000000	132.7406	0.0000000	1.0000000
MakePUCH	0.3333333	125.1490	0.0026635	0.9978749
MakeQUANTUM	0.0000000	153.2756	0.0000000	1.0000000
MakeQUATTRO	1.0909090	110.0121	0.0099163	0.9920881
MakeRAM	0.0000000	153.2756	0.0000000	1.0000000
MakeRANGER	0.0000000	153.2756	0.0000000	1.0000000
MakeRANGEROVER	0.0000000	132.7406	0.0000000	1.0000000
MakeRAPIDO	0.3076923	109.7630	0.0028032	0.9977633
MakeRELIANT	0.0000000	115.8655	0.0000000	1.0000000
MakeREMETZ	0.1250000	114.9567	0.0010874	0.9991324
MakeRENAULT	27.9816694	108.3975	0.2581394	0.7963003
MakeRENAULT SPORT	0.0000000	153.2756	0.0000000	1.0000000
MakeRICKMAN	0.0000000	153.2756	0.0000000	1.0000000
MakeRILEY	0.0000000	153.2756	0.0000000	1.0000000
MakeRIMOR	0.2857142	112.1863	0.0025468	0.9979680
MakeRIMOR S.P.A.	0.3571428	110.3007	0.0032379	0.9974165
MakeROBIN HOOD	0.0000000	125.1490	0.0000000	1.0000000
MakeROLLER TEAM	0.1000000	113.6723	0.0008797	0.9992981
MakeROLLS ROYCE	1.3750000	110.0627	0.0124929	0.9900324
MakeROVER	3.0019455	108.4876	0.0276709	0.9779248
MakeS&S COACH	0.0000000	125.1490	0.0000000	1.0000000
MakeSAAB	6.8522935	108.4319	0.0631944	0.9496119
MakeSACHSENRING	0.0000000	125.1490	0.0000000	1.0000000
MakeSAM	0.0000000	153.2756	0.0000000	1.0000000
MakeSANTANA	0.0000000	125.1490	0.0000000	1.0000000
MakeSAURER	0.0000000	153.2756	0.0000000	1.0000000
MakeSCANIA	0.0000000	153.2756	0.0000000	1.0000000
MakeSCHULZ	0.0000000	153.2756	0.0000000	1.0000000
MakeSCION	0.0000000	153.2756	0.0000000	1.0000000
MakeSEA	0.1333333	111.9367	0.0011911	0.9990496
MakeSEAT	20.8391105	108.4177	0.1922114	0.8475773
MakeSIATA	0.0000000	153.2756	0.0000000	1.0000000
MakeSIMCA	0.0833333	112.8078	0.0007387	0.9994106
MakeSKODA	26.3919860	108.4766	0.2432966	0.8077765
MakeSMART	14.7079646	108.8608	0.1351081	0.8925269
MakeSPARTAN	0.0000000	153.2756	0.0000000	1.0000000
MakeSSANGYONG	2.5000000	108.9338	0.0229497	0.9816905
MakeSTERKENS	0.0000000	153.2756	0.0000000	1.0000000
MakeSTEYR D PUCH	0.0000000	153.2756	0.0000000	1.0000000
MakeSTEYR-PUCH	0.0000000	132.7406	0.0000000	1.0000000
MakeSTIMULA	0.0000000	153.2756	0.0000000	1.0000000
MakeSTUART TAYLOR	0.0000000	153.2756	0.0000000	1.0000000

	Estimate	Std. Error	t value	Pr(>|t|)
MakeSTUDEBAKER	0.0000000	125.1490	0.0000000	1.0000000
MakeSUBARU	6.4351464	108.4956	0.0593125	0.9527034
MakeSUN LIVING	0.0000000	125.1490	0.0000000	1.0000000
MakeSUNBEAM	0.7142857	115.8655	0.0061648	0.9950813
MakeSUPER SEVEN	0.0000000	153.2756	0.0000000	1.0000000
MakeSUPERFORMANCE	0.0000000	153.2756	0.0000000	1.0000000
MakeSUPERIOR	0.0000000	132.7406	0.0000000	1.0000000
MakeSUZUKI	29.6867891	108.4296	0.2737885	0.7842481
MakeSWIFT	0.0000000	153.2756	0.0000000	1.0000000
MakeSYLVA	0.0000000	153.2756	0.0000000	1.0000000
MakeT AND J	0.0000000	153.2756	0.0000000	1.0000000
MakeTALBOT	0.0000000	132.7406	0.0000000	1.0000000
MakeTATRA	0.0000000	121.1750	0.0000000	1.0000000
MakeTEAL	0.0000000	153.2756	0.0000000	1.0000000
MakeTEC	0.0833333	110.6172	0.0007533	0.9993989
MakeTESLA	0.7500000	121.1750	0.0061894	0.9950616
MakeTESLA MOTORS	7.3333333	125.1490	0.0585968	0.9532735
MakeTHOR	0.0000000	153.2756	0.0000000	1.0000000
MakeTIFFANY COACHWORKS	0.0000000	153.2756	0.0000000	1.0000000
MakeTIFFANY ENTERPRISES	0.0000000	153.2756	0.0000000	1.0000000
MakeTIGER	0.0000000	153.2756	0.0000000	1.0000000
MakeTOYOTA	24.4682472	108.4052	0.2257110	0.8214270
MakeTOYOTA-CHINOOK	0.0000000	153.2756	0.0000000	1.0000000
MakeTRABANT	0.9473684	109.7991	0.0086282	0.9931158
MakeTRANSEUROP ENGINEERING	0.0000000	153.2756	0.0000000	1.0000000
MakeTRIGANO	0.1750000	109.7287	0.0015948	0.9987275
MakeTRIUMPH	2.2037037	108.8829	0.0202392	0.9838526
MakeTURRI	0.0000000	153.2756	0.0000000	1.0000000
MakeTURRI & BOARI	0.0000000	153.2756	0.0000000	1.0000000
MakeTVR	0.3529411	109.9645	0.0032096	0.9974391
MakeUAZ	0.0000000	153.2756	0.0000000	1.0000000
MakeUNIMOG	0.0000000	153.2756	0.0000000	1.0000000
MakeVALIANT	0.0000000	132.7406	0.0000000	1.0000000
MakeVAN HOOL	0.0000000	153.2756	0.0000000	1.0000000
MakeVANDEN PLAS PRINCESS	0.0000000	153.2756	0.0000000	1.0000000
MakeVAUXHALL	0.1250000	110.6172	0.0011300	0.9990984
MakeVEB	0.0000000	153.2756	0.0000000	1.0000000
MakeVIELHAUSER MENSCH VM	0.0000000	153.2756	0.0000000	1.0000000
MakeVINTAGE	0.0000000	153.2756	0.0000000	1.0000000
MakeVM	0.1428571	115.8655	0.0012330	0.9990162
MakeVOLGA	0.0000000	125.1490	0.0000000	1.0000000
MakeVOLKSWAGEN	23.9350547	108.3928	0.2208177	0.8252352
MakeVOLKSWAGEN-TABBERT	0.0000000	153.2756	0.0000000	1.0000000
MakeVOLVO	15.7701628	108.4028	0.1454775	0.8843344
MakeVW	0.7368421	111.1978	0.0066264	0.9947130
MakeVW-PORSCHE	0.4545454	113.2016	0.0040154	0.9967962
MakeWARTBURG	0.1428571	115.8655	0.0012330	0.9990162
MakeWEINSBERG	0.2000000	118.7268	0.0016845	0.9986559
MakeWESTFALIA	0.0000000	132.7406	0.0000000	1.0000000
MakeWESTFALIA VAN CONVERSION	0.5000000	132.7406	0.0037667	0.9969946
MakeWESTFIELD	0.0714285	112.1863	0.0006367	0.9994920
MakeWIESMANN	0.0000000	121.1750	0.0000000	1.0000000

	Estimate	Std. Error	t value	Pr(>|t|)
MakeWILLYS	0.7142857	115.8655	0.0061648	0.9950813
MakeWINNEBAGO	0.0000000	132.7406	0.0000000	1.0000000
MakeWOLSELEY	0.0000000	121.1750	0.0000000	1.0000000
MakeYUGO	0.0000000	153.2756	0.0000000	1.0000000
MakeZASTAVA	0.3333333	125.1490	0.0026635	0.9978749
MakeZIE BIJZONDERHEDEN	0.0000000	125.1490	0.0000000	1.0000000

```
knitr::kable(cars_anova, caption="ANOVA Table")
```

Table 5: ANOVA Table

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Make	376	7633474	20301.79	1.728296	0
Residuals	58493	687100346	11746.71	NA	NA

Note: I did not test for a relationship between number of defects and model since there are several thousand models in the dataset.