


STA238 - Winter 2021

Assignment 3

GROUP NUMBER: 28 - Wei-Han Wang, 

March 5, 2021

Part 1

Step 1 (Mathematical Justification)

We assume that X_1, \dots, X_n are independent and identically distributed random variables in the Poisson distribution.

The question mentions two types of distribution: *poisson* and *exponential*.

Use the Bayesian inference formula to find the Posterior Distribution.

$$P(\lambda|data) = \frac{P(data|\lambda)P(\lambda)}{P(data)}$$

λ is the parameter of interest in the question, is the exponential distribution. From slides of week 5, the probability function of exponential distribution is: $f(\lambda) = \frac{1}{\beta} e^{-\frac{\lambda}{\beta}}$

Substitute data for the given poisson distribution $X_1, \dots, X_n \stackrel{iid}{\sim} \text{Poisson}(\lambda)$; $P(\lambda)$ for exponential probability function since it represents the prior distribution of λ .

By the formula of Bayes' rules, we have

$$P(\lambda|X_1, \dots, X_n) = \frac{P(X_1, \dots, X_n|\lambda)P(\lambda)}{P(X_1, \dots, X_n)}$$

Now substitute the probability functions of *poisson* and *exponential* distributions into corresponding variables in the Bayes' rules formula above.

$$P(\lambda|X_1, \dots, X_n) = \frac{(e^{-\lambda} \lambda^{X_1} / X_1! \cdot e^{-\lambda} \lambda^{X_2} / X_2! \cdots e^{-\lambda} \lambda^{X_n} / X_n!)(\frac{1}{\beta} e^{-\frac{\lambda}{\beta}})}{P(X_1, X_2, \dots, X_n)}$$

Since we are **only** interested in λ , we can treat other variables without λ involved as a large constant C . In other words, pull out $X_1!, X_2!, \dots, X_n!, \frac{1}{\beta}, P(X_1, X_2, \dots, X_n) \Rightarrow$ re-expressed as a constant C

The equation will now become:

$$\begin{aligned} P(\lambda|X_1, \dots, X_n) &= C[(e^{-\lambda} \lambda^{X_1}, e^{-\lambda} \lambda^{X_2}, \dots, e^{-\lambda} \lambda^{X_n})(e^{-\frac{\lambda}{\beta}})] \\ &= C[e^{-n\lambda} \lambda^{X_n} e^{-\frac{\lambda}{\beta}}] \\ &= C e^{-n\lambda - \frac{\lambda}{\beta}} \lambda^n \\ &= C[e^{-n\lambda - \frac{\lambda}{\beta}} \lambda^n] \end{aligned}$$

We would like to show this function we've got above in terms of λ . Go through the table of probability functions of distribution in week 5 slides. We want a function that has similar layout as our result with e to the power of some number and λ to the power of some other number. The function that meets the requirement is the *gamma distribution*.

$$f(\lambda) = \frac{1}{\Gamma(\alpha)\beta^\alpha} e^{-\frac{\lambda}{\beta}} \lambda^{\alpha-1}$$

From that, we can see

$$\frac{\lambda}{\beta} = -\lambda(n + \frac{1}{\beta}) \quad \alpha - 1 = n$$

Isolate α and β to substitute them into the gamma distribution.

$$\alpha = n + 1 \quad \beta = (n + \frac{1}{\beta})^{-1}$$

λ is actually a gamma distribution. $\lambda \sim \text{Gamma}(\alpha = n + 1, \beta = (n + \frac{1}{\beta})^{-1})$

$$f(\lambda) = \frac{1}{\Gamma(n + 1)(n + \frac{1}{\beta})^{-n-1}} e^{-\lambda(n + \frac{1}{\beta})} \lambda^n$$

n represents numbers of data. β represents mean of the parameter. λ represents sample mean.

Step 2 (Simulation Justification)

Step 1- Random Sampling: Using 'rpois()', we can generate random poisson variables. We generate the numbers depending on the sample size. For first plot, we generate 10 random poisson variables. For second, we generate 100 random poisson variables.

Step 2- Calculate Prior & Posterior Distribution: Through previous section, we know the distribution for posterior is gamma distribution. We want to set the range of our x values from 0 to 30 by using `data.frame(x=c(0,30))`. After, we use `stat_function()` to calculate our three sets of poisson random variables using our posterior distribution function (which is gamma). We directly calculate the exponential distribution function of our prior distribution also under the same x value range.

Step 3- Plot: We combine the codes from previous step and organize it under `ggplot()` to graph all of our results onto one density plot.

Step 4- Repeat: After we've conducted one set of simulation using first sample size and first three sets of poisson random variables. We pick a new large sample size and draw another three sets of poisson random variables and repeat step 2.

```
# Mean parameter
beta <- 20
# Randomly sample from Poisson distribution with size n=10 (small sample)
set.seed(346)
s1 <- rpois(10, 7)
s2 <- rpois(10, 14)
s3 <- rpois(10, 21)
p1<- ggplot(data = data.frame(x= c(0,30)), aes(x))+
  # Prior Distribution
  stat_function(fun=dexp,
               args=list(1/beta),
```

```

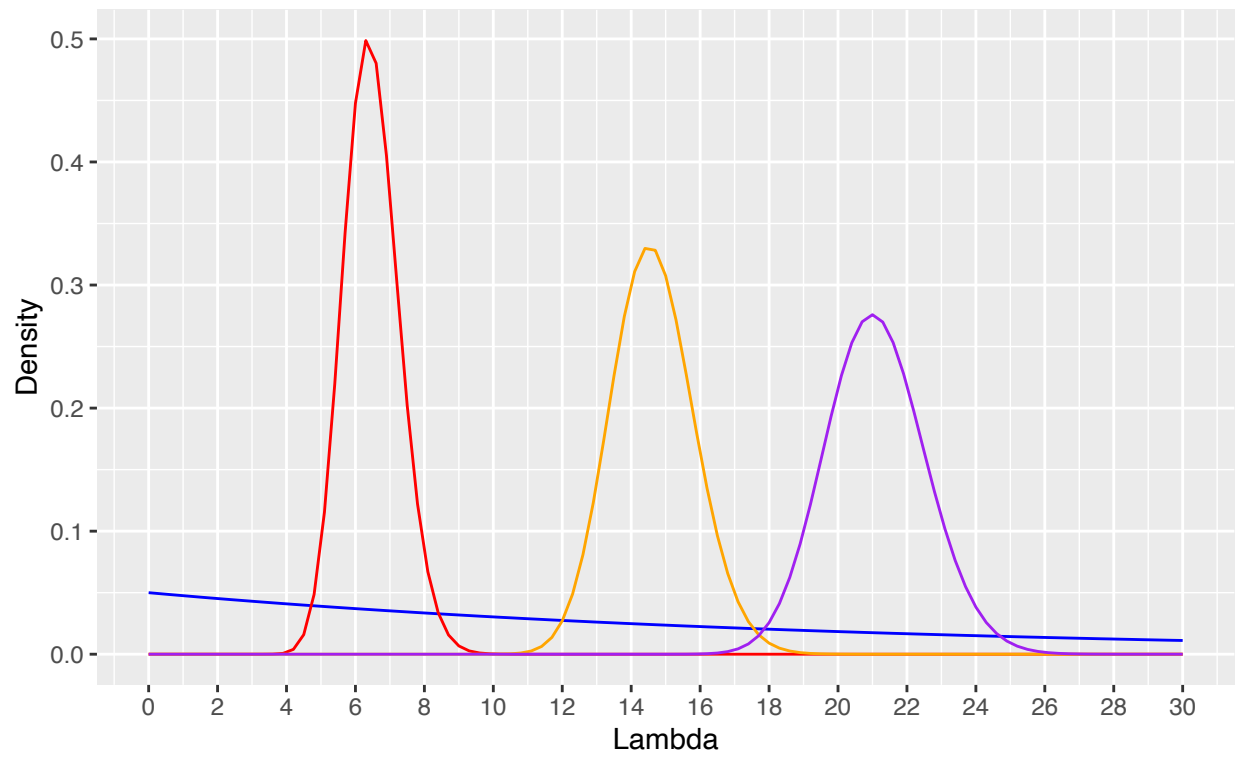
        color="blue")+
stat_function(fun=dgamma,
              args=list(sum(s1)+1, 10+1/beta),
              color="red")+
stat_function(fun=dgamma,
              args=list(sum(s2)+1, 10+1/beta),
              color="orange")+
stat_function(fun=dgamma,
              args=list(sum(s3)+1, 10+1/beta),
              color="purple")+
scale_x_continuous(breaks=seq(0,30, by=2))+
labs(title = 'Gamma Prior vs. Posterior for Lambda with sample size of 10',
      subtitle = "Blue: Prior. Red: 7 customers. Orange: 14 customers. Purple: 21 customers",
      x="Lambda", y="Density")

# Randomly sample from Poisson distribution with size n=100 (big sample)
s4 <- rpois(100, 7)
s5 <- rpois(100, 14)
s6 <- rpois(100, 21)
p2<- ggplot(data = data.frame(x= c(0,30)), aes(x))+
  stat_function(fun=dexp,
              args=list(1/beta),
              color="blue")+
  stat_function(fun=dgamma,
              args=list(sum(s4)+1, 100+1/beta),
              color="red")+
  stat_function(fun=dgamma,
              args=list(sum(s5)+1, 100+1/beta),
              color="orange")+
  stat_function(fun=dgamma,
              args=list(sum(s6)+1, 100+1/beta),
              color="purple")+
scale_x_continuous(breaks=seq(0,30, by=2))+
labs(title = 'Gamma Prior vs. Posterior for Lambda with Sample Size = 100',
      subtitle = "Blue: Prior. Red: 7 customers. Orange: 14 customers. Purple: 21 customers.",
      x="Lambda", y="Density")

```

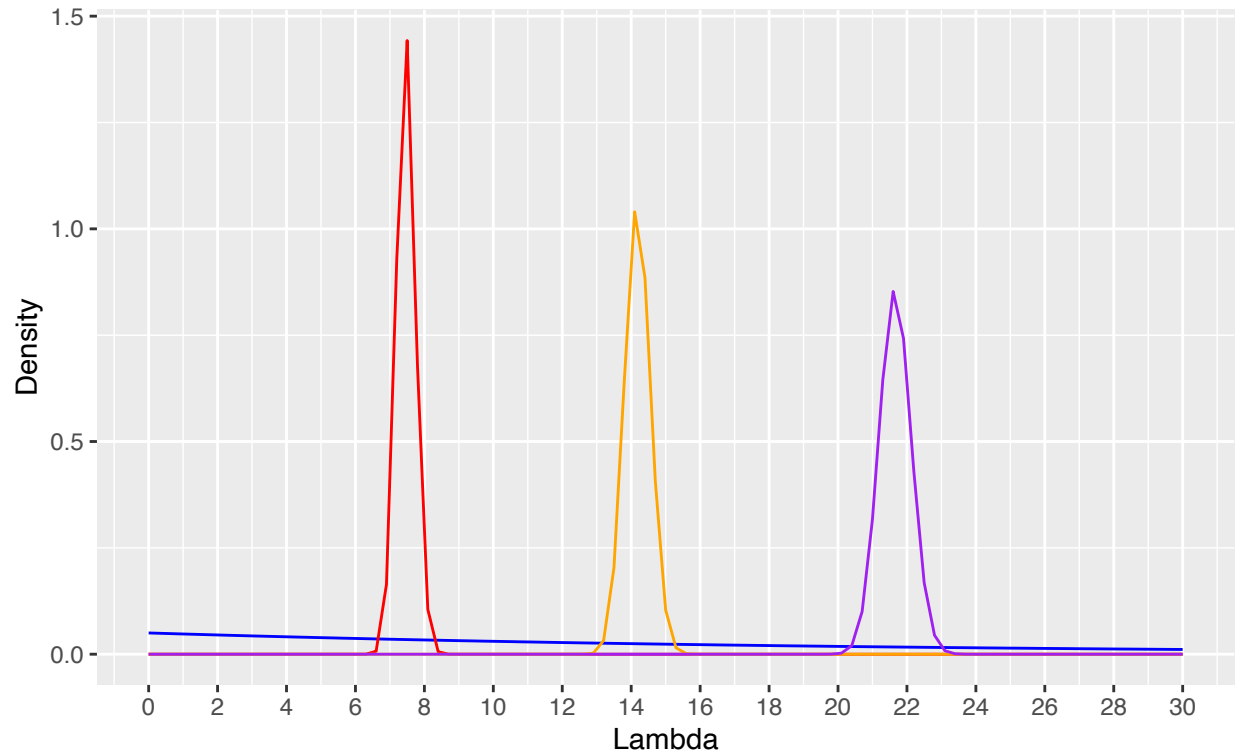
Gamma Prior vs. Posterior for Lambda with sample size of 10

Blue: Prior. Red: 7 customers. Orange: 14 customers. Purple: 21 customers



Gamma Prior vs. Posterior for Lambda with Sample Size = 100

Blue: Prior. Red: 7 customers. Orange: 14 customers. Purple: 21 customers.



We can clearly identify that the posterior represents a gamma distribution. We can confirm this because the conjugate prior for an exponential distribution is gamma distribution. With the prior distribution, we see that the website visits are decreasing. However with the graphs of the posterior functions, we see that the website visits vary depending on the sample mean. When the sample size, n , was 10, the posterior graphs peaked at 0.5 for 7 customers, approximately 0.33 for 14 customers and 0.27 for 21 customers. When the sample size is 100, we can see that the posterior graphs peaked at 1.5 for 7 customers, approximately 1.1 for 14 customers, and about 0.8 for 21 customers.

Part 2

Introduction

In modern days public transportation has grown to become a vital part of Torontonians' life. People commute to work, to school, and back home regularly using the ttc system. However, unforeseen events happen and there could be delays with bus, streetcar, or subway. In this simulation, we'd like to focus on the transportation of busses. We are interested in the average delayed minutes as well as the probability of having no delays. It's also important to use the most recent dataset to draw conclusions.

If we are getting conclusions of data from five years ago, changes may have been made and improvements may have been done. However, since last year the global pandemic struck every country hard. The transportation was used much less since February. Therefore, for this simulation, we will be simulating the data of January 2020.

We will first be cleaning the data and removing empty columns. Sometimes public data don't contain complete record and those empty inputs would affect the results of our simulation. Next, we will be conducting bootstraps simulations. Through bootstraps simulations, we can gather a better sense of the distribution of our data.

After bootstrapping, we will calculate the confidence interval of the two different bootstrap simulation topics. The confidence intervals (CIs) tell us the possible range around the estimate and know if the bootstrap sample is a normal distribution. A normal distribution would mean the peak of the graph will center on mean of the sample.

At the end of the simulation, we will summarise our simulation once more, except it is now after we've conducted simulations on our hypothesis. In addition, we will be further explaining the meanings of confidence intervals in the contexts of the data. To endm we will comment on any limitations of our simulations and future improvements.

Hypothesis: If a bus was delayed in Toronto, then its delay time would be less than 20 minutes; if it is not, then the proportion of no delay is less than 5% of January 2020.

Data

DATA INTRODUCTION:

The data used in this part of the assignment contains delays that occurred in TTC bus arrival times from January 2020 to August 2020. There are ten elements in this list. Their names and definitions are as follows:

Report Date: The date of the delay in the format YY/MM/DD. *Route*: The route number of the bus. *Time*: The time that the incident occurred in the format HH:MM:SS AM/PM. *Day*: The day of the week on which the incident occurred. *Location*: The general location of the incident. *Incident*: The cause of the delay. *Delay*: The time in minutes of the delay. *Gap*: The difference in time in minutes between the bus ahead and the delayed bus. *Direction*: The direction in which the bus was travelling. (B/W indicates both ways, NB indicates northbound, SB indicates southbound, EB indicates eastbound and WB indicates Westbound). *Vehicle*: The vehicle number.

DATA COLLECTION PROCESS:

Step 1 - Install and load the package: | The process starts off by installing the 'opendatatoronto' package using `install.packages()`. We load the package using the `library()` function, where the argument is the word 'delay' which shows all the packages in which TTC Delays occurred.

Step 2 - Search bus delays: | Once the 'opendatatoronto' package is loaded, we use `search_packages("Delay")` to search packages which brings all the packages which have delays in them. This returns TTC Bus delays,

TTC Subway delays and TTC Streetcar delays. Since we want bus delays, which is the third item, we use `'search_packages("Delay") [3,]'` to access the item on the third index, which is TTC Bus Delay Data.

Step 3 - Get the package ID: | Once we get TTC Bus Delay Data, we can run `list_package_resources()` to see what resources are available for this package. Since we want the most recent data which is of the year 2020, we use the `pull()` function with the index `[8, "id"]`, to get the eighth item which is TTC Bus Delay Data for 2020.

Step 4 - Get the data: | Once we have the id for the TTC Bus Delay for 2020, the final step in the data loading process is to actually acquire the dataset using the `get_resource()` function, where the argument is the id we found in the previous step.

The data is an excel sheet, and so we need to read it into RStudio, so we do that and then we are *only* looking at the delays that happened in the month of January and so we need to filter the data to only show observations recorded in the month of January.

Furthermore, in this dataset, there are some busses, which do not have their vehicle numbers recorded. So we remove all the busses in which have 'NA' recorded in the 'Vehicle' column, so we are left with all the busses that we know were tracked using their vehicle number and were truly delayed.

Below is the columns of the cleaned data:

Report Date: The date of the delay in the format YY/MM/DD.

Route: The route number of the bus.

Time: The time that the incident occurred in the format HH:MM:SS AM/PM.

Day: The day of the week on which the incident occurred.

Location: The general location of the incident.

Incident: The cause of the delay.

Delay: The time in minutes of the delay.

Gap: The difference in time in minutes between the bus ahead and the delayed bus.

Direction: The direction in which the bus was travelling. (B/W indicates both ways, NB indicates northbound, SB indicates southbound, EB indicates eastbound and WB indicates Westbound).

Vehicle: The vehicle number.

Average Delay Time	Average Gap Time	Standard Deviation of Delay Time	Proportion of No Delay
18.80641	29.41077	60.37528	3.83895%

The first numerical summary that we calculated was the mean of the Delay variable which told us the average delay time among all the incidents in January 2020. We used `'na.rm = TRUE'` to remove any 'NA' observations in case there were any. So we can see that the average delay time was 18.8 minutes.

The second numerical summary that we calculated was the mean of the Gap variable which told us the average difference between the bus ahead and the following bus and we removed any 'NA'. We can see that on average, the delayed bus was 29.4 minutes behind the bus ahead.

The third numerical summary that we calculated was the standard deviation of the 'Delay' variable which told us how spread out the graph is. Since the standard deviation is significantly higher than the mean, we know that the graph is spread out quite a bit.

Lastly, we calculated the proportion of busses that did not have any delays in January 2020. We filtered the data to only contain observations in which the `Jan.2020.Delay` variable was equal to zero. Then we divided the number of observations of that data by the number of observations of our original data to get approximately 0.0384 which translates to 3.8%.

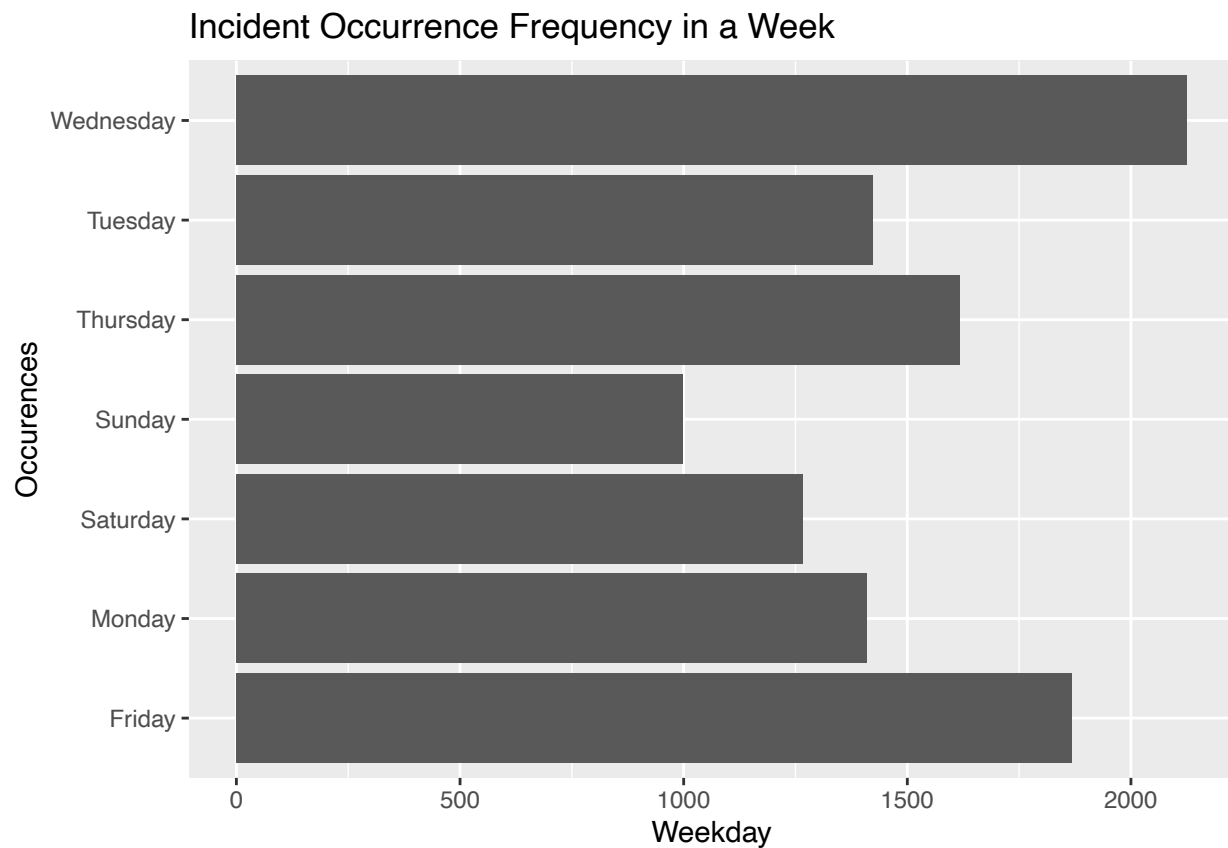
```
# Plot for which day had the most incidents
graph1 <- data_jan2020 %>%
  group_by(Jan.2020.Day) %>%
```

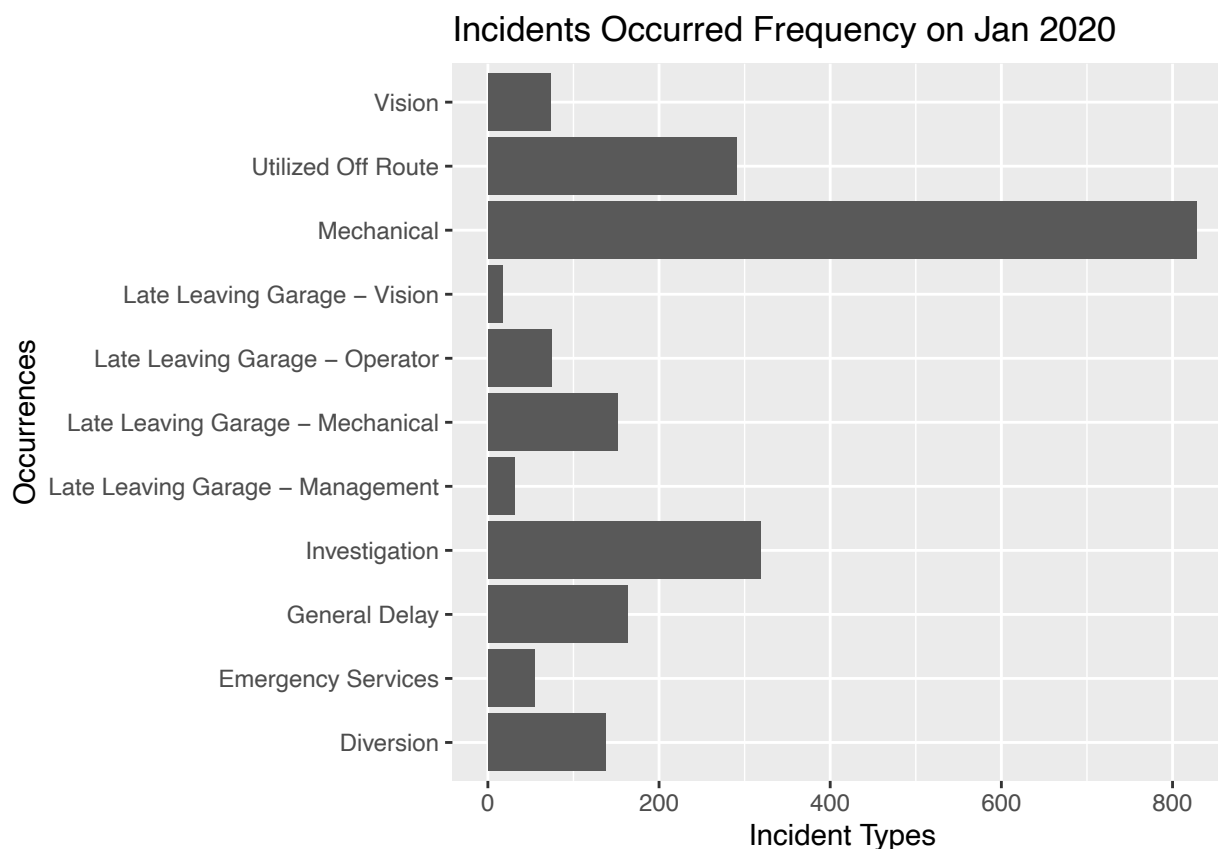
```

ggplot(aes(x = Jan.2020.Day, y = 2.5)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(title = 'Incident Occurrence Frequency in a Week',
        x = 'Occurrences',
        y = 'Weekday')

#Plot for which incident caused the most delays
graph2 <- data_jan2020 %>%
  group_by(Jan.2020.Incident) %>%
  ggplot(aes(x = Jan.2020.Incident, y = 0.5)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(title = 'Incidents Occurred Frequency on Jan 2020',
        x = 'Occurrences',
        y = 'Incident Types')

```





The first table is a barplot which shows on which day were delays the most common. The most delays happened right in the middle of the week on Wednesdays whereas the least delays happened on Sunday. This could be due to the fact that Sunday is a weekend and there might not be much traffic on Sundays. This was quite unsurprising as Wednesdays are right in the middle of the week and there could be more traffic on those days. To back this up we can see that the two days with the least delays were Saturday and Sunday which are weekends and there is not much traffic on the road.

The second plot is also a barplot showing the reason of the delays. We can see that the most common reason of delays were mechanical and the least common was lateness of leaving the bus depot due to vision. This was not predicted as one would think the most common reason would be general delay which might include traffic, however that was not the case.

SOFTWARE AND PROGRAM REFERENCES

1. RStudio Team (2020). RStudio: Integrated Development Environment for R. RStudio, PBC, Boston, MA URL <http://www.rstudio.com/>.
2. R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.

Methods

Bootstrap is a simulation method that allows statisticians to sample a data without the limitation of its size. Bootstrap sampling is the main method we used in analyzing, which is taking bootstrap sample in order to find out the variability of statistic. During a bootstrap simulation, we repeatedly pick random samples of same size from the population with replacement and obtain each simulation's numeric summaries. After we've conducted the number of bootstrapping we desire, we combine the data.

For our simulation, we will be simulating through empirical bootstrap sampling. The reason we chose to use this type of bootstrap on the bus delay of Toronto in January 2020 is we do not know the distribution of this dataset. In empirical bootstrap sampling, it doesn't require us knowing the distribution of our data. For the simulation, we will be doing 1000 iterations using the empirical bootstrap sampling. The sample size of each iteration we are doing is 4272, which is the number of rows in the dataset after cleaning. We are analyzing confidence intervals for the mean bus delay and proportion of no bus delay in Toronto which are our components of bootstrap. A confidence interval, as mentioned in the introduction, provides us with a plausible range of values that our population value lies in with certain percentage of confidence.

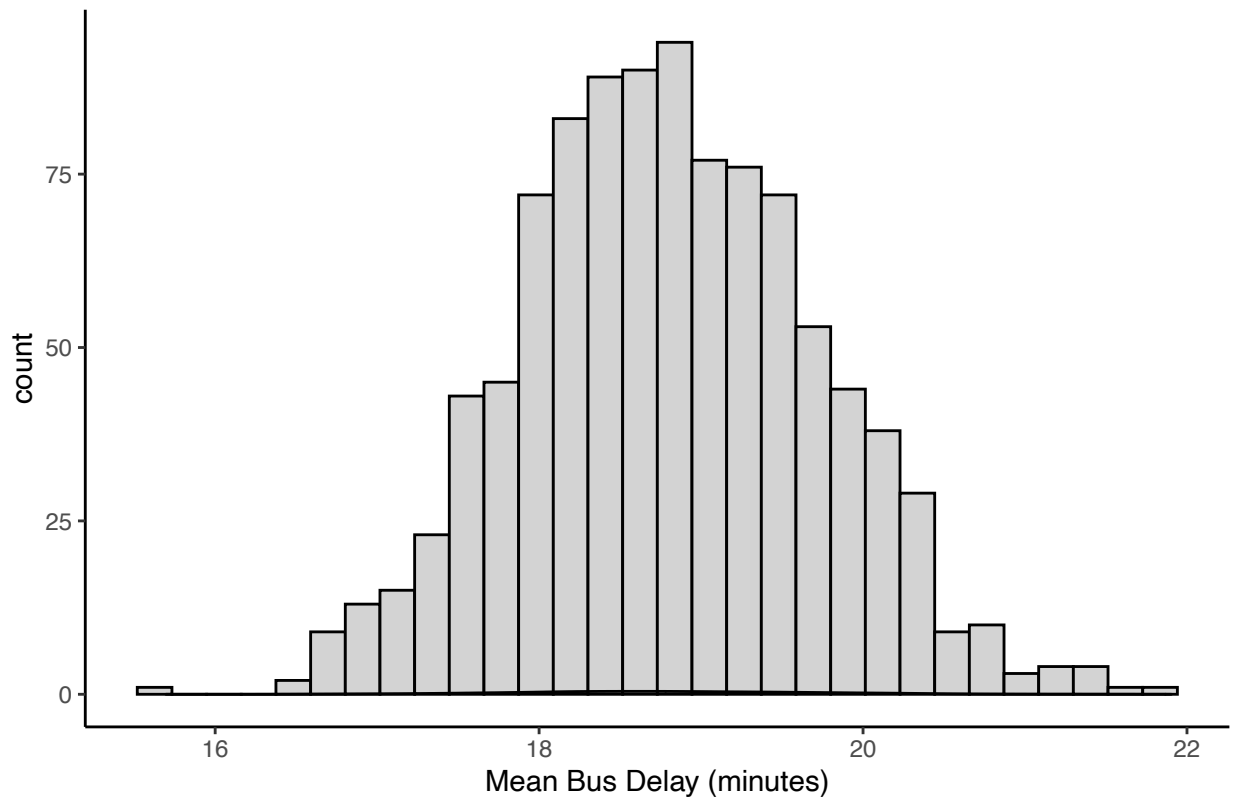
```
## [1] 4272    10
```

There are 4272 rows in the data that we've already cleaned and removed NA values. We cleaned NA values under *Jan.2020.Delay* column of dataset and assign the cleaned version again under the name of our original data.

```
# Bootstrap: sample with replacement
set.seed(587)
bt <- 1000 # Number of bootstrap resamples to take
n <- nrow(data_jan2020) # Sample size
btmeans <- as.numeric(bt)
for (i in 1:bt) {
  # Draw a bootstrap sample
  bootstrapsample <-
    sample(data_jan2020$Jan.2020.Delay,n,replace = TRUE)
  # Compute the bootstrap estimate
  btmeans[i] <- mean(bootstrapsample)
}
```

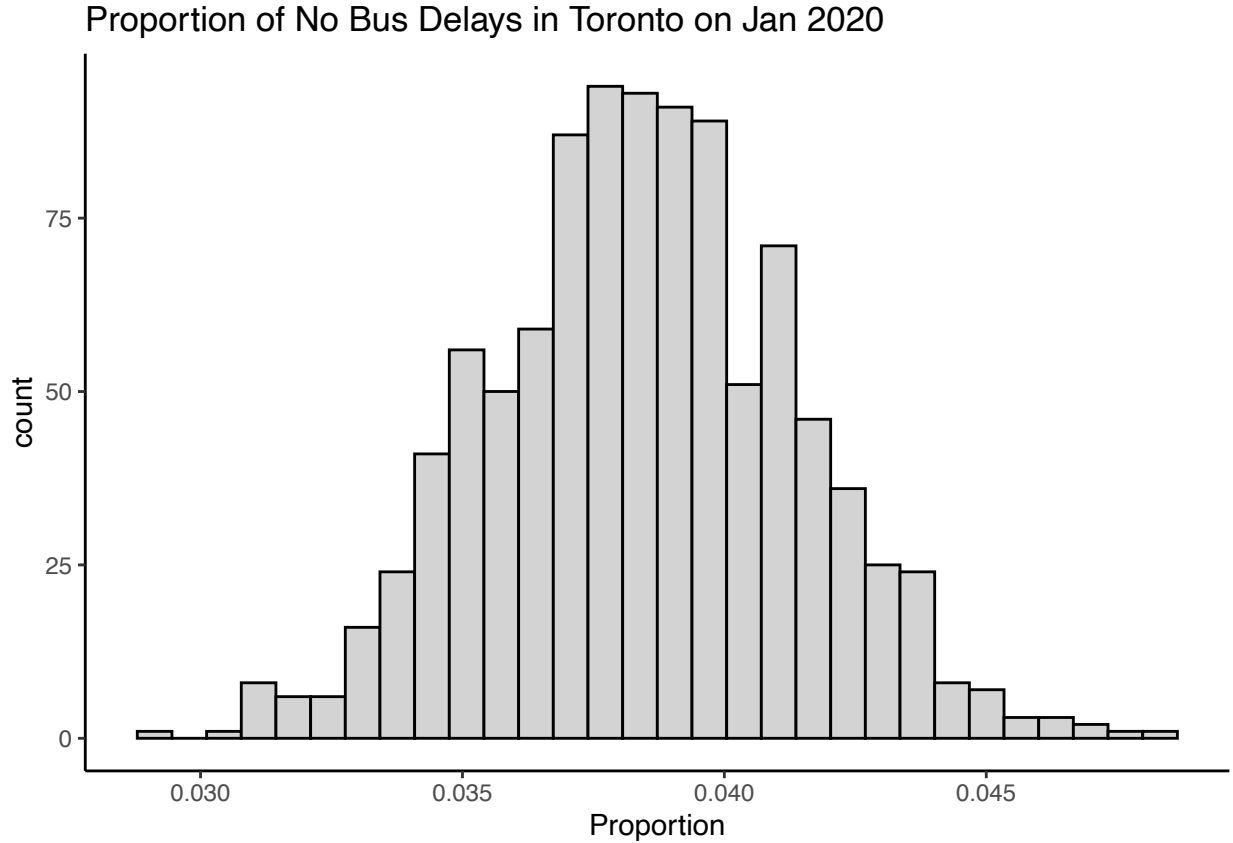
```
# Plot a hisrogram and density estimate, along with
# the normal approximation
tibble(x = btmeans) %>%
  ggplot(aes(x = x)) +
  theme_classic() +
  geom_histogram(bins = 30, colour = "black",fill = "light grey") +
  geom_density() +
  labs(title = 'Bootstrap Distribution for Mean Bus Delay in Toronto on Jan 2020',
        x = 'Mean Bus Delay (minutes)')
```

Bootstrap Distribution for Mean Bus Delay in Toronto on Jan 2020



We can see the histogram has a normal distribution and is unimodal. The peak of the histogram is centered at approximately 18.8 minutes of mean bus delay time in January 2020. The graph seems to be slightly skewed to the right with possible outliers on the left of the bell curve.

```
set.seed(346)
bt <- 1000 #number of resamples
n <- nrow(data_jan2020) #rows - sample size
prop <- as.numeric(bt)
nodelay <- as.numeric(data_jan2020$Jan.2020.Delay == 0)
for (i in 1:bt) {
  bootstrapsample <- sample(nodelay,n,replace = TRUE)
  # Compute the bootstrap estimate
  prop[i] <- mean(bootstrapsample)
}
tibble(x = prop) %>%
  ggplot(aes(x = x)) +
  theme_classic() +
  geom_histogram(bins = 30, colour = "black",fill = "light grey") +
  labs(title = 'Proportion of No Bus Delays in Toronto on Jan 2020',
        x = 'Proportion')
```



Since the graph is centered around 0.037 where most of the values lie, we can estimate the value of the proportion of no bus delays in January 2020 to be 0.037 which translates to 3.7%. As computed above, the actual percentage is 3.8%, so we can conclude that this is an accurate estimate.

PACKAGE REFERENCES

1. H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.
2. Thomas Lin Pedersen (2020). *patchwork: The Composer of Plots*. R package version 1.1.1. <https://CRAN.R-project.org/package=patchwork>
3. Mine Çetinkaya-Rundel, David Diez, Andrew Bray, Albert Kim, Ben Baumer, Chester Ismay and Christopher Barr (2020). *openintro: Data Sets and Supplemental Functions from ‘OpenIntro’ Textbooks and Labs*. R package version 2.0.0. <https://CRAN.R-project.org/package=openintro>

Results

Tables of Confidence Intervals

95-percent Confidence Interval of Mean

2.5%	97.5%
17.03927	20.61673

From this table we can say that we are 95% confident that the true mean bus delay time in January 2020 of our population will lie in the interval of 17.03927 to 20.61673 minutes.

99-percent Confidence Interval of Proportion

0.5%	99.5%
0.03113	0.04611

From this table we can conclude that we are 99% confident that the true proportion of no bus delayed in January 2020 of the population will be in the interval of 3.3113% and 4.611%.

Conclusions

From our first computation in *methods* that it is true that 95% of mean time of bus delays in January 2020 will be between 17.04 minutes and 20.62 minutes. We can conclude this confidently because when we computed the sample mean of the bus delay time, our value was approximately 18.806 minutes, which is in the confidence interval that we have calculated. It is also true that 99% of proportion of no bus delay in January 2020 is between 3.1% to 4.6%. We conclude this because the proportion of busses that were not delayed is 3.8% which lies between the confidence interval we calculated.

The length of our data is a month long which is a weakness of this simulation because the duration is too short to make any decisive decisions. It is not enough of an evidence to make any changes to the ttc system given the conclusion was drawn based on a month of delay data. In the future, it is important to increase the timeline of dataset to better the credibility of simulations since we can obtain a better picture of bus delays in Toronto over a longer period of time. There is a possibility that in January 2020 there was a mechanical or technological issue with the bus that causes longer average delay time. Therefore, we strongly recommend to increase the time length of data to make sure the result is more convincing to readers.

The next step would be to continue finding smaller confidence interval so we could be more accurate in our estimation. It would also be helpful if future reports the data was taken from not just one month but the whole year, and compare it with data from different years. This would help us to figure out common times of the year in which bus delays are longer or more frequent. We could even compare the data to different cities of a similar population and transit system, to see where we rank among the cities and to learn to implement different systems that other cities are using to decrease bus delays.

In the beginning, the first part of our hypothesis was: if a bus was delayed in Toronto, then its delay time would be less than 20 minutes. We have proven this hypothesis to be correct for our data selection by calculating the confidence interval of true value. The second part of our hypothesis was if a bus was not delayed in Toronto, then the proportion of no bus delay is less than 5% in January 2020. To prove this hypothesis, we went and calculated the probability of no bus delay in January 2020 and we received a value that was within the 99% confidence interval which means our confidence interval was correct.

In conclusion, we have successfully completed data cleaning process, produced various numeric and graphical summaries of the data, conducted a 1000-iteration bootstrap sampling, and finally calculated the two accurate confidence intervals of our hypothesis of the bus delay time in Toronto in January 2020.

All analysis for this report was programmed using **R version 4.0.2**.

Bibliography

1. Golemund, G. (2014, July 16) *Introduction to R Markdown*. RStudio. https://rmarkdown.rstudio.com/articles_intro.html. (Last Accessed: January 15, 2021)

2. Dekking, F. M., et al. (2005) *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer Science & Business Media.
3. Allaire, J.J., et. el. *References: Introduction to R Markdown*. RStudio. <https://rmarkdown.rstudio.com/docs/>. (Last Accessed: January 15, 2021)
4. “*Markdown Basics*.” R Markdown, http://rmarkdown.rstudio.com/authoring_basics.html. (Last accessed: March 3, 2021)
5. “*Align (Environment)*.” LaTeX Wiki, [http://latex.wikia.org/wiki/Align_\(environment\)](http://latex.wikia.org/wiki/Align_(environment)). (Last accessed: March 3, 2021)
6. Gibbs, Alison, and Alex Stringer. (2021, Jan 20) “*Chapter 11 Bayesian Inference: Estimation*.” awstringer1.Github.io <http://awstringer1.github.io/sta238-book/section-bayesian-inference-estimation.html#section-estimation-in-bayesian-inference-general-ideas>. (Last accessed: March 2, 2021)
7. “*LaTeX: Symbols*.” Art of Problem Solving. [<http://artofproblemsolving.com/wiki/index.php/LaTeX:Symbols#Arrows>] (<https://artofproblemsolving.com/wiki/index.php/LaTeX:Symbols#Arrows>). (Last accessed: March 3, 2021)
8. Gibbs, Alison, and Alex Stringer. (2021, Jan 20) “*Chapter 7 The Bootstrap*.” awstringer1.Github.io <https://awstringer1.github.io/sta238-book/section-the-bootstrap.html>. (Last accessed: March 5, 2021)
9. Thomas Lin Pedersen (2020). patchwork: The Composer of Plots. [<https://patchwork.data-imaginist.com>, <https://github.com/thomasp85/patchwork>.] (<https://patchwork.data-imaginist.com>, <https://github.com/thomasp85/patchwork>)
10. Kirill Müller and Hadley Wickham (2021). tibble: Simple Data Frames. <https://tibble.tidyverse.org/>, <https://github.com/tidyverse/tibble>.
11. Alex Douglas, Deon Roos. “An Introduction to R.” 28 Jan. 2021. Web. 05 Mar. 2021.