Assignment 4: Visualising Traffic Data

Submitted By: Kalyani Prashant Kawale | Student ID: 21237189

Data Setup

Loading the libraries and reading data in data-frame *TRAFFIC* from the long formatted csv file containing traffic data, recorded on 23 November 2016, for the N59 road at the junction outside the Data Science Building in the IDA Business park in Lower Dangan, Galway.

```
# Loading libraries
library(readr)
library(ggplot2)
library(dplyr)
library(ggforce)
library(colorspace)
library(viridis)
library(lubridate)
library(scales)
library(treemapify)
library(colorblindr)
library(plotly)

# Reading the Traffic data at Junction from csv file
TRAFFIC <- read_csv(file = 'Junction Turning Counts 2016 Outside DSI_LONG_FORMAT.csv')
```

Section 1:

Task:

To present a visualisation of the distributions of vehicles per 15 minute interval per vehicle type contributing to traffic at this junction. Vehicles types must be labelled with their full name (not abbreviations such as 'PCL').

Solution:

Data Setup:

- The visualization used for displaying the distribution of vehicles per vehicle type per time-stamp is a strip plot.
- For the given task a column with full names of given vehicles is added to the data-frame *TRAFFIC* using dplyr **mutate** function.
- Further, a data-frame *TRAFFIC_SEC1* has been created grouped by *TIME* and *vehicle_fullname* and the count is summarized into the sum of vehicle count per vehicle per time-stamp at each turn.

- An *Hour* column has been added to *TRAFFIC_SEC1* to be used in showing the hour and count of each point in plot in the interactive mode.

```
# Adding column vehicle_name for storing full names of vehicles
TRAFFIC <- TRAFFIC %>%  mutate(vehicle_fullname = case_when(
  vehicle == "PCL" ~ "Pedal Cycle",
  vehicle == "MCL"  ~ "Motorcycle",
  vehicle == "CAR" ~ "Cars",
  vehicle == "TAXI" ~ "Taxi Vehicles",
  vehicle == "LGV" ~ "Light Goods Vehicle",
  vehicle == "OGV1" ~ "Ordinary Goods Vehicle 1",
  vehicle == "OGV2" ~ "Ordinary Goods Vehicle 2",
  vehicle == "CDB" ~ "City Direct Bus",
  vehicle == "BEB" ~ "Bus Eireann Bus",
  vehicle == "OB" ~ "Other Bus"
))
# Calculating the vehicle count per 15 minute interval per vehicle type
TRAFFIC_SEC1 <- TRAFFIC %>% group_by(TIME, vehicle_fullname) %>%
  summarize(count=sum(count))
# Creating label for tooltip to be displayed in the interactive plot
TRAFFIC_SEC1 <- TRAFFIC_SEC1 %>%
  mutate(Hour = paste0(format.Date(TIME, "%H:%M"),"\nVehicle Type: ", vehicle_fullname,
                       "\nVolume of Traffic: ", count))
```

Visualization:

- The data consists of multiple points with very small values, mostly zeros, thus their quantile values are cluttered in a very small region, hence a simple strip plot has been used to display the distribution where each point in the plot represents the count of vehicle for a vehicle type at any 15 minute interval.
- The **geom_point** geometric along with **position_jitter** has been used to spread the points over a small area to provide better discernibility.

```
# Note: Code referred from Unit 5 - Week 11 Worksheet -
# Part 2: Visualising Moycullen 2019 Temperatures [1]

# Plotting the distribution strip plot
distribution_plot <- ggplot(TRAFFIC_SEC1, aes(y=vehicle_fullname, x=count, label=Hour)) +
  # adding point for each vehicle type per 15 minute interval with small jitter
  geom_point(size = 1.25, alpha = 0.2,
             position = position_jitter(width = 1.7)) +
  # setting the scale for x-axis
  scale_x_continuous(expand=c(0,0),
                     breaks = seq(0,450,by=50),
                     limits= c(0,450),
                     name = "Count of Vehicles at Junction per 15 Minutes") +
  ggtitle("Distribution of Traffic at Junction on November 23, 2016") +
  # setting plot theme
  theme_minimal() +
  theme (
    panel.grid.major.y =element_line(colour = "gray95", size=0.25),
    panel.grid.minor.x =element_line(colour = "gray95", size=0.15),
    panel.grid.major.x =element_line(colour = "gray95", size=0.25),
```

```
    axis.title.y = element_blank(),
    axis.text.y = element_text(size=10, face="bold"),
    axis.text.x = element_text(size=8),
    axis.title.x = element_text(size=12, vjust=-0.3),
    legend.title = element_blank(),
    plot.margin = margin(t = 4, r = 10, b = 4, l = 4, "pt"),
    plot.title = element_text(size=16, hjust=0.45, face="bold"),
    legend.position = "none")

distribution_plot
```

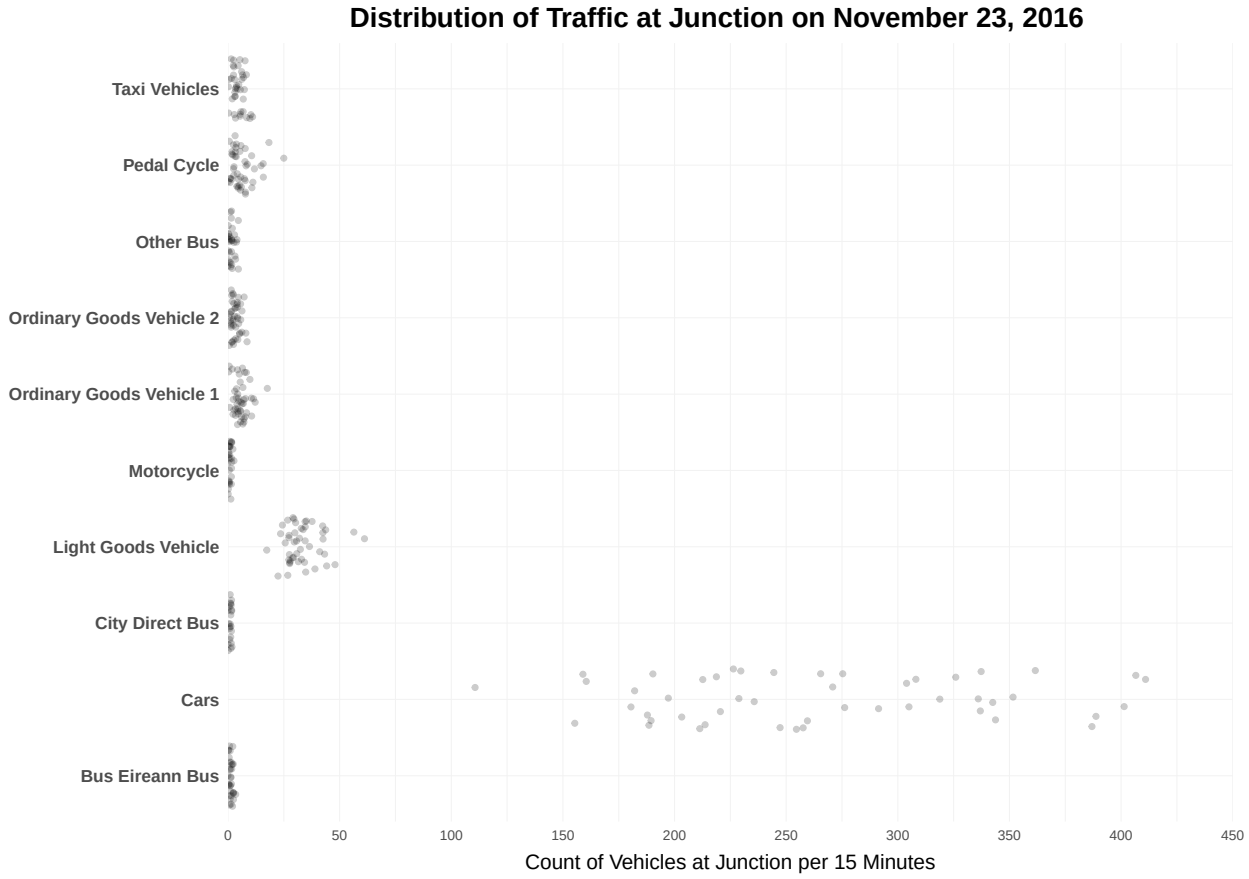

Figure 1: Strip Plot Showing the Distribution of Vehicles Per 15 Minute Interval Per Vehicle Type Contributing to Traffic at the Junction on November 23, 2016

- The plotly function **ggplotly** has been used to generate an interactive version of the strip plot to provide a tooltip for every point in the plot, with the aim of providing more detailed information.
- The plot in interactive mode displays information for each geom_point as follows,

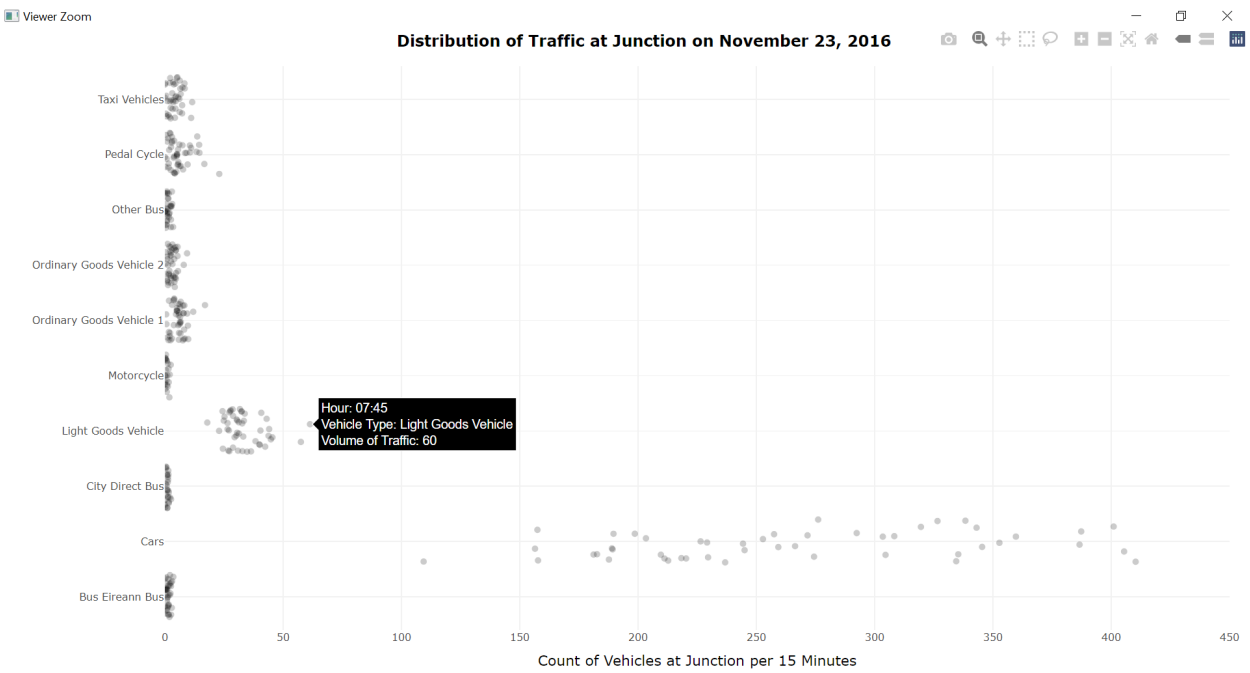

Figure 2: Tooltip Showing the Hour and Volume of Traffic for a Vehicle Type Light Goods Vehicle at the 15 Minute Time-Stamp of 7:45 in the Morning.

Section 2:

Task:

To present a visualisation that shows how the proportions of the traffic coming from D divides into the roads indicated by A, B and C at different times of the day. The times of the day are early morning (7 to 9.30 am), late morning (9.30 to 12 noon), afternoon (12 to 14.30), late afternoon (14.30 to 17.00) and evening (17.00 to 19.00).

Data Setup:

- A parallel set visualisation has been selected to display the proportions of the flow of traffic coming from road D and going to roads A, B and C at different times of the day.
- As the *TIME* variable has been divided into five categories representing the times of the day, which is an ordered categorical variable, rather than a time-series, the parallel set provides a good intuition of the flow and proportions of traffic for turns DA, DB and DC for interpretation.
- A data-frame *TRAFFIC_SEC2* has been generated to store the traffic data for the turns DA, DB and DC, since the proportions of traffic coming from D and going towards A, B and C needs to be compared.
- Further, the *TIME* date-time variable has been mutated and ordered into a categorical variable representing the times of the day.
- The dplyr **summarize** function has been used to sum the vehicle count for each turn and each time category, in order to get the proportions of the traffic for the said turn and time category.

```r
# Getting subset of TRAFFIC for turn DA, DB and DC
TRAFFIC_SEC2 <- filter(TRAFFIC, turn %in% c("DA", "DB", "DC"))

# Note: Following code has been referred from,
# Unit 5 - Week 9 Worksheet - Parallel Sets [2] and reference [3].

# Data Pre-Processing
TRAFFIC_SEC2 <- TRAFFIC_SEC2 %>%
  # setting the times of day
  mutate(TIME = case_when(
    (TIME >= "2016-11-23 07:00:00 UTC" & TIME < "2016-11-23 09:30:00 UTC")  ~
      "Early Morning",
    (TIME >= "2016-11-23 09:30:00 UTC" & TIME < "2016-11-23 12:00:00 UTC")  ~
      "Late Morning",
    (TIME >= "2016-11-23 12:00:00 UTC" & TIME < "2016-11-23 14:30:00 UTC")  ~
      "Afternoon",
    (TIME >= "2016-11-23 14:30:00 UTC" & TIME < "2016-11-23 17:00:00 UTC")  ~
      "Late Afternoon",
    (TIME >= "2016-11-23 17:00:00 UTC" & TIME < "2016-11-23 19:00:00 UTC")  ~
      "Evening",
    )) %>%
  # grouping the data-frame by turn and time of the day
  group_by(turn, TIME) %>%
  # adding the counts of vehicles for each turn at each time using summarize
  summarize(count=sum(count)) %>%
  # setting the custom name and order of levels for TIME column
  mutate(TIME = factor(TIME, levels =
                         c("Early Morning",
                           "Late Morning",
                           "Afternoon",
                           "Late Afternoon",
                           "Evening"))) %>%
  # renaming turn column
  mutate(turn = case_when(
  turn == "DA" ~ "Road D to Road A",
  turn == "DB" ~ "Road D to Road B",
  turn == "DC" ~ "Road D to Road C")) %>%
  # setting name and order of levels for turn column
  mutate(turn = factor(turn, levels = c(
  "Road D to Road A",
  "Road D to Road B",
  "Road D to Road C"
)))

# Using gather_set_data of ggforce library to transform TRAFFIC_SEC2 data frame
# into parallel set compatible format
TRAFFIC_SEC2_ps <- gather_set_data(TRAFFIC_SEC2, c(1,2))
# Setting level order for displaying turn axis before TIME axis
TRAFFIC_SEC2_ps$x <- factor(TRAFFIC_SEC2_ps$x, levels = c("turn",  "TIME"))
```

Visualization:

- The parallel set plotted below presents a comparison of proportion of all vehicles or the traffic for turns DA, DB and DC for different times of the day, where the y-axis represents proportion of traffic and the length of bars created using **geom_parallel_sets** is the aesthetic used to display these proportions for each turn and time category.
- The aesthetic representing the value of each turn is color, where the **orange color represents turn DA**, **green represents DB** and **blue represents DC**. The colors have been taken from the color blindness friendly palette created by Okabe and Ito.
- The time categories have been ordered to be displayed in their natural order of occurrence, which provides the insight that traffic for the concerned turns was minimum during early morning, while maximum traffic flowed during evening on November 23, 2016 (Wednesday), which could have been due to the fact that November 24 was Thanksgiving day and November 25 was Black Friday, hence more people might be leaving Galway city to go home for the long weekend, this can also be seen from the maximum proportion of traffic for turn DB displayed by green color, as the road B leads to the north western towns and townlands.

```
# Note: Following code has been referred from Unit 5 - Week 9 Worksheet - Parallel Sets[2]

# Plotting parallel-set
ggplot(TRAFFIC_SEC2_ps, aes(x, id=id, split=y, value=count)) +
  # creating the parallel sets
  geom_parallel_sets(aes(fill=turn), alpha=0.4, axis.width=0.11) +
  geom_parallel_sets_axes(axis.width=0.15, fill="grey30", color="grey80") +
  # setting the axes labels horizontally
  geom_parallel_sets_labels(color="grey98", size=3.5, angle=0) +
  scale_x_discrete(name=NULL, expand=c(0, 0), labels=c("TURN", "TIME")) +
  # scaling the y-axis based on sum of proportions for all turns
  scale_y_continuous(
    expand = c(0, 0),
    limits = c(0, 3800),
    breaks = seq(0, 3800, by=600),
    labels = seq(0, 3800, by=600),
    name = "Count of All Vehicles Contributing to Traffic"
  ) +
  # setting the colors for each turn
  scale_colour_OkabeIto(use_black=F, order = c(5, 1, 3), darken=0.05) +
  ggtitle("Proportions of Traffic Coming from Road D Turning into Roads A, B and C") +
  # setting the theme of plot
  theme(
    axis.line = element_blank(),
    plot.margin = margin(14, 2, 2, 2),
    axis.ticks.x = element_blank(),
    axis.ticks.y = element_line(color="grey30", size=0.5),
    axis.text.x = element_text(color = "grey25", size=12, vjust=-0.2, face="bold"),
    axis.text.y = element_text(color = "grey25", size=8, face="bold"),
    legend.background = element_blank(),
    legend.key        = element_blank(),
    panel.background  = element_blank(),
    panel.border      = element_blank(),
    strip.background  = element_blank(),
    plot.background   = element_blank(),
```

```
    panel.grid = element_blank(),
    plot.title = element_text(size=16, hjust=0.45, face="bold"),
    legend.position = "none"
)
```

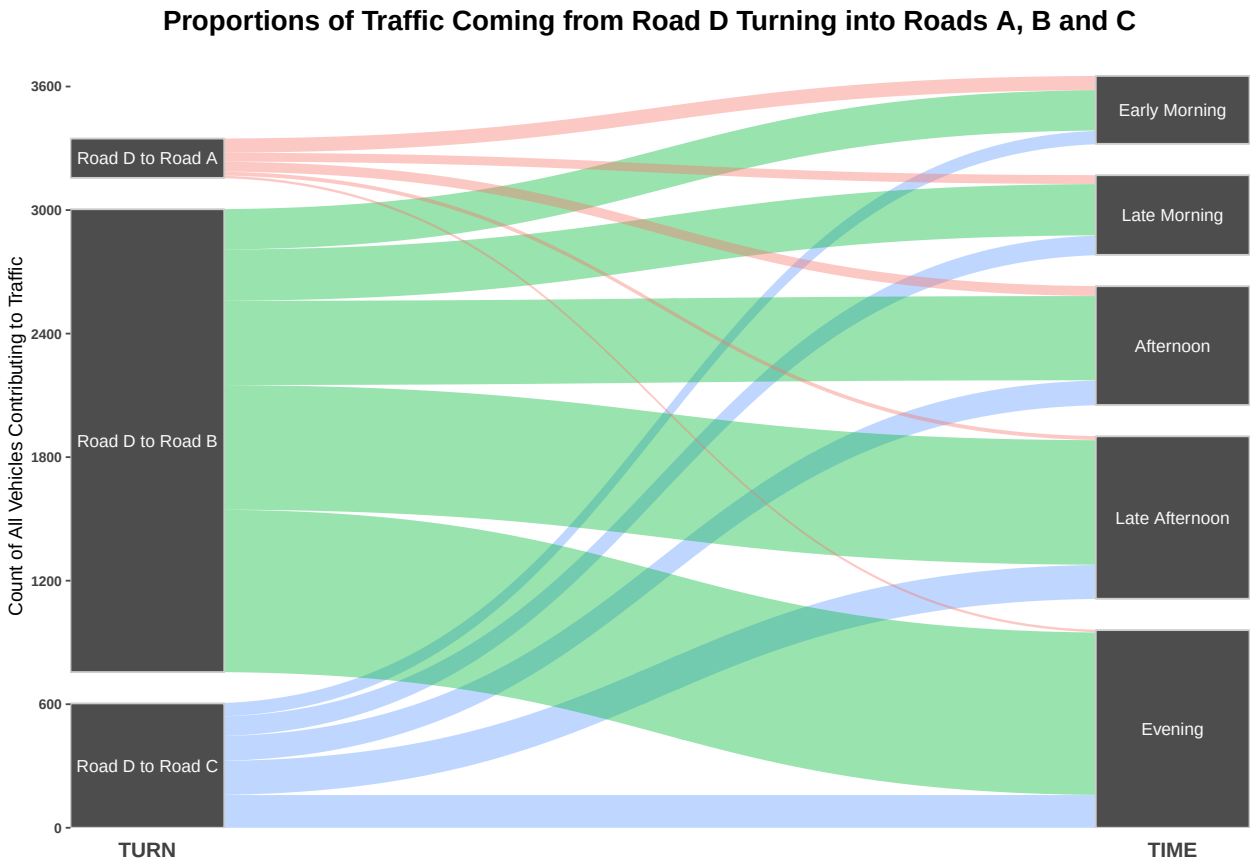

Figure 3: Parallel Set displaying the Flow and Proportions of Traffic coming from Road D, dividing into Roads A, B and C during Early Morning, Late Morning, Afternoon, Late Afternoon and Evening on November 23, 2016.

Section 3:

Task:

To present a visualisation of the volume of vehicles at this junction per vehicle type at each timestamp in the data set. This should be a single plot. A line graph, area or bar plots are not acceptable solutions. The reader should be able to perceive any patterns in the data.

Data Setup:

- To display the volume of vehicles at the junction per vehicle type at each time-stamp, a heat-map has been used.

- In order to get the volumes, the data in *TRAFFIC* data-frame has been grouped according to *vehicle_fullname* (vehicle type) and *TIME* variables and the count of vehicles is summed to get the volume using dplyr **group_by** and **summarize** functions and saved into column named *all_count*.

```
# Summarizing the count of vehicles according to vehicle type and time-stamp,
# and storing into TRAFFIC_SEC3 data-frame
TRAFFIC_SEC3 <- TRAFFIC %>% group_by(vehicle_fullname, TIME) %>%
  summarize(all_count=sum(count))
```

Visualization:

- The cells of heat-map have been created using the geometric **geom_tile**, which uses the aesthetic of color gradients to represent the volume of vehicles at the junction for each vehicle represented on the y-axis and each time-stamp represented on the x-axis.
- The color scale used has been taken from the viridis package, and the colors are ordered to represent smaller values with lighter colors and vice-versa.
- Darker the color more the volume of vehicles of a certain type at a given time-stamp and lighter the color lesser the volume.
- The heat-map is able to show the dominance of cars over any other type of vehicle over the entire time-frame and the volume for any vehicle type at any given time-stamp is also visible.
- The **geom_text** geometric has been used to double-encode the volume (given by *all_count* column).

```
# Note: Following code referred from Unit 5 - Week 9 Worksheet - Time Series Heat Maps.[4]

# Plotting the heat-map
ggplot(TRAFFIC_SEC3, aes(x=TIME, y=vehicle_fullname, fill=all_count)) +
  # setting the heat-map cells to be filled by scale_fill_viridis_C
  geom_tile(colour="white", size=1) +
  geom_text(aes(label=all_count),
            colour = ifelse(TRAFFIC_SEC3$all_count>100,"white", "black"),
            size=1.95) +
  # removing the spaces between plot and y-axis
  scale_y_discrete(name=NULL, expand=c(0,0)) +
  # scaling the TIME variable
  scale_x_datetime(date_breaks="15 min", date_labels="%H:%M", expand=c(0,0)) +
  # setting the gradients in reverse order
  scale_fill_viridis_c(option="B", direction=-1,
                       guide=guide_colourbar(direction="horizontal",
                                             barwidth=14,
                                             barheight=1.5)) +
  ggtitle("Volume of Vehicles at Junction on November 23, 2016") +
  # setting the theme of plot
  theme(axis.text.y = element_text(size=10, face="bold", vjust=-0.2),
        axis.text.x = element_text(size=8, angle=45, vjust=0.55),
        axis.ticks.x = element_line(size=0.5, colour="grey30"),
        axis.line.x = element_line(size=0.5, colour="grey30"),
        axis.ticks.y = element_blank(),
        axis.line.y = element_blank(),
        legend.text = element_text(size=9),
        legend.position = c(0.5, -0.175),
```

```
        legend.title = element_blank(),
        legend.margin = margin(r=5, l=5, t=2, b=2.5),
        panel.background = element_blank(),
        panel.grid = element_blank(),
        plot.margin = unit(c(1, 0.25, 2.5, 0), "cm"),
        plot.title = element_text(size=16, hjust=0.45, face="bold")
        )
```

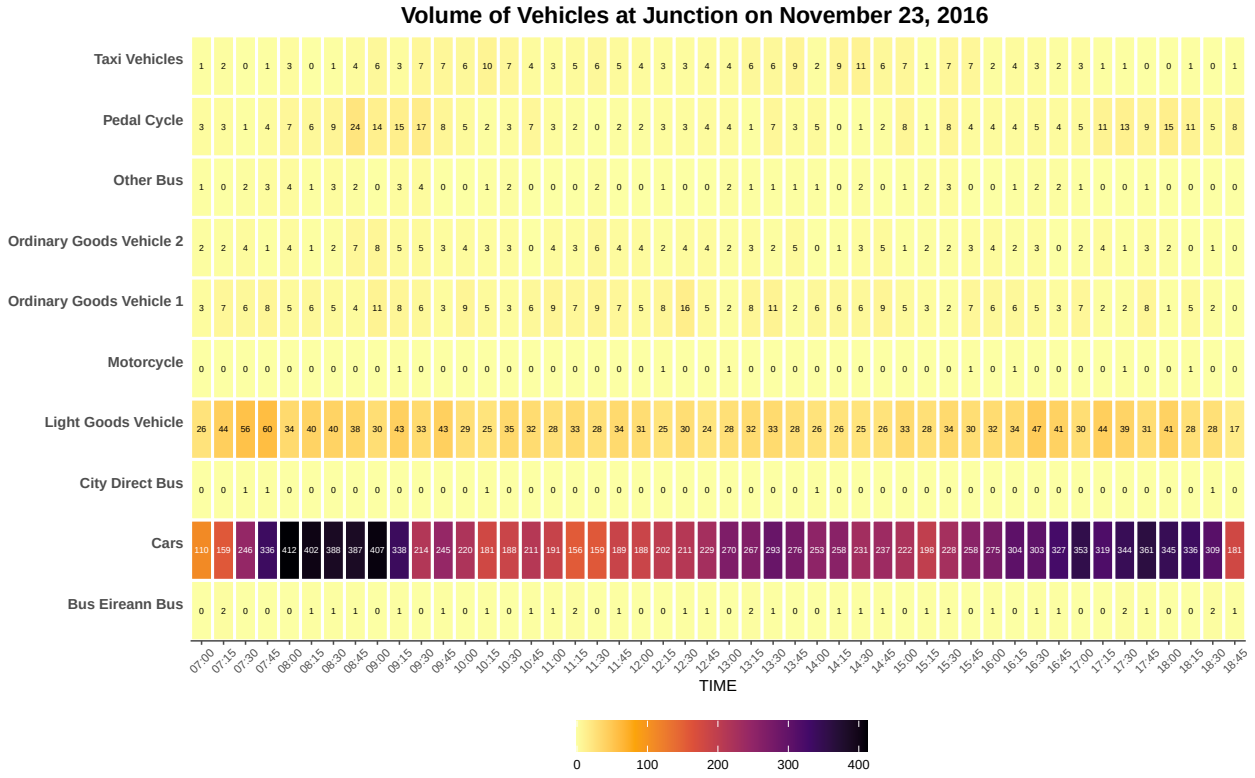

Figure 4: Volumes of Vehicle for Each Type of Vehicle at Each Time-Stamp Between 7 am to 7 pm on November 23, 2016.

Section 4:

Task:

To present a visualisation that shows the proportion of categories of vehicles and their subcategories at this junction over the full 12 hour period, given the following categories,

Two-Wheel Vehicles: PCL, MCL
Cars : Car, Taxi
Goods Vehicles: LGV, OGV1, OGV2
Buses and Public Transport: CDB, BEB, OB

Data Setup:

- Since the requirement of the task includes dividing the ten vehicle types into categories and displaying the proportions of the said categories along with the sub-categories, the visualisation chosen for the task is a Tree Map.
- Tree Maps are useful to show hierarchical proportions, and categorizing the vehicle types creates such a hierarchy, further the proportions are to be presented over the full 12 hour period and not individual points of time, thus a tree map can certainly provide the gist of the number of vehicles per category and sub-category.
- The data-frame *TRAFFIC_SEC4* has been created to organize the data for the tree map, in multiple steps as follows,
  - Firstly, a *vehicle_category* column has been added which contains the category of each vehicle type
  - The data-frame is then grouped according to the vehicle category and type and the count of each vehicle type is summed using the dplyr *summarize* function.
  - Since the volume of each vehicle type is distributed very unevenly, the data has been normalised using min-max normalization to bring the data in the range of 0 to 1, and a small amount (0.00001) is added to each value to ensure that extremely small values don't get converted to zero. These values are multiplied by a common factor of 2200, to ensure that all categories get displayed in the tree map.
  - Lastly, the colors for each category and gradients of the colors for sub-categories are calculated using **gradient_n_pal** method of scales package and stored in column named colour.

```
# Adding column for categories of vehicles
TRAFFIC_SEC4 <- TRAFFIC %>% mutate(vehicle_category = case_when(
  vehicle == "PCL" ~ "Two-Wheel Vehicles",
  vehicle == "MCL"  ~ "Two-Wheel Vehicles",
  vehicle == "CAR" ~ "Cars",
  vehicle == "TAXI" ~ "Cars",
  vehicle == "LGV" ~ "Goods Vehicles",
  vehicle == "OGV1" ~ "Goods Vehicles",
  vehicle == "OGV2" ~ "Goods Vehicles",
  vehicle == "CDB" ~ "Buses & Public Transport",
  vehicle == "BEB" ~ "Buses & Public Transport",
  vehicle == "OB" ~ "Buses & Public Transport"
))
# Saving the category column as a factor
TRAFFIC_SEC4$vehicle_category <- factor(TRAFFIC_SEC4$vehicle_category,
                                       levels = c("Two-Wheel Vehicles",
                                                  "Cars",
                                                  "Goods Vehicles",
                                                  "Buses & Public Transport"))
# Summarizing the vehicle count according to category and sub category
TRAFFIC_SEC4 <- TRAFFIC_SEC4 %>% group_by(vehicle_category, vehicle_fullname) %>%
  summarise(vehicle_count=sum(count))
# Normalizing data to ensure display of each category
TRAFFIC_SEC4$vehicle_count_norm <- as.numeric((TRAFFIC_SEC4$vehicle_count -
                                  min(TRAFFIC_SEC4$vehicle_count)) /
  (max(TRAFFIC_SEC4$vehicle_count) - min(TRAFFIC_SEC4$vehicle_count))) +
  (0.00001 * 2200)

# Assigning numeric value to each category and storing in column named index
```

```
TRAFFIC_SEC4$index <- as.numeric(factor(TRAFFIC_SEC4$vehicle_category))
# Getting the number of categories
n <- length(unique(TRAFFIC_SEC4$vehicle_category))
# Creating colors for each category and gradients of the color for
# each sub-category using the index value
TRAFFIC_SEC4 <- TRAFFIC_SEC4 %>% group_by(index) %>%
  mutate(max_count = max(vehicle_count), colour = gradient_n_pal(
    # settings to get colors based on hue, chroma and lightness depending on category
      sequential_hcl(6, h = 360 * index[1]/n, c = c(40, 10), l = c(20, 90), power = 0.6)
  )(1-(vehicle_count/max_count)))
```

Visualization:

- The aesthetic of area is used to display the count of vehicles per category and sub-category type.
- The geometric **geom_treemap** is used to create the tree map, while the labels of sub-categories of the vehicles and the count of the sub-categories of vehicles are displayed using two **geom_treemap_text**.
- **geom_treemap_subgroup_text** displays the categories of vehicles.

```
# Setting values for blocks with lighter color to set label color accordingly
small_values <- c(188, 7, 5, 136)
# Note: Following code referred from Unit 5 - Week 9 Worksheet -
# Hierarchical Proportions.[5]

# Plotting the Tree-Map
ggplot(data=TRAFFIC_SEC4, aes(area=vehicle_count_norm,
                              fill=colour,
                              subgroup=vehicle_category)) +
  geom_treemap(colour="grey100", size=0.5*.pt, alpha = NA) +
  # setting the labels for vehicle type
  geom_treemap_text(aes(label=vehicle_fullname),
                    colour=ifelse(TRAFFIC_SEC4$vehicle_count %in% small_values,
                                  "grey30","grey97"),
                    alpha=0.7,
                    size=24,
                    place="center",
                    fontface="bold") +
  # setting the labels for count of the vehicle type at the junction,
  # the smaller values are labelled with dark color and larger values
  # are colored with light color to make labels readable against
  # the light or dark backgrounds
  geom_treemap_text(aes(label = format(vehicle_count, nsmall=0, big.mark=",",trim=TRUE)),
                    color=ifelse(TRAFFIC_SEC4$vehicle_count %in% small_values,
                                 "grey30", "grey97"),
                    alpha=0.7,
                    size=14.5,
                    min.size=0,
                    place = "topleft",
                    padding.x=grid::unit(1, "mm"),
                    padding.y=grid::unit(1.5, "mm")) +
  # setting the borders between tree map categories and sub-catgories
  geom_treemap_subgroup_border(colour="grey99", size=2.5) +
```

```
  # setting the category labels
  geom_treemap_subgroup_text(grow=FALSE,
                             colour="grey96",
                             size=52,
                             place="bottomleft",
                             padding.y=grid::unit(1.75, "mm"),
                             fontface = "bold",
                             alpha = 0.4) +
  # setting to ensure the scaling applied does not change
  scale_fill_identity()
```

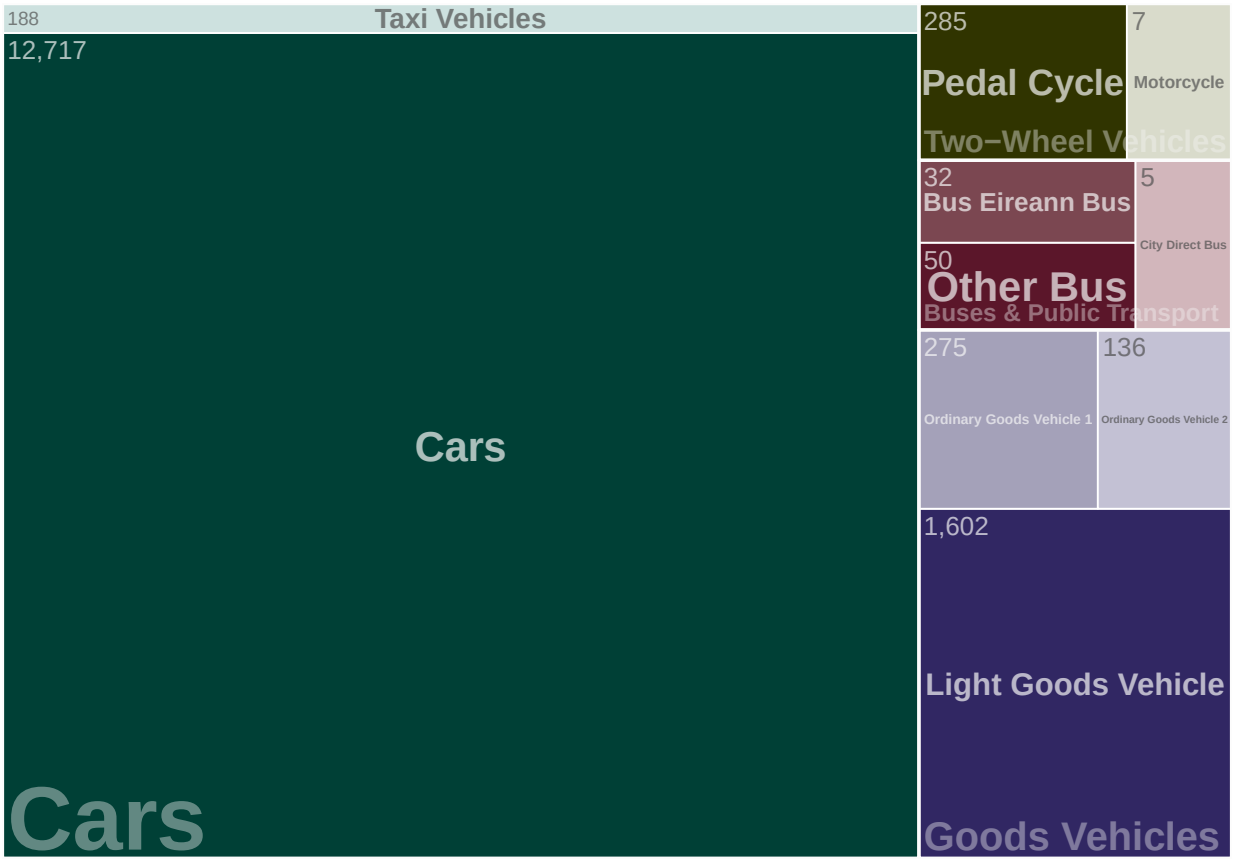

Figure 5: Tree Map Showing Proportions of Categories and Sub-Categories of Vehicles at the Junction Over 12-Full Hours Starting from 7 am to 7 pm.

Acknowledgements:

Following resources were referred to perform the above tasks,

[1] Dr. Conor Hayes. (2022). CT5100: Week 11 Worksheet - Visualising Moycullen 2019 Temperatures.
[2] Dr. Conor Hayes. (2022). CT5100: Week 9 Worksheet - Parallel Sets.
[3] Custom sorting (non-alphabetical). Available at: https://stackoverflow.com/questions/23995285/custom-sorting-non-alphabetical [Accessed on:22/03/2022]
[4] Dr. Conor Hayes. (2022). CT5100: Week 9 Worksheet - Time Series Heat Maps.

[5] Dr. Conor Hayes. (2022). CT5100: Week 9 Worksheet - Hierarchical Proportions.
[6] Dr. Conor Hayes. (2022). CT5100: Unit 2 - Section 2 Worksheet - CVD simulation using the colorblindr package.
[7] Dr. Conor Hayes. (2022). CT5100: Unit 2 - Section 1 Worksheet - Colours.