Traffic accident Density in New Zealand

By Phua, Jemin, Jack, Chathu and Ziling

Original Ideas

- Covid19 dataset combined with boxing events
- Relationship between profitability of games and their genre's and other tags
- Weather correlated with hospital data

Chosen Idea

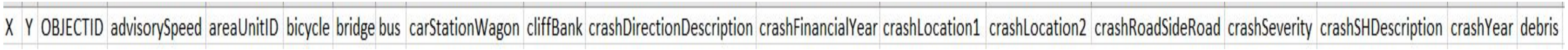
- Traffic Dataset combined with crash dataset from stats nz

Ethics

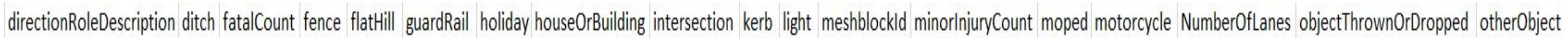
- Privacy concerns
- Inherited bias from dataset

The Data

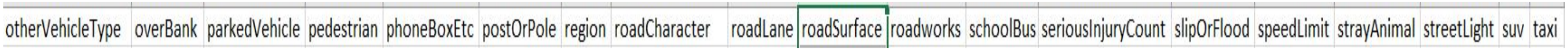
CAS CSV



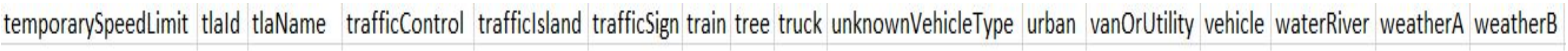
X	Y	OBJECTID	advisorySpeed	areaUnitID	bicycle	bridge	bus	carStationWagon	cliffBank	crashDirectionDescription	crashFinancialYear	crashLocation1	crashLocation2	crashRoadSideRoad	crashSeverity	crashSHDescription	crashYear	debris



directionRoleDescription	ditch	fatalCount	fence	flatHill	guardRail	holiday	houseOrBuilding	intersection	kerb	light	meshblockId	minorInjuryCount	moped	motorcycle	NumberOfLanes	objectThrownOrDropped	otherObject

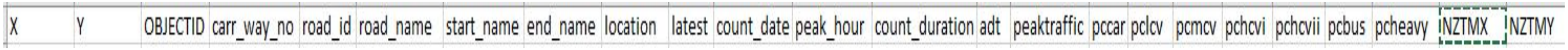


otherVehicleType	overBank	parkedVehicle	pedestrian	phoneBoxEtc	postOrPole	region	roadCharacter	roadLane	roadSurface	roadworks	schoolBus	seriousInjuryCount	slipOrFlood	speedLimit	strayAnimal	streetLight	suv	taxi

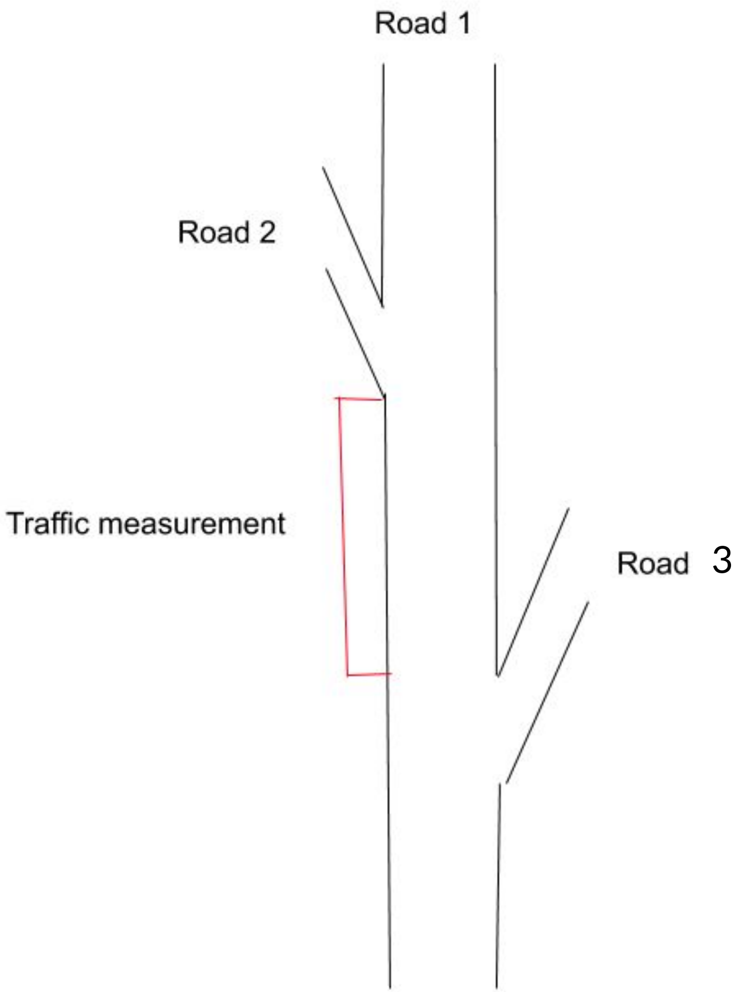


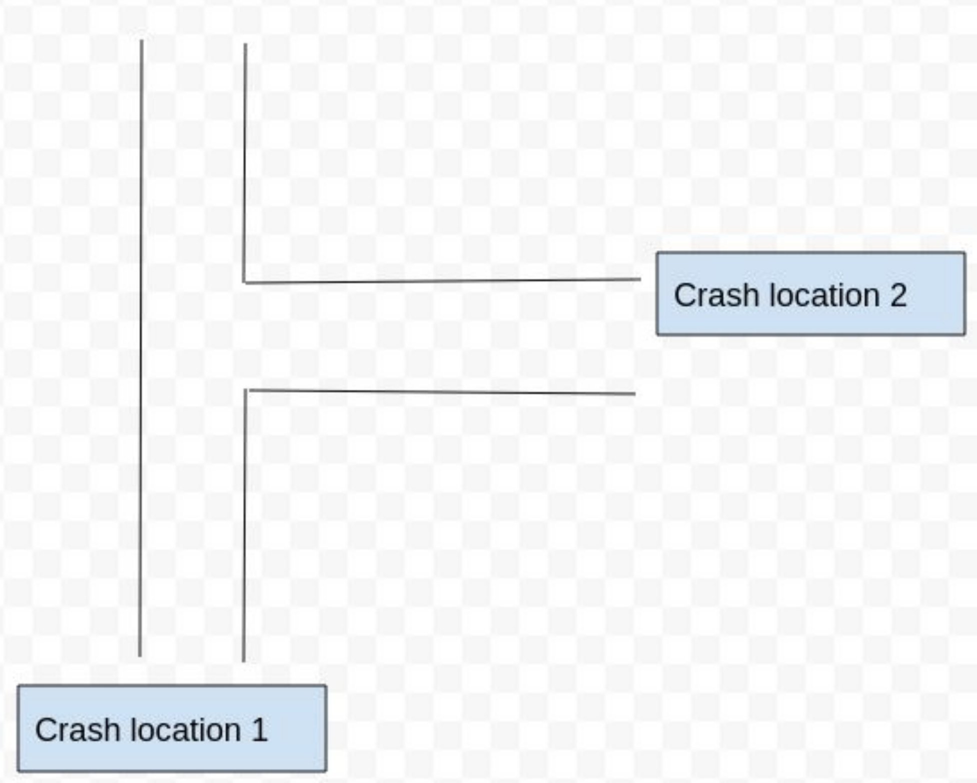
temporarySpeedLimit	tlaId	tlaName	trafficControl	trafficIsland	trafficSign	train	tree	truck	unknownVehicleType	urban	vanOrUtility	vehicle	waterRiver	weatherA	weatherB

Traffic Service CSV



X	Y	OBJECTID	carr_way_no	road_id	road_name	start_name	end_name	location	latest	count_date	peak_hour	count_duration	adt	peaktraffic	pccar	pclcv	pcmcv	pchcvi	pchcvii	pcbus	pcheavy	NZTMX	NZTMY

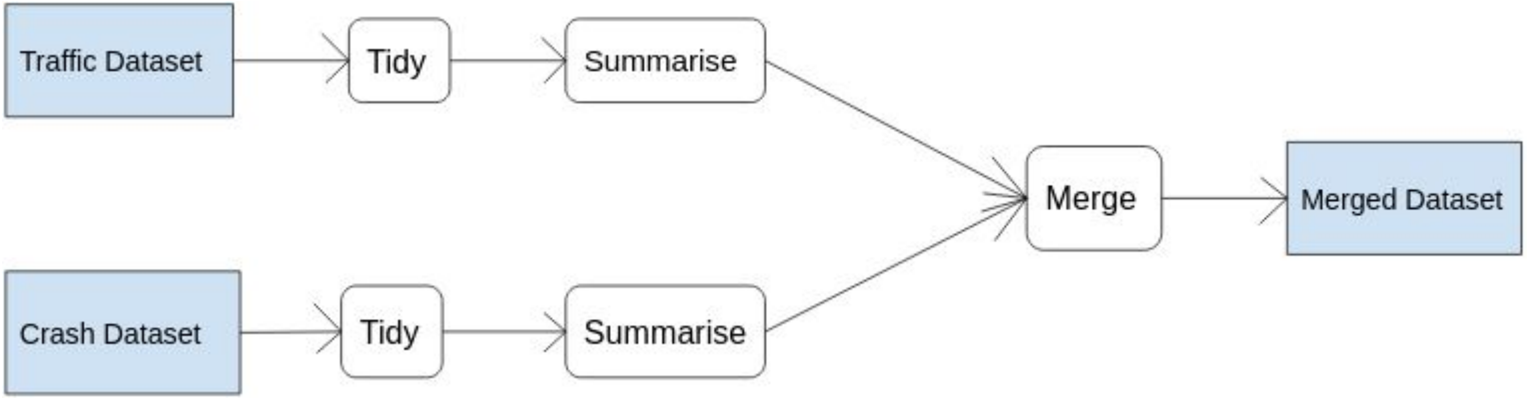

Road 1
Road 2
Traffic measurement
Road 3

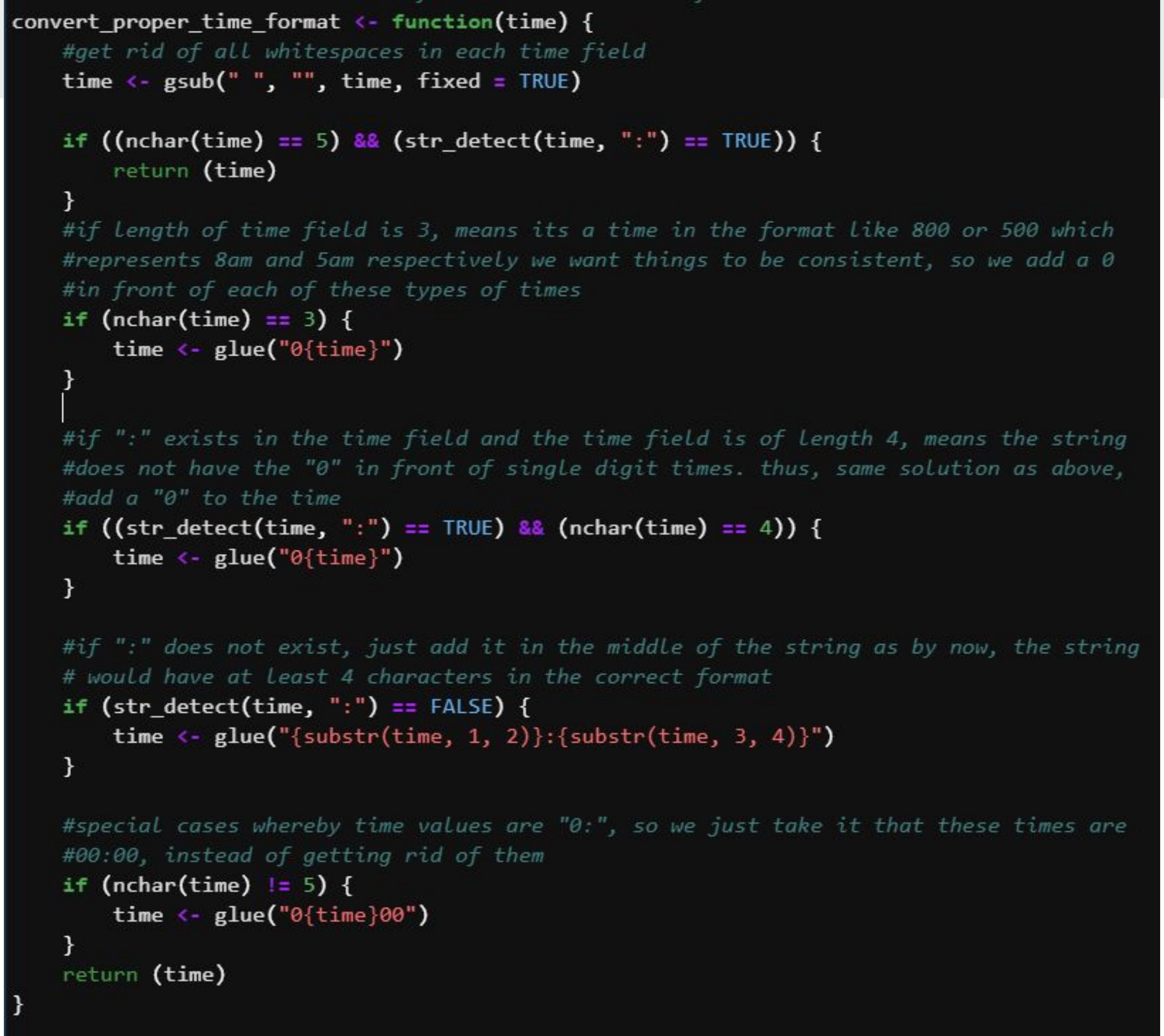

Crash location 2
Crash location 1

Our End Goals

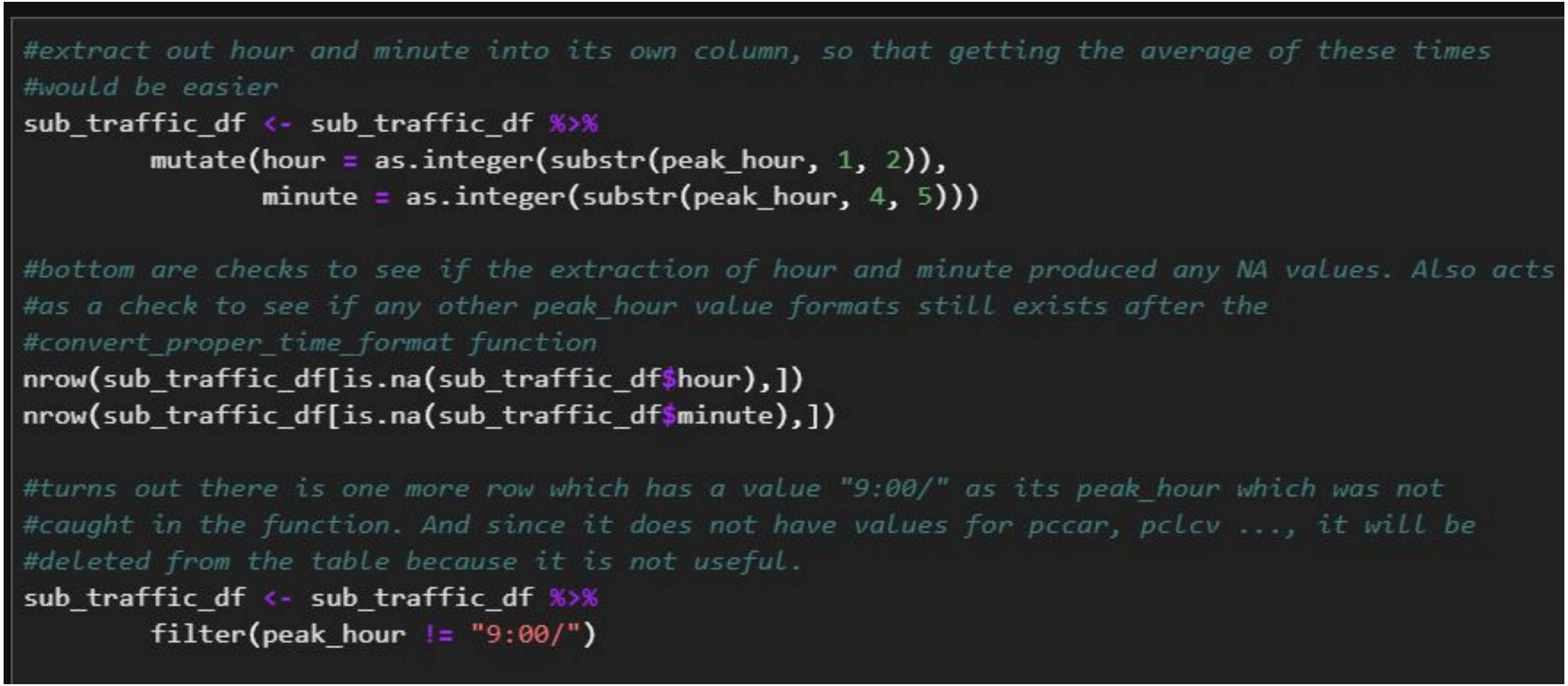
- Determine the number of Crashes for each Road Segment recorded in the Traffic Dataset.
- Determine most crash dense regions and graph the crashes over time for specifically problematic roads

Technical Difficulties

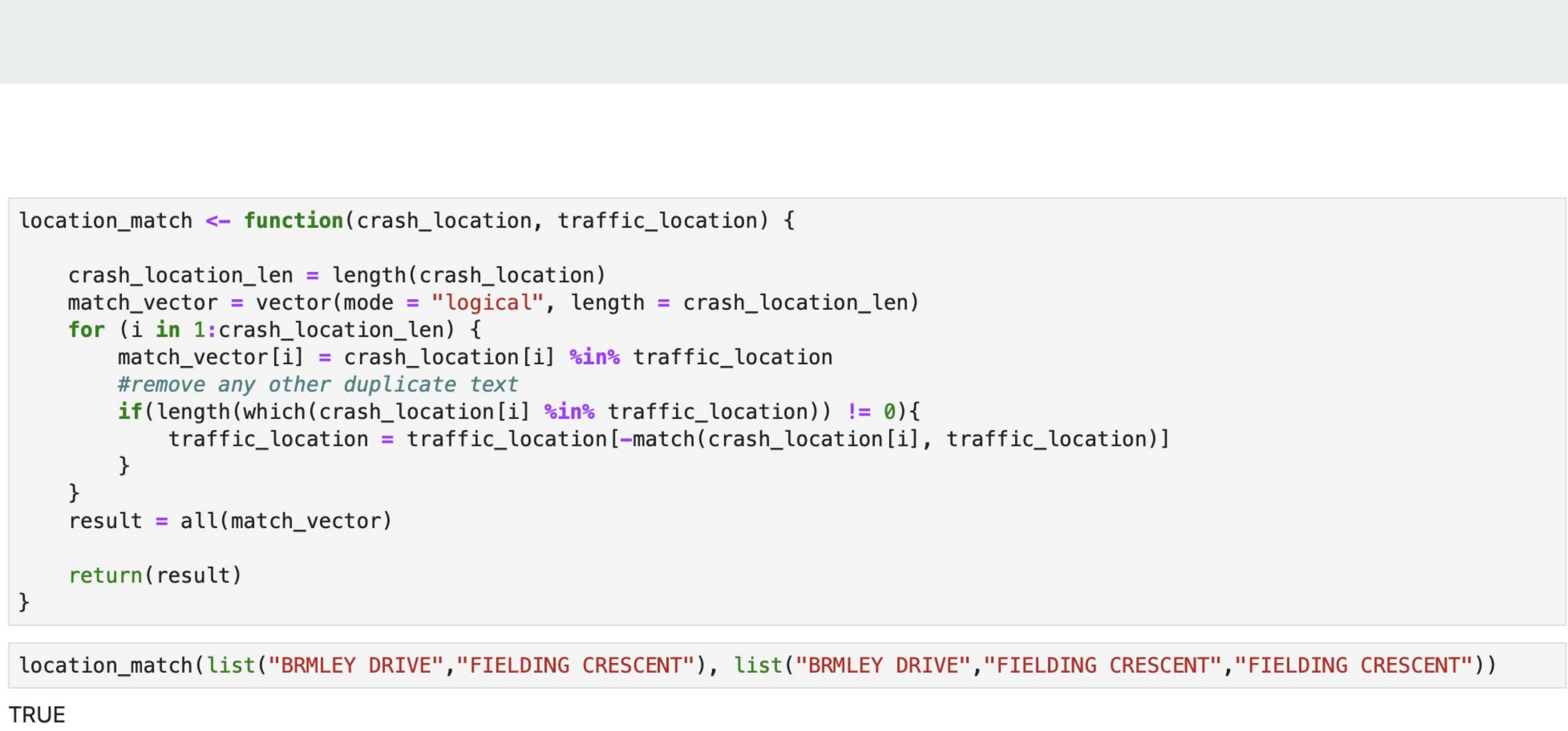




```r
convert_proper_time_format <- function(time) {
    #get rid of all whitespaces in each time field
    time <- gsub(" ", "", time, fixed = TRUE)

    if ((nchar(time) == 5) && (str_detect(time, ":") == TRUE)) {
        return (time)
    }
    #if length of time field is 3, means its a time in the format like 800 or 500 which
    #represents 8am and 5am respectively we want things to be consistent, so we add a 0
    #in front of each of these types of times
    if (nchar(time) == 3) {
        time <- glue("0{time}")
    }

    #if ":" exists in the time field and the time field is of length 4, means the string
    #does not have the "0" in front of single digit times. thus, same solution as above,
    #add a "0" to the time
    if ((str_detect(time, ":") == TRUE) && (nchar(time) == 4)) {
        time <- glue("0{time}")
    }

    #if ":" does not exist, just add it in the middle of the string as by now, the string
    # would have at least 4 characters in the correct format
    if (str_detect(time, ":") == FALSE) {
        time <- glue("{substr(time, 1, 2)}:{substr(time, 3, 4)}")
    }

    #special cases whereby time values are "0:", so we just take it that these times are
    #00:00, instead of getting rid of them
    if (nchar(time) != 5) {
        time <- glue("0{time}00")
    }
    return (time)
}
```



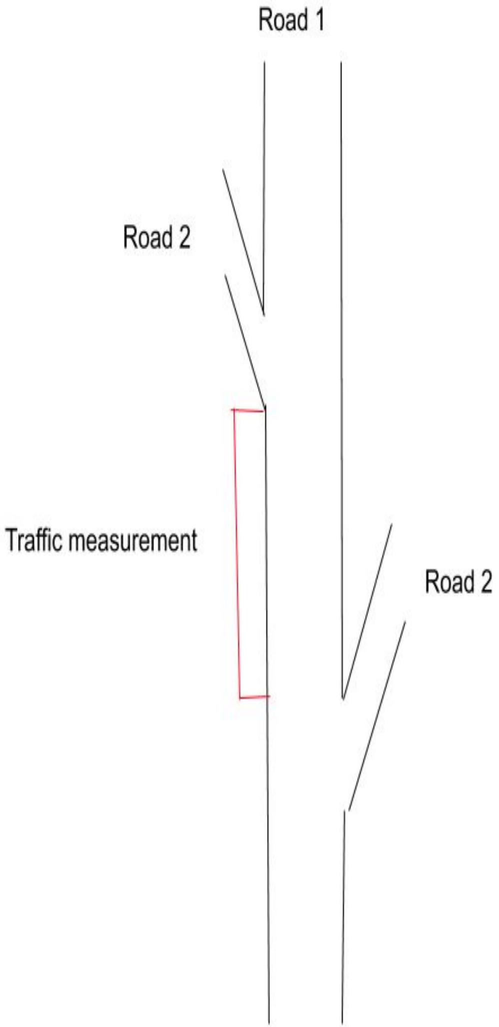
```
#extract out hour and minute into its own column, so that getting the average of these times
#would be easier
sub_traffic_df <- sub_traffic_df %>%
        mutate(hour = as.integer(substr(peak_hour, 1, 2)),
               minute = as.integer(substr(peak_hour, 4, 5)))

#bottom are checks to see if the extraction of hour and minute produced any NA values. Also acts
#as a check to see if any other peak_hour value formats still exists after the
#convert_proper_time_format function
nrow(sub_traffic_df[is.na(sub_traffic_df$hour),])
nrow(sub_traffic_df[is.na(sub_traffic_df$minute),])

#turns out there is one more row which has a value "9:00/" as its peak_hour which was not
#caught in the function. And since it does not have values for pccar, pclcv ..., it will be
#deleted from the table because it is not useful.
sub_traffic_df <- sub_traffic_df %>%
        filter(peak_hour != "9:00/")
```

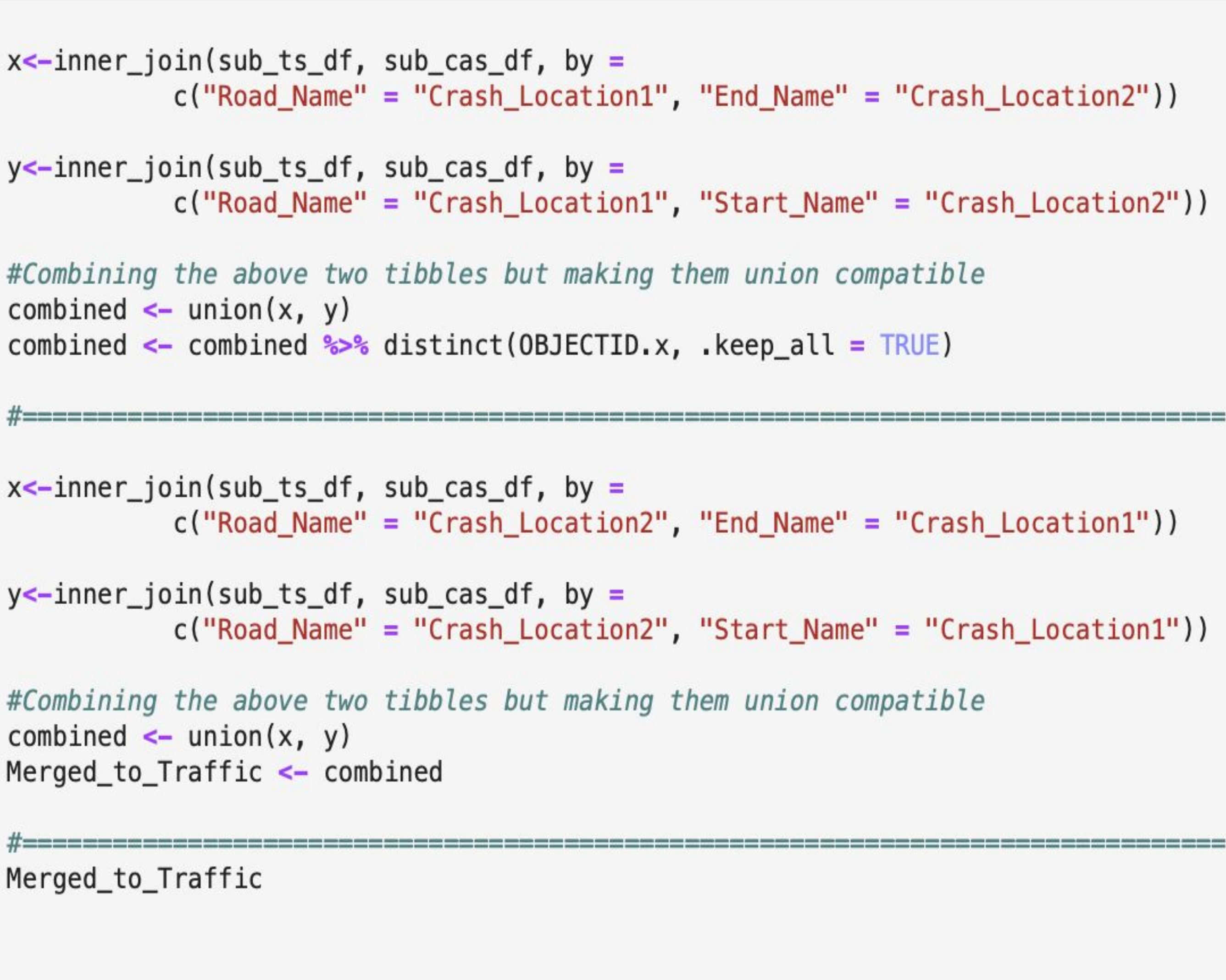


```
location_match <- function(crash_location, traffic_location) {

    crash_location_len = length(crash_location)
    match_vector = vector(mode = "logical", length = crash_location_len)
    for (i in 1:crash_location_len) {
        match_vector[i] = crash_location[i] %in% traffic_location
        #remove any other duplicate text
        if(length(which(crash_location[i] %in% traffic_location)) != 0){
            traffic_location = traffic_location[-match(crash_location[i], traffic_location)]
        }
    }
    result = all(match_vector)

    return(result)
}
```

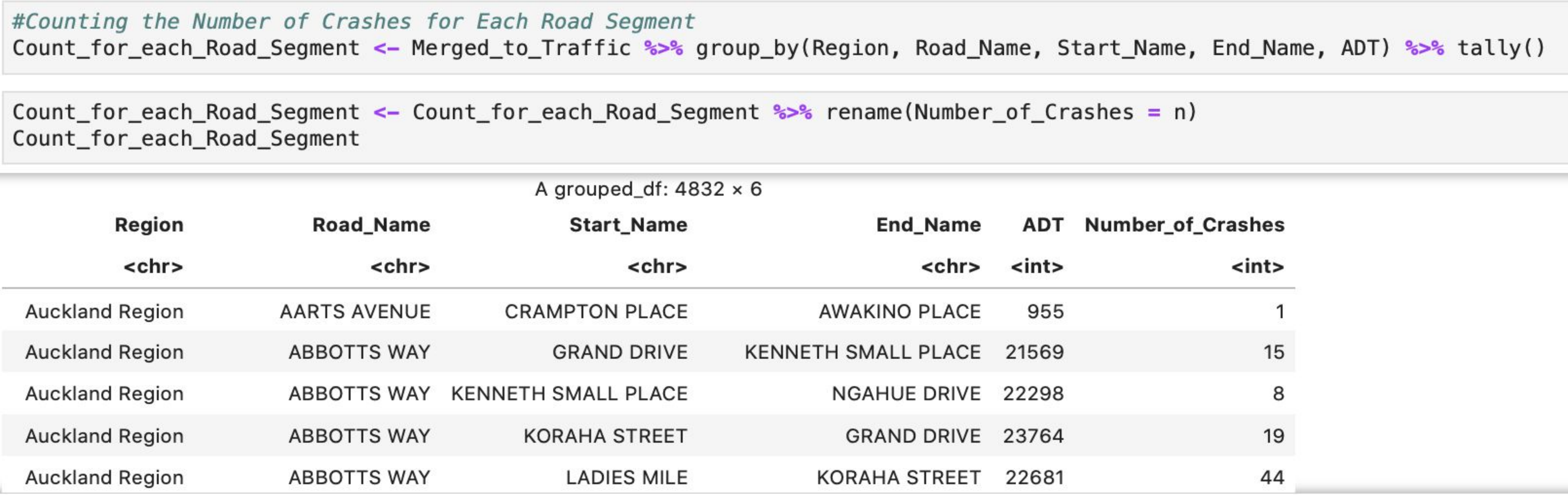
```
location_match(list("BRMLEY DRIVE","FIELDING CRESCENT"), list("BRMLEY DRIVE","FIELDING CRESCENT","FIELDING CRESCENT"))
```

```
TRUE
```





```r
x<-inner_join(sub_ts_df, sub_cas_df, by =
           c("Road_Name" = "Crash_Location1", "End_Name" = "Crash_Location2"))

y<-inner_join(sub_ts_df, sub_cas_df, by =
           c("Road_Name" = "Crash_Location1", "Start_Name" = "Crash_Location2"))

#Combining the above two tibbles but making them union compatible
combined <- union(x, y)
combined <- combined %>% distinct(OBJECTID.x, .keep_all = TRUE)

#==============================================================================

x<-inner_join(sub_ts_df, sub_cas_df, by =
           c("Road_Name" = "Crash_Location2", "End_Name" = "Crash_Location1"))

y<-inner_join(sub_ts_df, sub_cas_df, by =
           c("Road_Name" = "Crash_Location2", "Start_Name" = "Crash_Location1"))

#Combining the above two tibbles but making them union compatible
combined <- union(x, y)
Merged_to_Traffic <- combined

#==============================================================================
Merged_to_Traffic
```

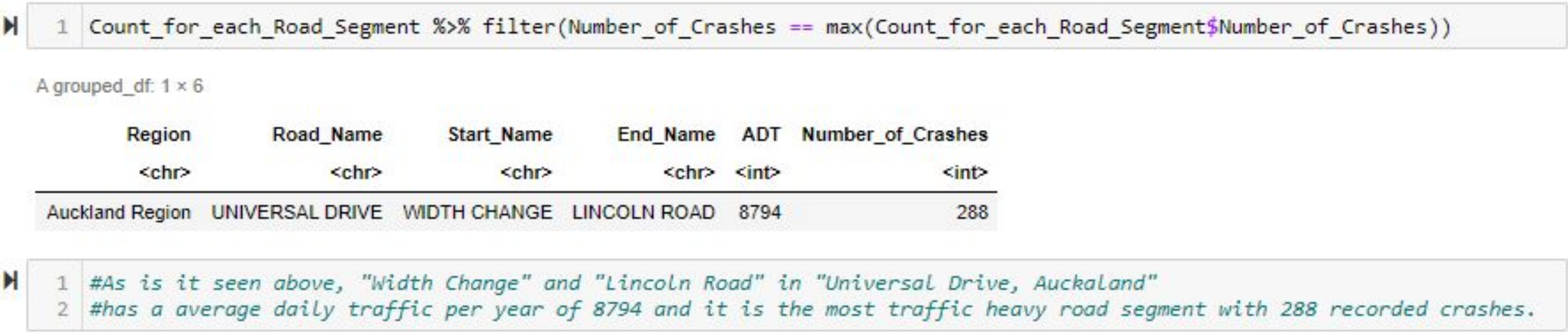


```r
#Counting the Number of Crashes for Each Road Segment
Count_for_each_Road_Segment <- Merged_to_Traffic %>% group_by(Region, Road_Name, Start_Name, End_Name, ADT) %>% tally()
```

```r
Count_for_each_Road_Segment <- Count_for_each_Road_Segment %>% rename(Number_of_Crashes = n)
Count_for_each_Road_Segment
```

A grouped_df: 4832 × 6

Region	Road_Name	Start_Name	End_Name	ADT	Number_of_Crashes
<chr>	<chr>	<chr>	<chr>	<int>	<int>
Auckland Region	AARTS AVENUE	CRAMPTON PLACE	AWAKINO PLACE	955	1
Auckland Region	ABBOTTS WAY	GRAND DRIVE	KENNETH SMALL PLACE	21569	15
Auckland Region	ABBOTTS WAY	KENNETH SMALL PLACE	NGAHUE DRIVE	22298	8
Auckland Region	ABBOTTS WAY	KORAHA STREET	GRAND DRIVE	23764	19
Auckland Region	ABBOTTS WAY	LADIES MILE	KORAHA STREET	22681	44

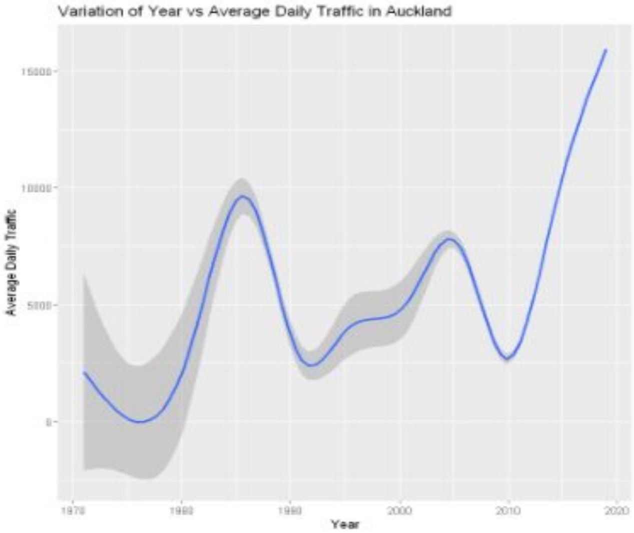


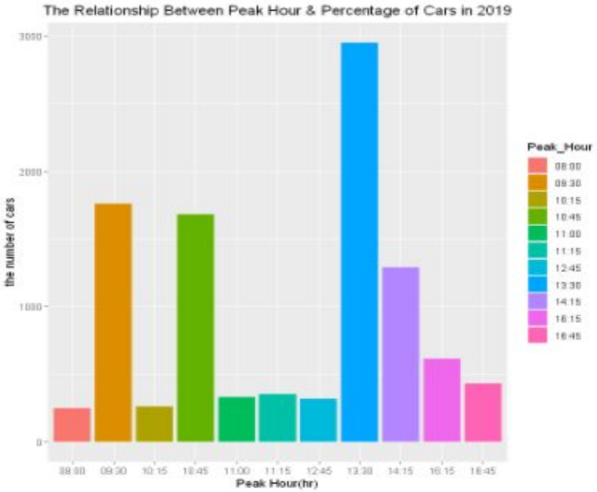
```
Count_for_each_Road_Segment %>% filter(Number_of_Crashes == max(Count_for_each_Road_Segment$Number_of_Crashes))
```

A grouped_df: 1 × 6

Region	Road_Name	Start_Name	End_Name	ADT	Number_of_Crashes
<chr>	<chr>	<chr>	<chr>	<int>	<int>
Auckland Region	UNIVERSAL DRIVE	WIDTH CHANGE	LINCOLN ROAD	8794	288

```
#As is it seen above, "Width Change" and "Lincoln Road" in "Universal Drive, Auckaland"
#has a average daily traffic per year of 8794 and it is the most traffic heavy road segment with 288 recorded crashes.
```


Variation of Year vs Average Daily Traffic in Auckland
Average Daily Traffic
Year


The Relationship Between Peak Hour & Percentage of Cars in 2019
the number of cars
08:00
09:30
10:15
10:45
11:00
11:15
12:45
13:30
14:15
16:15
16:45
Peak Hour(hr)
Peak_Hour
08:00
09:30
10:15
10:45
11:00
11:15
12:45
13:30
14:15
16:15
16:45