Group 4 project: Cab Rides Price Prediction Modelling

Please remember to change your files paths at beigining.

Summary

We started with cleaning the data sets and here are the steps:

1. For the dataset `cab_rides`, we selected the variables (distance, cab_type, time_stamp, destination, source, price, surge_multiplier, id, product_id and name), converted time_stamp to standard time format, got the hour of the day and deleted all the rows with NAs.
2. For the dataset `weather`, we selected the variables (temp, location, clouds, pressure, time_stamp, humidity and wind), got the hour of the day ,converted time_stamp to standard time format, and deleted all the rows with NAs.
3. We combined the two data sets by time and location.
4. At last, we used web crawler to improve our dataset and them make better price prediction model.
5. Then we explored the internal and external influence, and made linear models to predict the price by using multiple variables. For the internal influence, we draw the scatter plots and histograms of `distance`, `locations`, `time` and `carb type`; For the external influence, we draw the plots of `weather`. Finally, we used the multiple linear regression model to see how well it fitted our dataset by comparing the predicted values with the actual values. The prediction model shows that `distance`, `ride hour`, `rain`, and `surge multiplier` have positive influence to the cab price. By explore the relationship between three variables, we can look deeper about how one variable can influence the relationship between price and and multiple variables from multi-dimension.

Set Envoronment

At very begining, we set the R environment we needed.

```
require('devtools')
devtools::install_github("dkahle/ggmap")
require(lubridate)
require(readr)
require(tidyverse)
require(plotly)
require(gapminder)
require(ggthemes)
require(forcats)
require(stringr)
require(ggplot2)
require(ggmap)
require(httr)
require(xml2)
require(rvest)
require(magrittr)
require(dplyr)
require(formattable)
```

Data Cleaning

At the begining, we did the data cleaning process. Firstly, we loaded the raw data, and named two data sets with specific names. Secondly, we checked and corrected the data tpye. Thirdly, we omitted NA data.

```
# load data
cab_rides <- read.csv('cab_rides.csv')
weather <- read.csv('weather.csv')

# data type
## cab rides
cab_rides$cab_type <- as.character(cab_rides$cab_type)
cab_rides$destination <- as.character(cab_rides$destination)
cab_rides$source <- as.character(cab_rides$source)
cab_rides$id <- as.character(cab_rides$id)
cab_rides$product_id <- as.character(cab_rides$product_id)
cab_rides$name <- as.character(cab_rides$name)
cab_rides$time <- ymd_hms(as.POSIXct(cab_rides$time_stamp / 1000, origin = '1970-01-01'), tz = 'EST')
cab_rides$hour <- substr(cab_rides$time, 12, 13) %>% as.numeric()

##weather
weather$location <- as.character(weather$location)
weather$time <- ymd_hms(as.POSIXct(weather$time_stamp, origin = '1970-01-01'), tz = 'EST')
names(weather)[2] <- 'source'

# NA omit
## cab rides
cab_rides_omit <- na.omit(cab_rides)
any(is.na(cab_rides_omit))
```

```
## [1] FALSE
```

```
## weather
weather$rain[is.na(weather$rain)] <- 0
weather_omit <- na.omit(weather)
any(is.na(weather_omit))
```

```
## [1] FALSE
```

After the baisc data cleaning, we merged two data sets together by using variables `time` and `source`.

```
# merge
cab_weather <- merge(cab_rides_omit, weather_omit, by = c('time', 'source'))

# save
write.csv(cab_weather, file = 'cab_weather.csv')
write.csv(cab_rides_omit, file = 'cab_rides_omit.csv')
write.csv(weather_omit, file = 'weather_omit.csv')
```

At last, we saved the cleaned data as new csv files.

Below are our two cleaned datasets: `cab_Weather` and `cab_rides_omit`.

```
head(cab_weather)
```

```
head(cab_rides_omit)
```

Cab Type & Avg Price

```
cab_weather %>%

  select(price, distance, cab_type,name) %>%
  group_by(cab_type,name) %>%
  summarize(average_price = mean(price/distance)) %>%
  arrange(average_price) -> type_summary


#### Define a theme
n_theme <- theme_hc() + theme(
  plot.title = element_text(size = 14, face = 'bold',),
  axis.title.x = element_text(face = 'italic'),
  axis.title.y = element_text(face = 'italic'),
  legend.direction = 'vertical',
  legend.position = 'right',
  legend.title = element_text(face = 'bold'))

cab_type <- ggplot(type_summary)+
  geom_bar(aes(x = as.factor(name),y = average_price,fill = as.factor(cab_type)),stat = "identity")+
  labs(x = "Cabs",
       y = "The Average Price",
       fill = "Lyft/Uber",
       title = "Cab Type VS Average Price") + n_theme+theme(text = element_text(size = 10),
        axis.text.x=element_text(angle=45,hjust=1,vjust=0.9))

ggplotly(cab_type)
```

It's clear that different cab types would have different prices. We use bar chart to analyze the relationship between cab type and price. According to the plot, `Lux Black XL` has the highest average price, while `shared cab` has the lowest.

Price vs Distance

Read data from csv file for the further use

```
cab_rides_omit <- read_csv("cab_rides_omit.csv")
```

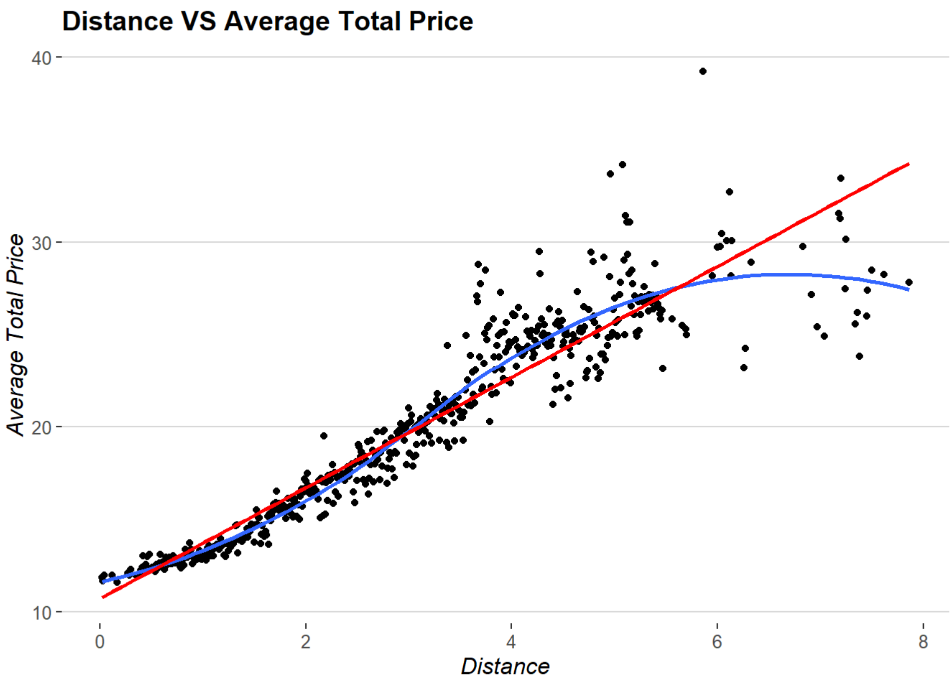
Then with the data we got the scatter plot of average total price vs `distance`. In this plot, we are showing the average total price for with different `distance`.

Average Total Price vs Distance

```
cab_rides_omit %>%
  select(price, distance) %>%
  group_by(distance) %>%
  summarize(average_totalprice =mean(price)) %>%
  arrange(average_totalprice) -> distance_summary

distance_plot <- ggplot(distance_summary,aes(x = distance, y = average_totalprice )) +
  geom_point()+
  geom_smooth(se=FALSE)+
  geom_smooth(method = "lm", se = FALSE, colour = 'red')+
  labs(title = "Distance VS Average Total Price",
     x="Distance",
     y="Average Total Price")  + n_theme

distance_plot
```



From the plot we see that as the distance of trip increases, the overall price(in average) for each trip increases. But the increment is found to slow down. Therefore, we hypothesized that the average price `per mile` should decrease with distance. To prove this we made a second plot.

Average Price vs Distance

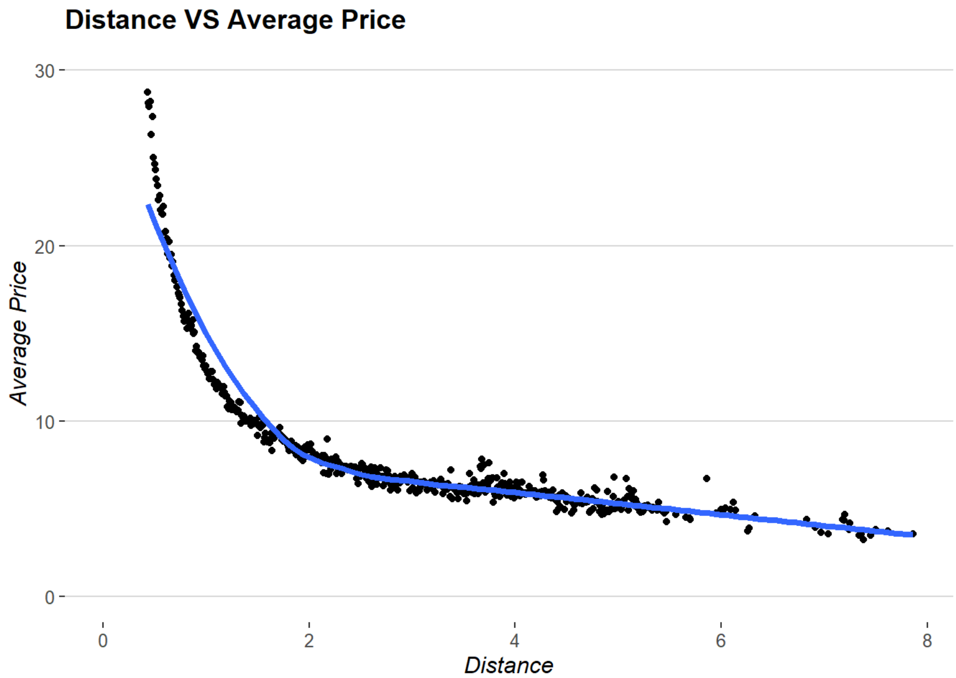
```
cab_rides_omit %>%
  select(price, distance) %>%
  group_by(distance) %>%
  summarize(average_price =mean(price/distance)) %>%
  arrange(average_price) -> distance_summary2
?geom_smooth

distance_plot2 <- ggplot(distance_summary2,aes(x = distance, y = average_price)) +
  geom_point()+
  geom_smooth(se=FALSE,size=1.5)+ n_theme +
  labs(title = "Distance VS Average Price",
       x="Distance",
       y="Average Price") +
  ylim(0,30)  + n_theme

distance_plot2
```

```
## Warning: Removed 14 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 14 rows containing missing values (geom_point).
```

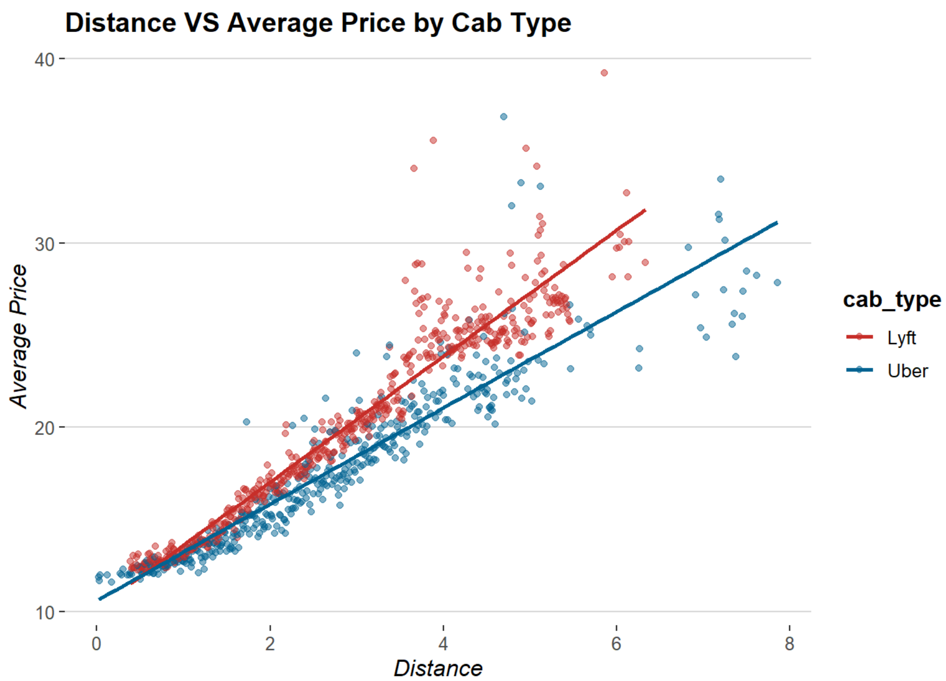


From this one we can see that the average price `per mile` is going down as the distance increases which is also supporting the observations from the last plot. In our data set, we have two different companies, Uber and Lyft, we compaired these two with catter plots also.

Price VS Distance & Cab type

```
cab_rides_omit %>%
  select(price, distance,cab_type) %>%
  group_by(distance,cab_type) %>%
  summarize(average_price =mean(price)) %>%
  arrange(average_price) -> distance_type_summary

distance_type_plot <- ggplot(distance_type_summary,aes(x = distance, y = average_price,col=cab_type)) +
  geom_point(alpha=0.5)+
  geom_smooth(method = "glm",se=FALSE)+
  labs(title = "Distance VS Average Price by Cab Type",
       x="Distance",
       y="Average Price") + n_theme + scale_color_wsj()
distance_type_plot
```



We can see that the Lyft has a larger slop which means it has a higher increase rate, as the distance increases, the Lyft's price increases faster than Uber's.

Price vs 24hr Time Range/Destination

Average Price vs 24hr Time Range/Destination

```
cab_rides_omit %>%
  select(price, hour) %>%
  group_by(hour) %>%
  summarize(average_price = mean(price)) %>%
  arrange(hour) -> hours_summary
```

We took `price` and `hour` (24hr time range) as the first two variables to compare from the dataset, and group them by `hour`. In this bar chart, we defined `average price` as the `mean` of the cab price, which is a little different from last time that we use price divided by `distance` as the average price.

```
hours_plot <- ggplot(hours_summary, aes(x = hour, y = average_price)) +
  geom_bar(stat = "identity", show.legend = FALSE, fill = "blue") +
  coord_cartesian(ylim = c(16, 16.75)) +
  labs(title = "24hr VS Average Price",
       x = "24hr",
       y = "Average Price")

ggplotly(hours_plot)
```

We drew bar charts to analyze the relationship between average cab price and 24hr time range. From the chart we can tell, the average price for the whole day ranges from 16.48 to 16.61, and the highest happened around 5pm which is during the rush hour; while the lowest happened around 12pm which is when most people took their lunch break and the need for Uber/Lyft would be lower.

```
cab_rides_omit %>%
  select(price, hour, destination) %>%
  group_by(hour, destination) %>%
  summarize(average_price = mean(price)) %>%
  arrange(hour) -> hours_summary2
```

Then we add `destination` as the third variable to compare together with the previous two variables, and group them by `hour` and `destination`.

```
hours_plot2 <- ggplot(hours_summary2, aes(x = hour, y = average_price, col = destination)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ destination) +
  coord_cartesian(ylim = c(13.5, 19.5)) +
  labs(title = "24hr&Destination VS Average Price",
       x = "24hr",
       y = "Average Price") +  n_theme
ggplotly(hours_plot)
```

We drew another bar chart to analyze the relationship between average cab price and 24hr time range/destination. From the chart we can tell, Boston University and Northeastern University has the highest average price no matter what time it is; and Haymarket Square/South Station has the lowest average price. The overall highest average price will be 19.4 around 2am heading to Boston University; while the lowest will be 14.05 around midnight heading to Haymarket Square. The average price didn't float too much based on the destination but have slightly difference around peak hour and slack hour.

Average Price VS Hour & Distance

```
cab_rides_omit %>%
  select(price,distance,hour) %>%
  group_by(distance,hour) %>%
  summarize(average_price = mean(price)) %>%
  arrange(hour) -> dis_hr_pr

movingplot <- dis_hr_pr %>%
  plot_ly(
    x = ~distance,
    y = ~average_price,
    color = ~as.factor(hour),
    frame = ~hour,
    hoverinfo = "text",
    type = 'scatter',
    mode = 'markers'
  )

movingplot
```

```
This is the moving plot which shows the scatter plot of average price and distance, and changes by hours (24hr time ran
```

Price vs Geographic Locations

Data transformation

```
class(cab_rides_omit$destination)
```

```
## [1] "character"
```

```
cab_rides_omit$destination <- as.factor(cab_rides_omit$destination)
cab_rides_omit$source <- as.factor(cab_rides_omit$source)
```

Overview of `source` and `destination`

```
# construct the table for overviewing source & destination
fct_count(cab_rides_omit$source)
```

```
fct_count(cab_rides_omit$destination)
```

```
# bar plots for overviewing source & destination
cab_rides_omit %>%
  ggplot(aes(x = destination)) +
  geom_bar(aes(fill = destination),show.legend = F) + theme(text = element_text(size=10),  axis.text.x=element_
text(angle=45,hjust=1,vjust=0.9,size=13)) +
  coord_cartesian(ylim = c(40000,55000)) +
  n_theme -> plot_d

ggplotly(plot_d)
```

```
cab_rides_omit %>%
  ggplot(aes(x = source)) +
  geom_bar(aes(fill = source),show.legend = F) +
  theme(text=element_text(size=10),
        axis.text.x=element_text(angle=45,hjust=1,vjust=0.9,size=13)) +
  coord_cartesian(ylim = c(40000,55000)) +
  n_theme -> plot_s
ggplotly(plot_s)
```

Average price vs source and destination

Last time, for the relationship between `price` and location, we defined `average price` as the sum of `price` / `the total number of trips` for each different geographic location. Tables are recreated in r code below:

```
cab_rides_omit %>%
  select(price, distance,source) %>%
  group_by(source) %>%
  summarize(avg_priceForTrips = mean(price)) %>%
  arrange(desc(avg_priceForTrips)) %>% formattable
```

source	avg_priceForTrips
Boston University	18.85303
Fenway	18.37949
Financial District	18.18137
Northeastern University	17.90112
Theatre District	16.59699
North Station	16.36401
West End	16.10850
Back Bay	16.04739
South Station	15.67248
Beacon Hill	15.66403
North End	15.15337
Haymarket Square	13.57811

```
cab_rides_omit %>%
  select(price, distance,destination) %>%
  group_by(destination) %>%
  summarize(avg_priceForTrips=mean(price)) %>%
  arrange(desc(avg_priceForTrips)) %>% formattable
```

destination	avg_priceForTrips
Boston University	18.94214
Fenway	18.14642
Financial District	18.04628
Northeastern University	17.82752
North Station	16.80524
Beacon Hill	16.24833
West End	16.22584
Back Bay	16.21015
Theatre District	15.97445
North End	15.00221
South Station	14.82855
Haymarket Square	14.25555

Similar to the results in sql, for both source and destination, `Boston University` has the highest `average_price` based upon the total number of trips for each location. It implies that the trips of people (mostly students) who take uber or lyft from or to BU tend to be more expensive trips. One possible explanation is that most BU students live far away from the campus, so that the average price for trips is high.

Instead of just focusing on the average price computed by the sum of `price` / `the number of trips`, we defined `avg_price` as `price` / `distance`, which is the price per mile, to better investigate the relationship among `price`, `distance`, and the geographic locations.

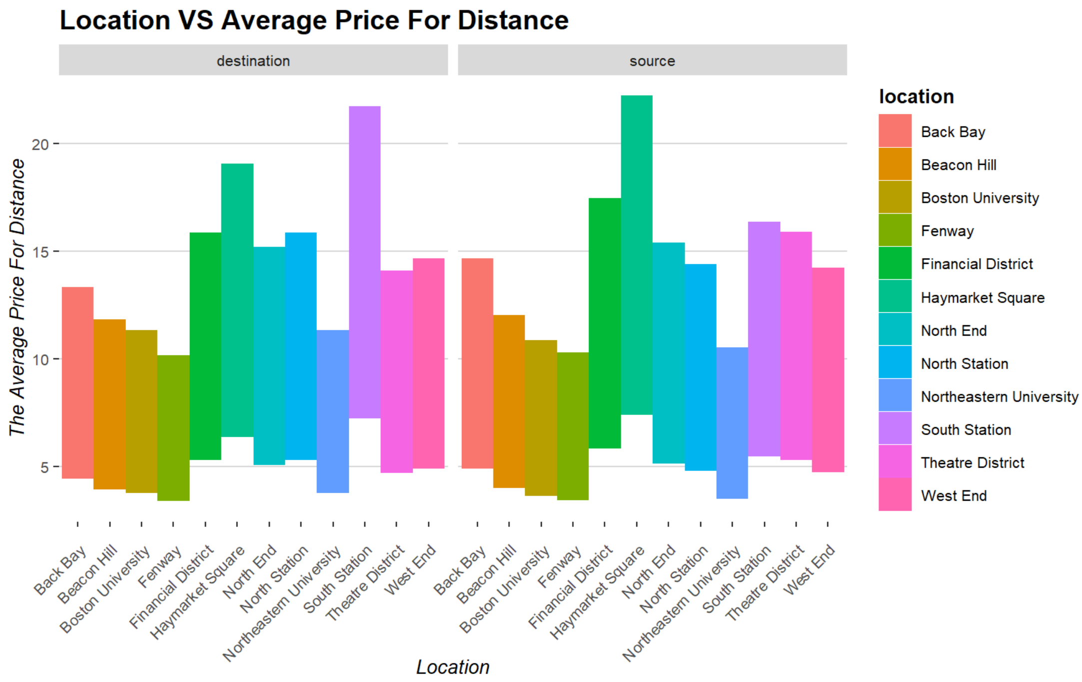
```
cab_rides_omit %>%
  select(price, distance,source,cab_type) %>%
  group_by(source) %>%
  summarize(avg_price=mean(price/distance)) %>%
  arrange(desc(avg_price)) -> summary1

cab_rides_omit %>%
  select(price, distance,destination,cab_type) %>%
  group_by(destination) %>%
  summarize(avg_price=mean(price/distance)) %>%
  arrange(desc(avg_price))->summary2


sums<-rename(summary1, 'location'='source')
sums$id<-'source'
sumd<-rename(summary2,'location'='destination')
sumd$id<-'destination'
summary4 <- rbind(sums,sumd)
summary4$id<-as.factor(summary4$id)

sum4 <-  ggplot(summary4)+
  geom_tile(aes(x=location,y=avg_price,height=avg_price,fill=location),stat = "identity", show.legend=T) +
  labs(x="Location",
       y="The Average Price For Distance",
       title = "Location VS Average Price For Distance") +n_theme+
   theme(text=element_text(size=10),axis.text.x=element_text(angle=45,hjust=1,vjust=0.9),legend.background=elem
ent_rect(size=0.3)) +
    facet_wrap(~id)

sum4
```




```
ggplotly(sum4)
```

From two faceted tile graphs above, for starting point (`source`), two locations that have the highest average price per mile are `Haymarket square` and `Financial District` , whereas for terminal (`destination`), two location that have the highest average price per mile are `South Station` and `Haymarket square` .

To dig into the relationship of price and locations, this time we also concate `source` and `destination` to construct a new variable `route` , which stores the routes of each trip in the overall dataset.

We constrct the bar plot and label the y-axis as locations and x-axis as average price per mile.

```
cab_rides_omit %>%
  transmute(source, destination, price, distance,
            route=paste(source,destination,sep = '-')) %>%
    group_by(route) %>%
  select(route, price, distance, source, destination) %>%
  summarize(avg_priceForDistance=mean(price/distance)) %>%
  arrange(desc(avg_priceForDistance)) -> summary5

formattable(summary5)
```

route	avg_priceForDistance
Financial District-South Station	29.583947
Haymarket Square-North Station	24.184915
Theatre District-South Station	23.874688
Haymarket Square-West End	20.964050
South Station-Financial District	20.821795
North Station-Haymarket Square	19.573905
West End-Haymarket Square	18.344484
Back Bay-Boston University	14.240548
South Station-Theatre District	13.361892
Boston University-Back Bay	13.215032
Haymarket Square-Financial District	13.167127
North End-North Station	12.751336
West End-North End	12.672485
North Station-North End	12.291386
North End-Financial District	12.153508
Back Bay-Northeastern University	11.777078
Haymarket Square-Beacon Hill	11.769144
Financial District-Haymarket Square	11.579812
Haymarket Square-Theatre District	11.297438
Financial District-North End	11.249975
North End-West End	11.249817
Beacon Hill-Haymarket Square	10.230469
North End-Theatre District	10.212765
Back Bay-Fenway	9.883102
Northeastern University-Back Bay	9.345640
Fenway-Back Bay	9.240733
Theatre District-North End	9.160522
Theatre District-Haymarket Square	9.131260

route	avg_priceForDistance
Beacon Hill-North End	8.794343
North End-Beacon Hill	8.653441
Back Bay-South Station	8.397657
South Station-West End	8.367170
Theatre District-Northeastern University	8.221310
South Station-North Station	8.052346
West End-South Station	7.926349
Northeastern University-Theatre District	7.855436
Haymarket Square-Back Bay	7.789728
North Station-South Station	7.659300
Back Bay-Haymarket Square	7.639394
Beacon Hill-South Station	7.553671
Beacon Hill-Northeastern University	7.373532
Beacon Hill-Boston University	7.177543
North End-Back Bay	7.017469
Back Bay-North End	6.968452
Fenway-Theatre District	6.957144
Beacon Hill-Fenway	6.894926
South Station-Back Bay	6.874539
South Station-Beacon Hill	6.858686
Fenway-Beacon Hill	6.843079
Northeastern University-Beacon Hill	6.685368
Boston University-Theatre District	6.646693
Theatre District-Fenway	6.516223
Fenway-West End	6.497008
Theatre District-Boston University	6.474455
West End-Boston University	6.409994
Northeastern University-West End	6.364574
Boston University-Beacon Hill	6.346749
Fenway-North Station	6.241733
West End-Fenway	6.176099
Northeastern University-North Station	6.176050
West End-Northeastern University	6.175555
North Station-Boston University	6.157269
Boston University-West End	6.063460
North Station-Fenway	5.963193
North Station-Northeastern University	5.917999
Boston University-North Station	5.883060
Financial District-Northeastern University	5.855642
Northeastern University-Financial District	5.633071
Fenway-Financial District	5.529593
Boston University-Financial District	5.382756
Financial District-Fenway	5.251419
Financial District-Boston University	5.027223

```
rou <- ggplot(summary5,aes(x = route,
                           y = avg_priceForDistance))+
  geom_bar(stat = "identity", fill = 'royalblue', show.legend = F)+
  labs(x = "Routes",
       y = "The Average Price For Distance",
       title = " Routes VS Average Price") + n_theme+
  theme(text=element_text(size=6.8),
        axis.text.x=element_text(angle=90,hjust=1)) +
  coord_flip()

ggplotly(rou,width=1)
```

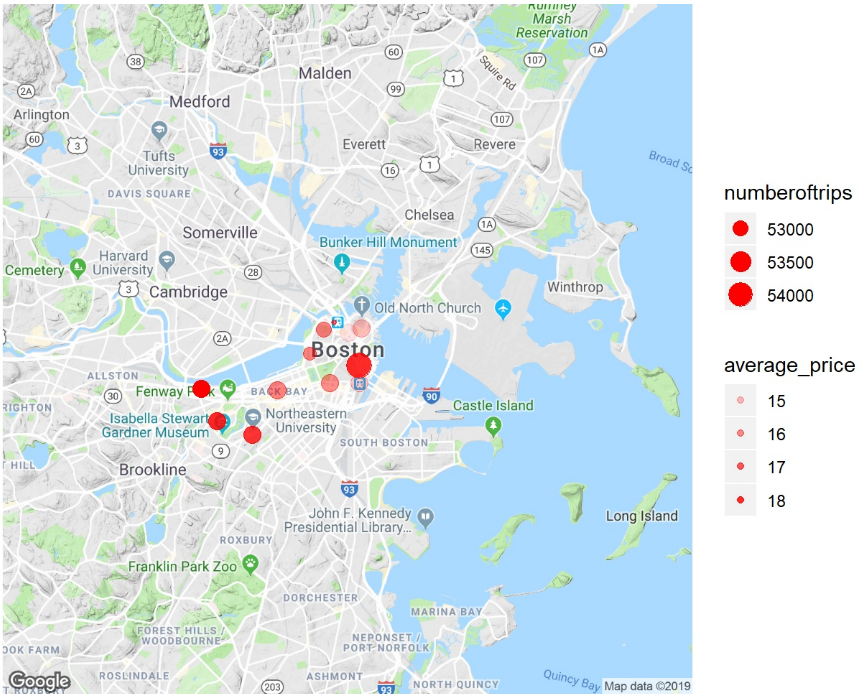
From the plot, `Financial District` to `South Station` and `Theatre District` to `South Station` have the most highest average price per mile. From geographic map, these two locations are adjacent, which contradicts our assumption that the longer the route, the more expensive the price per mile would be, since customers need to pay toll for long routes or pay more to find a driver that will be willing to travel in a long distance. In other words, when the route is very short, the average price per mile would be high, since the total cost of the trip is smaller due to the limited miles for some short routes, and the result is that no so many drivers would like to drive. To induce drivers, Uber and Lyft rate some very short routes more expensive.

Map

```
library(ggmap)
map <- read.csv('destination_MAP_summary.csv')

register_google(key = 'AIzaSyAGhQYCuw8X4OXxDlafFUqhvz3fMSuT_E8')
area <- geocode('Boston, MA')
Boston_map <- qmap(c(long = area$lon, lat = area$lat), zoom = 12)

g.map <- Boston_map +
          geom_point(data = map,
                     aes(x = lat, y = long,
                         size = numberoftrips,
                         alpha = average_price),
                     col = 'red')
g.map
```



Weather Analysis

Humidity

We computed the average price of (price/distance) by humidity, then draw a scatter plot.

```
cab_weather <- read_csv("cab_weather.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   time = col_datetime(format = ""),
##   source = col_character(),
##   cab_type = col_character(),
##   destination = col_character(),
##   id = col_character(),
##   product_id = col_character(),
##   name = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
cab_weather %>%
  select(price, distance, humidity) %>%
  group_by(humidity) %>%
  summarize(average_price = mean(price/distance)) %>%
  arrange(average_price) -> humidity_summary

humidity_plot <- ggplot(humidity_summary,aes(x = humidity, y = average_price)) +
  geom_point(col = 'red',alpha = 0.4)+
  xlim(0.55,1)+
  ylim(0,28)+
  geom_smooth(method = "glm",se = F)+
  labs(title = "Humidity VS Average Price",
       x = "Humidity",
       y = "Average Price")

ggplotly(humidity_plot)
```

This scatter plot shows most data points are distributed on a horizontal line, thus we conclude there is nonlinear relationship between humidity and average price.

Wind

We computed the average price of (price/distance) by wind, deleted one extreme value and then draw a scatter plot.

```
cab_weather %>%
  select(price, distance, wind) %>%
  group_by(wind) %>%
  summarize(average_price = mean(price/distance)) %>%
  arrange(average_price) -> wind_summary
head(wind_summary)
```

```
wind_plot <- ggplot(wind_summary,aes(x = wind, y = average_price,)) +
  geom_point(alpha = 0.4, col = 'red')+
  geom_smooth(method = "glm",se=FALSE)+
  labs(title = "Wind VS Average Price",
       x = "Wind",
       y = "Average Price")+
   scale_colour_gradient(low = "white", high = "black") +
  ylim(0,40)

ggplotly(wind_plot)
```

Based on this scatter plot, it seems there is nonlinear relationship between wind and average price.

Temperature

We computed the average price of (price/distance) by temperature, deleted one extreme value and then drew a scatter plot.

```
cab_weather_n <- read.csv("cab_weather_new.csv")

cab_weather_n %>%
  select(price, distance, temp) %>%
  group_by(temp) %>%
  summarize(average_price = mean(price/distance)) %>%
  arrange(average_price) -> temp_summary


temp_plot <- ggplot(temp_summary,aes(x = temp, y = average_price)) +
  geom_point(alpha = 0.4, col = 'red')+
  geom_smooth(method = "glm",se = FALSE)+
  labs(title = "Temperature VS Average Price",
       x = "Temperature",
       y = "Average Price")+
  ylim(0,30)
ggplotly(temp_plot)
```

Based on this scatter plot, most data points are distributed mostly at (8,40) and it seems that there is no relationship between the average price and temperature. We also added a trend line to see the relationship but the line seems like very horizontal.

Pressure

We computed the average price of (price/distance) by pressure, deleted one extreme value and then draw a scatter plot.

```
cab_weather_n %>%
  select(price, distance, pressure) %>%
  group_by(pressure) %>%
  summarize(average_price = mean(price/distance)) %>%
  arrange(average_price) -> pressure_summary


pressure_plot <- ggplot(pressure_summary,aes(x = pressure, y = average_price)) +
  geom_point(alpha = 0.4, col = 'red')+
  geom_smooth(method = "glm",se=FALSE)+
  labs(title = "Pressure VS Average Price",
       x = "Pressure",
       y = "Average Price")+
  ylim(0,40)

ggplotly(pressure_plot)
```

We added a trendline on this scatter plot but it seems like there is no relationship between avergae price and pressure since most data points are distributed in a square so we conclude there is nonlinear relationship between pressure and average price.

Rain

```
cab_weather_n %>%
  select(price, distance, rain) %>%
  group_by(rain) %>%
  summarize(average_price = mean(price/distance)) %>%
  arrange(average_price) -> rain_summary


rain_plot <- ggplot(rain_summary,aes(x = rain, y = average_price)) +
  geom_point(alpha = 0.4, col = 'red')+
  geom_smooth(method = "glm",se = FALSE)+
  labs(title = "Rain VS Average Price",
       x = "Rain",
       y = "Average Price")+
  ylim(0,20)
ggplotly(rain_plot)
```

Based on this scatter plot, it seems a week positive realtionship between rain and average price. In addition, we added a linear trend line which is y=5.187*(10^-3)x + 0.163.

Predictive Modelling For Cab Rides Price

Web Crawler

To expand our datasets of weather, we decided to do web crawler by using ‘httr’ and ‘rvest’ packages.

After observed the structures of some weather report websites, we chose DARK SKY to do the crawler, since this website has the most weather information. We could do a easy web crawler of one website by the following code

```r
# get URL
weather_url <- 'https://darksky.net/details/42.3523,-71.1214/2018-12-1/us12/en'

# Raw Data
weather_raw <- weather_url %>% GET() %>% read_html(encoding = 'UTF-8')
weather_info_raw <- weather_raw %>% html_nodes("script") %>% html_text()

# Draw Data
weather_info_vec <- weather_info_raw %>% strsplit(',\\{') %>% unlist
result_df <- NULL
for(i in seq(length(weather_info_vec))){
  weather_info_vec2 <- unlist(strsplit(weather_info_vec[i], ","))[1:17]
  weather_info_vec3 <- gsub('\"', "", weather_info_vec2)
  name_vec <- str_match(weather_info_vec2, '\\\"(.*?)\\\":')[,2]
  val_vec  <- gsub('\"|}', "",str_match(weather_info_vec2, ':(.*)')[,2])
  result_df <- rbind(result_df, val_vec)
  colnames(result_df) <- name_vec
}
print(result_df)
```

```
##         time         summary         icon                  precipIntensity
## val_vec "1543640400" "Mostly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543644000" "Mostly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543647600" "Overcast"      "cloudy"              "0"
## val_vec "1543651200" "Overcast"      "cloudy"              "0"
## val_vec "1543654800" "Overcast"      "cloudy"              "0"
## val_vec "1543658400" "Mostly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543662000" "Partly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543665600" "Mostly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543669200" "Partly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543672800" "Mostly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543676400" "Clear"         "clear-day"           "0"
## val_vec "1543680000" "Partly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543683600" "Partly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543687200" "Partly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543690800" "Partly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543694400" "Mostly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543698000" "Partly Cloudy" "partly-cloudy-day"   "0"
## val_vec "1543701600" "Partly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543705200" "Partly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543708800" "Clear"         "clear-night"         "0"
## val_vec "1543712400" "Clear"         "clear-night"         "0"
## val_vec "1543716000" "Partly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543719600" "Partly Cloudy" "partly-cloudy-night" "0"
## val_vec "1543723200" "Overcast"      "cloudy"              "0"
##         precipProbability temperature apparentTemperature dewPoint
## val_vec "0"               "33.97"     "33.97"             "27.52"
## val_vec "0"               "34.19"     "34.19"             "27.48"
## val_vec "0"               "34.08"     "34.08"             "28"
## val_vec "0"               "33.83"     "30.15"             "27.58"
## val_vec "0"               "32.87"     "29.64"             "26.92"
## val_vec "0"               "31.75"     "28.09"             "26.51"
## val_vec "0"               "31.85"     "28.14"             "26.51"
## val_vec "0"               "31.95"     "28.22"             "26.15"
## val_vec "0"               "33.69"     "28.85"             "27.8"
## val_vec "0"               "35.76"     "30.95"             "27.01"
## val_vec "0"               "38.59"     "35.73"             "27.5"
## val_vec "0"               "39.3"      "36.52"             "27.12"
## val_vec "0"               "41.09"     "41.09"             "27.44"
## val_vec "0"               "43.26"     "43.26"             "27.76"
## val_vec "0"               "43.9"      "43.9"              "27.83"
## val_vec "0"               "42.66"     "42.66"             "27.99"
## val_vec "0"               "41.01"     "41.01"             "28.71"
## val_vec "0"               "38.69"     "38.69"             "29.21"
## val_vec "0"               "37.05"     "37.05"             "29.35"
## val_vec "0"               "35.29"     "35.29"             "29.17"
## val_vec "0"               "35.03"     "35.03"             "29.87"
## val_vec "0"               "35.22"     "35.22"             "30.67"
## val_vec "0"               "36.57"     "36.57"             "31.05"
## val_vec "0"               "36.52"     "36.52"             "31.75"
##         humidity pressure  windSpeed windGust windBearing cloudCover
## val_vec "0.77"   "1018.68" "2.14"    "2.14"   "299"       "0.85"
## val_vec "0.76"   "1018.93" "2.81"    "2.81"   "290"       "0.85"
## val_vec "0.78"   "1019.57" "2.66"    "2.66"   "297"       "0.98"
## val_vec "0.78"   "1019.71" "4.03"    "4.03"   "304"       "1"
## val_vec "0.79"   "1019.44" "3.51"    "3.89"   "310"       "0.98"
## val_vec "0.81"   "1019.95" "3.73"    "3.73"   "313"       "0.53"
## val_vec "0.8"    "1021.75" "3.78"    "3.78"   "308"       "0.4"
```

```
## val_vec "0.79"    "1021.72" "3.82"    "4.11"   "326"       "0.74"
## val_vec "0.79"    "1022.22" "5.23"    "5.23"   "335"       "0.43"
## val_vec "0.7"     "1022.35" "5.67"    "6.66"   "344"       "0.52"
## val_vec "0.64"    "1022.64" "3.96"    "5.81"   "353"       "0"
## val_vec "0.61"    "1022.63" "3.99"    "5.04"   "340"       "0.48"
## val_vec "0.58"    "1022.2"  "2.39"    "4.49"   "351"       "0.41"
## val_vec "0.54"    "1022.94" "2.71"    "3.56"   "308"       "0.27"
## val_vec "0.53"    "1022.39" "2.07"    "3.35"   "123"       "0.4"
## val_vec "0.56"    "1022.91" "1.65"    "3.6"    "107"       "0.55"
## val_vec "0.61"    "1023.12" "2.47"    "2.62"   "100"       "0.46"
## val_vec "0.68"    "1023.24" "2.08"    "2.86"   "113"       "0.47"
## val_vec "0.73"    "1023.54" "2.08"    "2.75"   "132"       "0.39"
## val_vec "0.78"    "1023.9"  "1.52"    "2.72"   "168"       "0.09"
## val_vec "0.81"    "1023.5"  "1.49"    "2.14"   "154"       "0"
## val_vec "0.83"    "1023.6"  "2.03"    "2.43"   "123"       "0.49"
## val_vec "0.8"     "1023.22" "1.26"    "1.94"   "215"       "0.29"
## val_vec "0.83"    "1023.24" "1.52"    "2.37"   "104"       "0.88"
##         uvIndex visibility ozone
## val_vec "0"     "9.89"     "290.1"
## val_vec "0"     "9.942"    "287.6"
## val_vec "0"     "9.837"    "286.7"
## val_vec "0"     "9.819"    "285.9"
## val_vec "0"     "9.865"    "284.6"
## val_vec "0"     "9.693"    "282.8"
## val_vec "0"     "9.905"    "280.6"
## val_vec "0"     "9.874"    "281.2"
## val_vec "0"     "9.762"    "280.4"
## val_vec "1"     "9.874"    "279.9"
## val_vec "1"     "9.956"    "279.1"
## val_vec "2"     "9.98"     "277.9"
## val_vec "2"     "9.965"    "276.6"
## val_vec "1"     "9.98"     "274.9"
## val_vec "1"     "9.991"    "274"
## val_vec "0"     "9.959"    "273.2"
## val_vec "0"     "9.899"    "272.6"
## val_vec "0"     "9.686"    "271.9"
## val_vec "0"     "9.822"    "271.2"
## val_vec "0"     "9.781"    "274.2"
## val_vec "0"     "9.789"    "274.6"
## val_vec "0"     "9.832"    "274.6"
## val_vec "0"     "9.742"    "274.2"
## val_vec "0"     "9.832"    "272.8]"
```

As the weather data is separate on 12 locations and in 90 diffierent dates, our final crawler loop is designed as followed.

```r
location <- data.frame(place = c('Boston University',
                                 'Back Bay',
                                 'Becon Hill',
                                 'Fenway',
                                 'Financial District',
                                 'Haymarket Square',
                                 'North End',
                                 'North Station',
                                 'Northeastern University',
                                 'South Station',
                                 'Theatre District',
                                 'West End'),
                       coordinate = c('42.3523,-71.1214',
                                      '42.3518,-71.0805',
                                      '42.3593,-71.0682',
                                      '42.3495,-71.0992',
                                      '42.3524,-71.0562',
                                      '42.364,-71.0576',
                                      '42.3644,-71.0547',
                                      '42.3671,-71.0633',
                                      '42.3355,-71.0889',
                                      '42.3529,-71.0555',
                                      '42.3504,-71.0649',
                                      '42.3646,-71.0659'))
result_total <- NULL

for(area in location$place) {
  month <- 10
  day <- 1
  result <- NULL
  while(month < 13) {
    weather_url <- paste0('https://darksky.net/details/', location$coordinate[location$place == area], '/2018-'
, month, '-', day,'/us12/en')
    if(day < 32) {
      weather_raw <- weather_url %>% GET() %>% read_html(encoding = 'UTF-8')
      weather_info_raw <- weather_raw %>% html_nodes("script") %>% html_text()
      weather_info_vec <- weather_info_raw %>% strsplit(',\\{') %>% unlist
      for(i in seq(length(weather_info_vec))){
        weather_info_vec2 <- unlist(strsplit(weather_info_vec[i], ","))[1:17]
        weather_info_vec3 <- gsub('\"', "", weather_info_vec2)
        name_vec <- str_match(weather_info_vec2, '\\\"(.*?)\\\":')[,2]
        val_vec  <- gsub('\"|}', "",str_match(weather_info_vec2, ':(.*)')[,2])
        result <- rbind(result, val_vec)
        colnames(result) <- name_vec
      }
      day <- day + 1
    } else {
      month <- month + 1
      day <- 1
    }
  }
  result_df <- as.data.frame(result)
  result_df$location <- area
  result_total <- rbind(result_total, result_df)
}

# Save result
write.csv(result_total, 'web_weather.csv')
```

After the web crawler, we cleaned the data as follow.

```r
# load data
web_weather <- read.csv('web_weather.csv')
cab_rides2 <- read.csv('cab_rides_omit.csv')

# data type
str(web_weather)
```

```
## 'data.frame':    26796 obs. of  19 variables:
##  $ X                 : Factor w/ 26796 levels "val_vec","val_vec.1",..: 1 2 13334 17462 18794 20126 21458 2
2790 24122 25454 ...
##  $ time              : int  1538366400 1538370000 1538373600 1538377200 1538380800 1538384400 1538388000 15
38391600 1538395200 1538398800 ...
##  $ summary           : Factor w/ 16 levels "Clear","Drizzle",..: 10 8 8 9 9 8 8 9 8 8 ...
##  $ icon              : Factor w/ 9 levels "clear-day","clear-night",..: 6 6 6 3 3 6 6 3 5 5 ...
##  $ precipIntensity   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ precipProbability : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ precipType        : Factor w/ 5184 levels "12.24","12.54",..: 3696 3676 3742 3850 3756 3883 3851 3912 42
02 4343 ...
##  $ temperature       : num  54.8 54.6 55.2 56.3 55.4 ...
##  $ apparentTemperature: num  49.9 51.2 52.5 52.5 53.5 ...
##  $ dewPoint          : num  0.83 0.88 0.9 0.87 0.93 0.87 0.92 0.92 0.88 0.83 ...
##  $ humidity          : num  1026 1026 1026 1025 1025 ...
##  $ pressure          : num  1.26 0.62 1.52 1.57 1.79 2.02 2.17 1.76 2.93 3.29 ...
##  $ windSpeed         : num  2.35 1.99 3.19 3.53 3.26 3.2 3.25 3.28 5.02 5.88 ...
##  $ windGust          : num  201 209 223 217 190 221 210 221 222 226 ...
##  $ windBearing       : num  0.47 0.82 0.85 1 1 0.82 0.84 1 0.84 0.87 ...
##  $ cloudCover        : num  0 0 0 0 0 0 0 0 0 1 ...
##  $ uvIndex           : num  9.92 9.78 9.76 9.76 9.8 ...
##  $ visibility        : Factor w/ 4694 levels "0","0.199","0.221",..: 1017 1012 978 976 980 983 982 981 978
973 ...
##  $ location          : Factor w/ 12 levels "Back Bay","Becon Hill",..: 3 3 3 3 3 3 3 3 3 3 ...
```

```
web_weather$summary <- as.character(web_weather$summary)
web_weather$icon <- as.character(web_weather$icon)
web_weather$location <- as.character(web_weather$location)

# data cleaning
## web weather
any(is.na(web_weather))
```

```
## [1] FALSE
```

```
web_weather$X <- NULL
web_weather$visibility <- NULL
web_weather$cab_hour <- as.POSIXct(web_weather$time, origin = '1970-01-01') %>% round_date(unit = 'hour')
names(web_weather)[17] <- 'source'

# cab rides
cab_rides2$X <- NULL
cab_rides2$time <- as.POSIXct(cab_rides2$time)
cab_rides2$cab_hour <- round_date(cab_rides2$time, unit = 'hour')

# merge
cab_web.weather <- merge(cab_rides2, web_weather, by = c('cab_hour', 'source'))
write.csv(cab_web.weather, 'cab_web.weather.csv')
```

By the aboved codes, we could get the dataset merged together with correct data structures.

Modelling

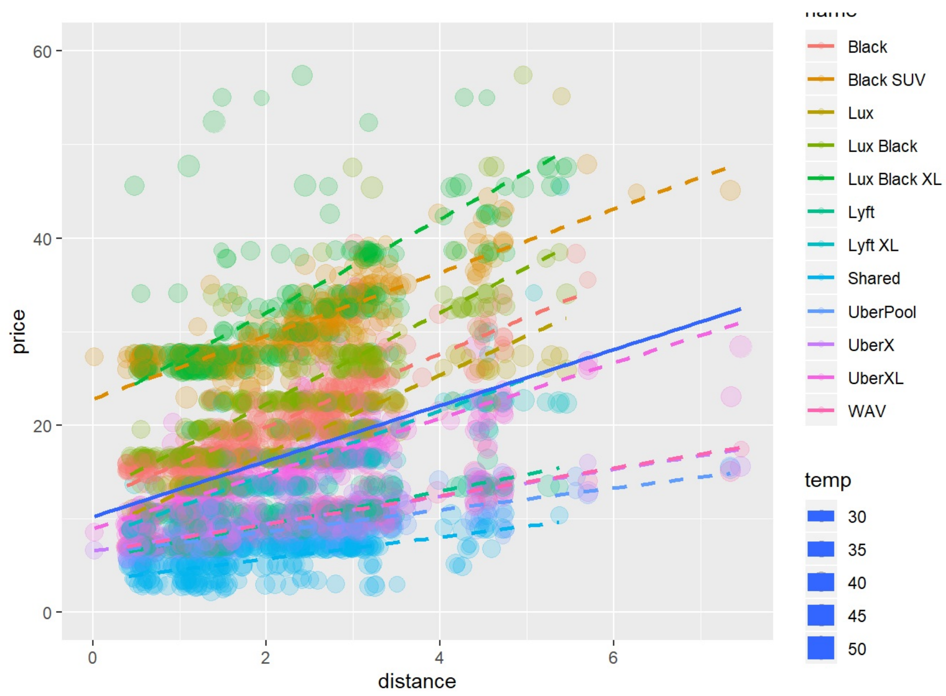
In order to predict the cab price, we tried to set a linear regression model of `price` and some other factors. By using the dataset given by kaggle, we can set the model as followed.

```
# Load Data
cab_weather_n <- read.csv('cab_weather_new.csv')

# Linear Regression
lm.sol <- lm(price ~ distance + temp + clouds + pressure + rain + humidity + wind, data = cab_weather_n)
summary(lm.sol)
```

```
##
## Call:
## lm(formula = price ~ distance + temp + clouds + pressure + rain +
##     humidity + wind, data = cab_weather_n)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.022  -6.998  -1.774   5.004  66.009
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 23.43425   29.42651   0.796    0.426
## distance     2.97367    0.13296  22.366   <2e-16 ***
## temp         0.05401    0.07204   0.750    0.453
## clouds       0.75628    0.78098   0.968    0.333
## pressure    -0.01365    0.03113  -0.439    0.661
## rain        -0.53281    2.00944  -0.265    0.791
## humidity    -1.92142    2.56040  -0.750    0.453
## wind        -0.09229    0.08448  -1.092    0.275
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.961 on 3540 degrees of freedom
## Multiple R-squared:  0.1256, Adjusted R-squared:  0.1239
## F-statistic: 72.66 on 7 and 3540 DF,  p-value: < 2.2e-16
```

```
# plot
ggplot(cab_weather_n, aes(distance, price, col = name, size = temp)) +
  geom_jitter(alpha = 0.2) +
  geom_smooth(method = 'lm', se = F, linetype = 2) +
  geom_smooth(method = 'lm', se = F, aes(group = 1)) +
  coord_cartesian(ylim = c(0, 60))
```



As modelling aboved, we can find that only `distance` factor is significant at confidence level of 5%. We can generate a plot show how factors influenced `price` variable.

Better Model

Since we get better weather dataset by using web crawler. We can set a better model as followed.

```
# Better Model
cab_web.weather <- read.csv('cab_web.weather.csv')
lm.sol.web <- lm(price ~ distance + apparentTemperature + precipIntensity + humidity + pressure + windSpeed, da
ta = cab_web.weather)
summary(lm.sol.web)
```

```
##
## Call:
## lm(formula = price ~ distance + apparentTemperature + precipIntensity +
##     humidity + pressure + windSpeed, data = cab_web.weather)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -22.686  -6.960  -1.718   4.867  74.544
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         9.8392753  1.2135878   8.108 5.17e-16 ***
## distance            2.8391998  0.0095002 298.857  < 2e-16 ***
## apparentTemperature 0.0016222  0.0016160   1.004    0.315
## precipIntensity     0.8354742  0.5848311   1.429    0.153
## humidity            0.0004764  0.0011614   0.410    0.682
## pressure            0.0004075  0.0011679   0.349    0.727
## windSpeed           0.0005182  0.0011627   0.446    0.656
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.82 on 623014 degrees of freedom
## Multiple R-squared:  0.1254, Adjusted R-squared:  0.1254
## F-statistic: 1.489e+04 on 6 and 623014 DF,  p-value: < 2.2e-16
```

As shown aboved, the `distance` factor is also the only significant variable at 5% confidence level. We are curious about with out this main influence, how would other factors influence the cab price.

```
lm.sol.weather <- lm(price ~ apparentTemperature + precipIntensity + humidity + pressure + windSpeed, data = ca
b_web.weather)
summary(lm.sol.weather)
```

```
##
## Call:
## lm(formula = price ~ apparentTemperature + precipIntensity +
##     humidity + pressure + windSpeed, data = cab_web.weather)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.299  -7.605  -3.082   5.895  80.879
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         18.510321   1.297293  14.268   <2e-16 ***
## apparentTemperature -0.002398   0.001728  -1.388    0.165
## precipIntensity      0.863254   0.625348   1.380    0.167
## humidity            -0.001767   0.001242  -1.423    0.155
## pressure            -0.001821   0.001249  -1.458    0.145
## windSpeed           -0.001596   0.001243  -1.283    0.199
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.431 on 623015 degrees of freedom
## Multiple R-squared:  9.824e-06,  Adjusted R-squared:  1.799e-06
## F-statistic: 1.224 on 5 and 623015 DF,  p-value: 0.2946
```

Surprisingly, after removing the main influencing factor, more variables begin to significantly affect the dependent variable. `apparentTemperature`, `humidity`, and `pressure` are significant under 10% confidence level. Seems this 3 factors could slightly influence the cab price.