# Predicting Ride Share Prices

BA810 - Team 8A

03/02/2021

```
library(data.table)
library(ggplot2)
library(ggthemes)
library(scales)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──
```

```
## ✓ tibble  3.0.6     ✓ purrr   0.3.4
## ✓ tidyr   1.1.2     ✓ stringr 1.4.0
## ✓ readr   1.4.0     ✓ forcats 0.5.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::between()    masks data.table::between()
## x readr::col_factor() masks scales::col_factor()
## x purrr::discard()    masks scales::discard()
## x dplyr::filter()     masks stats::filter()
## x dplyr::first()      masks data.table::first()
## x dplyr::lag()        masks stats::lag()
## x dplyr::last()       masks data.table::last()
## x purrr::transpose()  masks data.table::transpose()
```

```r
library(janitor)
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```r
library(skimr)
library(RSocrata)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library(tidytext)
library(fastDummies)
library(lubridate)
library(ISLR)
library(quantmod)
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'xts'
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
## The following objects are masked from 'package:data.table':
##
##     first, last
```

```
## Loading required package: TTR
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(tibble)

library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
## Loaded glmnet 4.1
```

```
library(Metrics)
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
theme_set(theme_minimal())
```

```
cab <- fread('~/Google Drive/Shared drives/BA810 - Team 8A/data/clean_cab_weather.csv')
head(cab)
```

```
##                                      id distance company      destination price
## 1: ef4771c2-c88d-4730-aaf7-a95751e9d27e     3.03    Lyft Theatre District  34.0
## 2: 00ea74ea-2c49-416c-bfc5-f7877025f6eb     1.30    Uber Theatre District  18.5
## 3: 8682f9bf-5cc0-4dfc-b8fe-4e22070d1684     2.71    Uber           Fenway  19.5
## 4: edfc7f44-97e1-48cd-930c-e4fe20e88ac8     2.43    Lyft      Beacon Hill  10.5
## 5: 6172077a-22de-481b-aae2-b5763c87a6c4     2.71    Uber           Fenway  32.0
## 6: bb3f969d-3190-4bb8-9a84-dff2deba0a98     2.19    Uber        North End   8.0
##    surge_multiplier         type                datetime  temp clouds pressure
## 1:                1 Lux Black XL 2018-11-26 03:40:46.318 41.07   0.86  1014.39
## 2:                1        Black 2018-11-26 03:40:46.319 40.86   0.87  1014.39
## 3:                1        UberX 2018-11-26 03:40:46.320 40.80   0.87  1014.39
## 4:                1         Lyft 2018-11-26 03:40:46.320 40.81   0.89  1014.35
## 5:                1       UberXL 2018-11-26 03:40:46.320 40.80   0.87  1014.39
## 6:                1        UberX 2018-11-26 03:40:46.358 41.02   0.87  1014.39
##    rain humidity wind                  origin    day hour    time_stamp luxury
## 1:    0     0.92 1.36       Boston University Monday    3 1543203646318      1
## 2:    0     0.93 1.60           South Station Monday    3 1543203646319      1
## 3:    0     0.93 1.55        Theatre District Monday    3 1543203646320      0
## 4:    0     0.93 1.36 Northeastern University Monday    3 1543203646320      0
## 5:    0     0.93 1.55        Theatre District Monday    3 1543203646320      0
## 6:    0     0.92 1.50             Beacon Hill Monday    3 1543203646358      0
##       size
## 1:   large
## 2: regular
## 3: regular
## 4: regular
## 5:   large
## 6: regular
```

# Exploratory Data Analysis

```
str(cab)
```

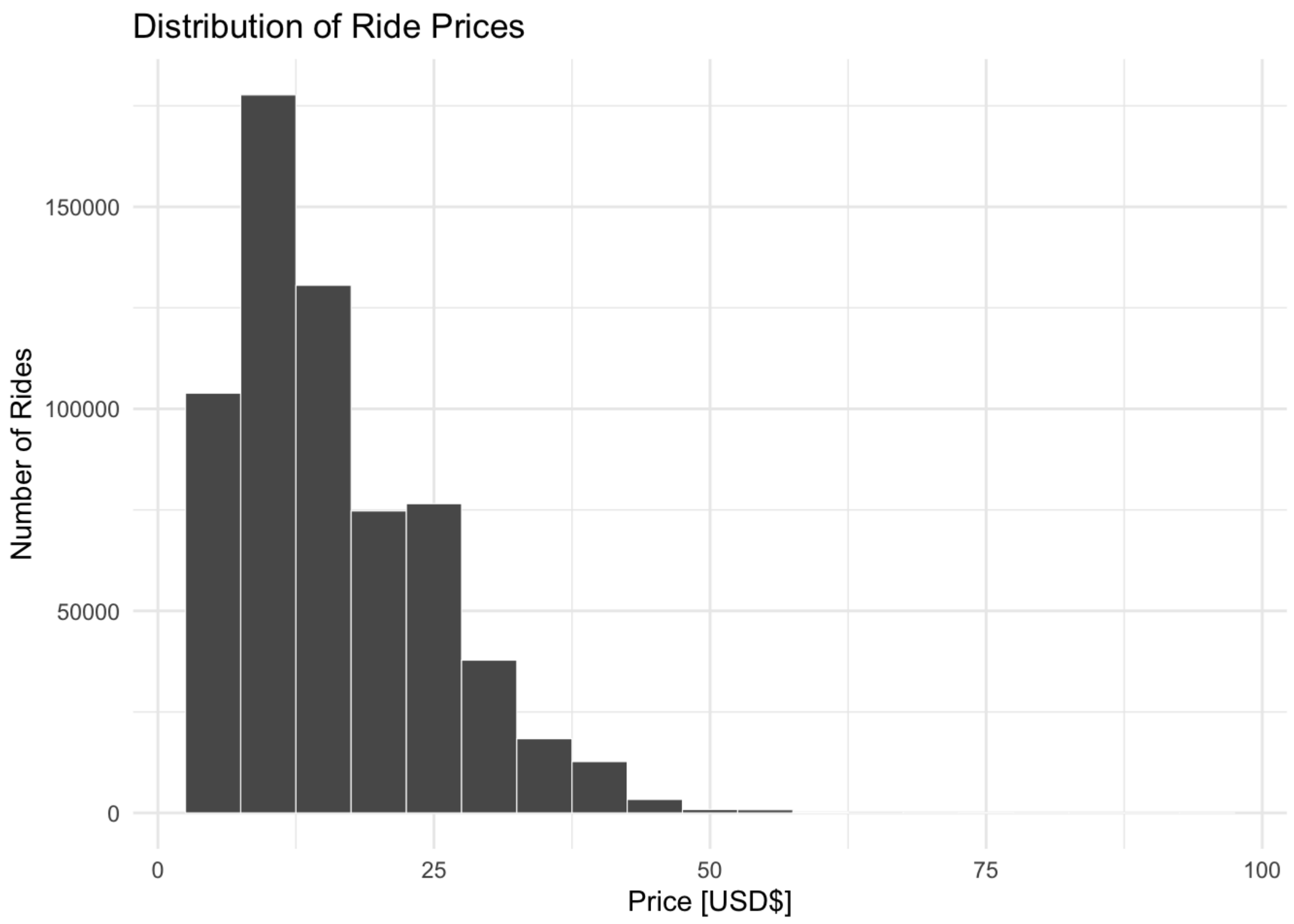
```
## Classes 'data.table' and 'data.frame':   637905 obs. of  20 variables:
##  $ id              : chr  "ef4771c2-c88d-4730-aaf7-a95751e9d27e" "00ea74ea-2c49-416c-bfc5-f78
77025f6eb" "8682f9bf-5cc0-4dfc-b8fe-4e22070d1684" "edfc7f44-97e1-48cd-930c-e4fe20e88ac8" ...
##  $ distance        : num  3.03 1.3 2.71 2.43 2.71 2.19 3.05 2.19 2.19 2.22 ...
##  $ company         : chr  "Lyft" "Uber" "Uber" "Lyft" ...
##  $ destination     : chr  "Theatre District" "Theatre District" "Fenway" "Beacon Hill" ...
##  $ price           : num  34 18.5 19.5 10.5 32 8 10.5 13 17.5 7 ...
##  $ surge_multiplier: num  1 1 1 1 1 1 1 1 1 1 ...
##  $ type            : chr  "Lux Black XL" "Black" "UberX" "Lyft" ...
##  $ datetime        : chr  "2018-11-26 03:40:46.318" "2018-11-26 03:40:46.319" "2018-11-26 03:
40:46.320" "2018-11-26 03:40:46.320" ...
##  $ temp            : num  41.1 40.9 40.8 40.8 40.8 ...
##  $ clouds          : num  0.86 0.87 0.87 0.89 0.87 0.87 0.89 0.87 0.87 0.87 ...
##  $ pressure        : num  1014 1014 1014 1014 1014 ...
##  $ rain            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ humidity        : num  0.92 0.93 0.93 0.93 0.93 0.92 0.92 0.92 0.92 0.93 ...
##  $ wind            : num  1.36 1.6 1.55 1.36 1.55 1.5 1.43 1.5 1.5 1.55 ...
##  $ origin          : chr  "Boston University" "South Station" "Theatre District" "Northeaster
n University" ...
##  $ day             : chr  "Monday" "Monday" "Monday" "Monday" ...
##  $ hour            : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ time_stamp      :integer64 1543203646318 1543203646319 1543203646320 1543203646320 1543203
646320 1543203646358 1543203646358 1543203646358 ...
##  $ luxury          : int  1 1 0 0 0 0 0 0 1 0 ...
##  $ size            : chr  "large" "regular" "regular" "regular" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

# Price

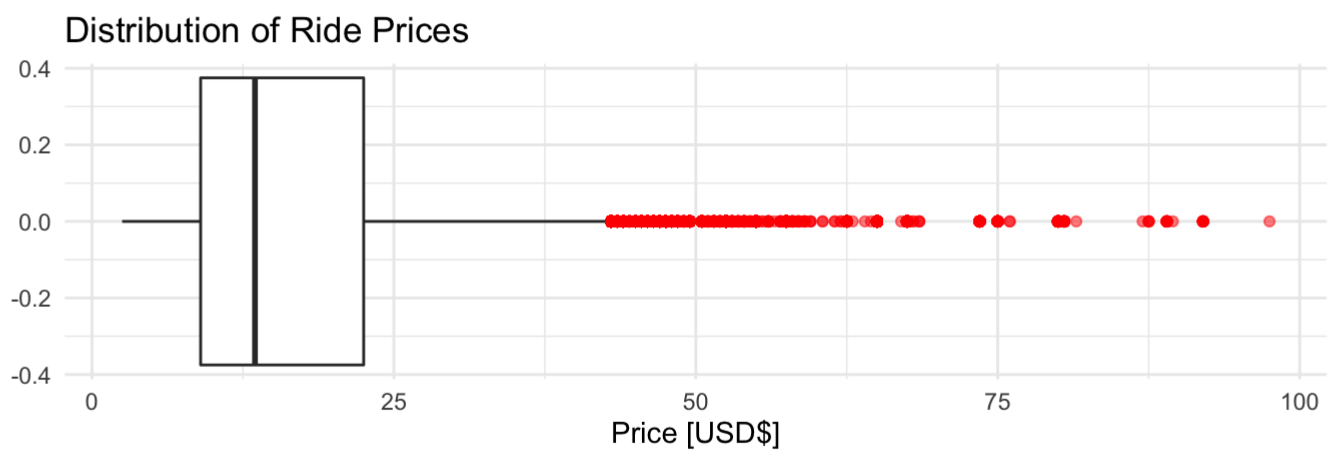
```
summary(cab$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.50    9.00   13.50   16.55   22.50   97.50
```

```
ggplot(cab, aes(x=cab$price)) +
  geom_histogram(binwidth=5,
                 color='white',
                 size=0.2) +
  labs(title = 'Distribution of Ride Prices',
       x = 'Price [USD$]',
       y = 'Number of Rides')
```



```
ggplot(cab, aes(x=cab$price)) +
  geom_boxplot(outlier.color = 'red',
               outlier.alpha = 0.5) +
  labs(title = 'Distribution of Ride Prices',
       x = 'Price [USD$]') +
  theme(aspect.ratio=5/20)
```



Based on the summary statistics and the histogram distribution, it seems that the ride share prices are normally distributed with a right skew. It’s interesting to see that there are high-priced outliers. Let’s investigate what these data points look like:

```
nrow(cab[price>50])
```

```
## [1] 2033
```

```
sample_n(cab[price>50], 10)
```

```
##                                       id distance company        destination
##  1: b93b539e-e02c-426c-a505-b4be1027dda4     3.34    Lyft           West End
##  2: 6e1e9a3a-308f-428a-9c36-ed34ab680ee8     2.08    Lyft           West End
##  3: 28716da3-cba6-4cc3-8b14-02c47f5bccdf     3.22    Lyft           West End
##  4: a7b22aad-245c-4c48-81ac-52f35577acb6     3.09    Lyft             Fenway
##  5: ec73c4d4-6f1b-4cad-8f09-3ba85a1d465d     4.71    Lyft  Boston University
##  6: 20bb5893-ef4f-490c-9ba8-e1e5838957f1     2.46    Lyft      South Station
##  7: 01087b6b-1776-40fc-ac3c-d43f73ab6c46     7.25    Uber Financial District
##  8: 91cbd7ac-287b-493e-81bd-a8a096a62187     4.25    Lyft Financial District
##  9: 3e51bc78-c428-4026-abb0-b08c68de05a7     1.49    Lyft             Fenway
## 10: 2a348b00-6987-4b2b-a18c-5cf2f283f7b5     4.80    Lyft  Boston University
##     price surge_multiplier         type                datetime  temp clouds
##  1:  55.0             1.50 Lux Black XL 2018-11-27 08:00:21.982 43.47   1.00
##  2:  57.5             2.00 Lux Black XL 2018-11-27 07:00:21.863 43.74   1.00
##  3:  55.0             2.00    Lux Black 2018-12-18 14:40:10.275 26.27   0.48
##  4:  57.5             1.50 Lux Black XL 2018-11-29 16:33:06.595 44.12   0.57
##  5:  65.0             2.00    Lux Black 2018-12-03 22:23:02.450 48.01   0.18
##  6:  55.0             2.00 Lux Black XL 2018-12-18 17:40:11.530 29.94   0.15
##  7:  57.0             1.00    Black SUV 2018-12-16 09:50:05.037 38.36   0.28
##  8:  52.5             2.00          Lux 2018-12-18 12:45:05.351 23.84   0.50
##  9:  55.0             2.00 Lux Black XL 2018-12-03 19:37:56.002 51.97   0.67
## 10:  57.5             1.75    Lux Black 2018-12-16 05:35:09.833 40.04   0.41
##     pressure   rain humidity  wind                  origin      day hour
##  1:   995.03 0.0117     0.92  9.90       Boston University  Tuesday    8
##  2:   996.36 0.0103     0.90 10.92           South Station  Tuesday    7
##  3:  1010.12 0.0000     0.47 13.64 Northeastern University  Tuesday   14
##  4:  1006.85 0.0000     0.53 11.60        Theatre District Thursday   16
##  5:  1001.45 0.0000     0.55 10.91        Theatre District   Monday   22
##  6:  1011.43 0.0000     0.46 13.52             Beacon Hill  Tuesday   17
##  7:  1022.37 0.0000     0.76  7.57       Boston University   Sunday    9
##  8:  1008.83 0.0000     0.51 14.74                  Fenway  Tuesday   12
##  9:   999.97 0.0000     0.52  8.02                Back Bay   Monday   19
## 10:  1023.48 0.0000     0.71  7.86        Theatre District   Sunday    5
##        time_stamp luxury    size
##  1: 1543305621982      1   large
##  2: 1543302021863      1   large
##  3: 1545144010275      1 regular
##  4: 1543509186595      1   large
##  5: 1543875782450      1 regular
##  6: 1545154811530      1   large
##  7: 1544953805037      1   large
##  8: 1545137105351      1 regular
##  9: 1543865876002      1   large
## 10: 1544938509833      1 regular
```

```
cab[price>50, .(car_type = unique(type))]
```

```
##        car_type
## 1:    Lux Black
## 2: Lux Black XL
## 3:          Lux
## 4:       UberXL
## 5:    Black SUV
## 6:        Black
## 7:      Lyft XL
```

It seems that there are 2000+ rows that have prices of more than $50. Additionally, based on a random sample of 10 rows, all these 'expensive rides are made up of the higher end ride types (e.g. Uber Black SUV, Lyft Lux Black XL, etc.). Therefore, it's likely that the type of ride (lux vs normal vs share) might be an important factor. It might be worth doing some feature engineering to capture this categorical variable, especially since the names are not the same across Uber and Lyft rides.

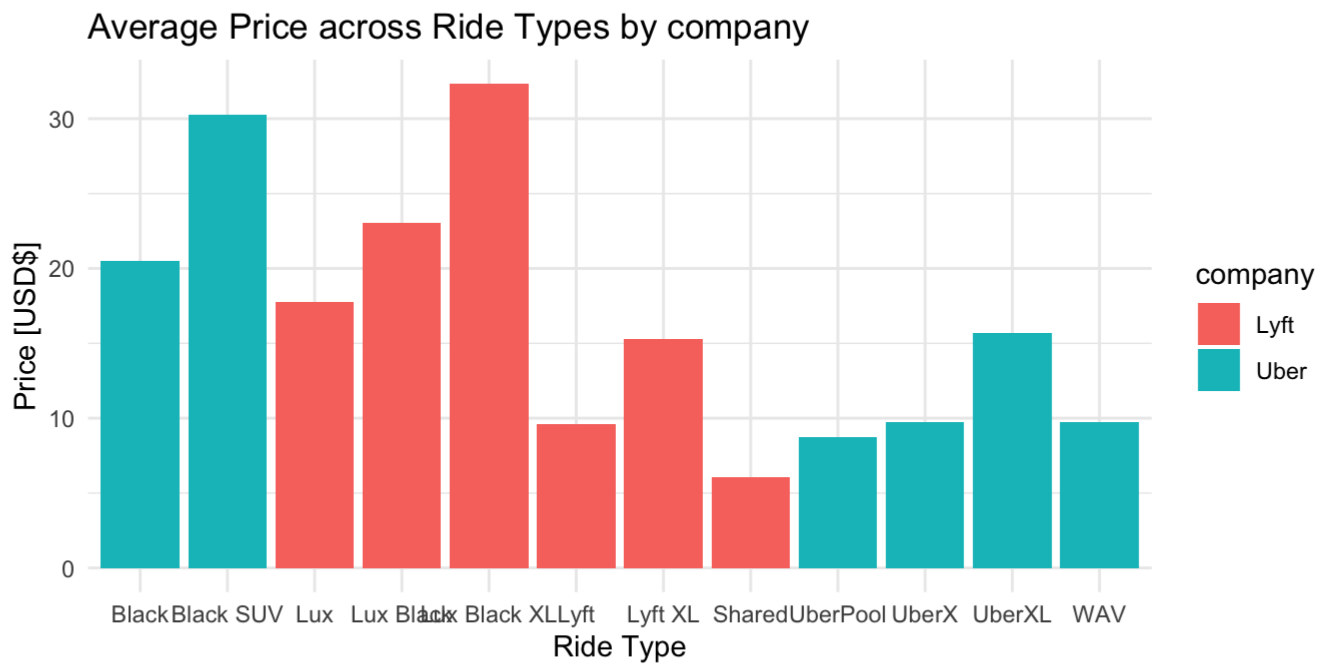
## Cab Types and Size

As previously seen during the EDA of the price, it seems that there's significance in the type and size of the cars in determining price (as seen by the fact that rides with price > $50 is made up of the luxury type of vehicles).

```
tmp <- cab[order(type), .(company = unique(company), count = .N, median_price = median(price), a
vg_price = mean(price)), by = .(car_type = type)]

tmp[order(-avg_price)]
```

```
##         car_type company count median_price avg_price
##  1: Lux Black XL    Lyft 51231         30.0 32.324102
##  2:    Black SUV    Uber 55088         28.5 30.286859
##  3:    Lux Black    Lyft 51231         22.5 23.062619
##  4:        Black    Uber 55085         19.5 20.524045
##  5:          Lux    Lyft 51232         16.5 17.771256
##  6:       UberXL    Uber 55086         15.0 15.678167
##  7:      Lyft XL    Lyft 51231         13.5 15.309475
##  8:          WAV    Uber 55086          9.5  9.765176
##  9:       UberX    Uber 55088          9.5  9.765103
## 10:         Lyft    Lyft 51229          9.0  9.611073
## 11:     UberPool    Uber 55085          8.5  8.752673
## 12:       Shared    Lyft 51233          7.0  6.029893
```

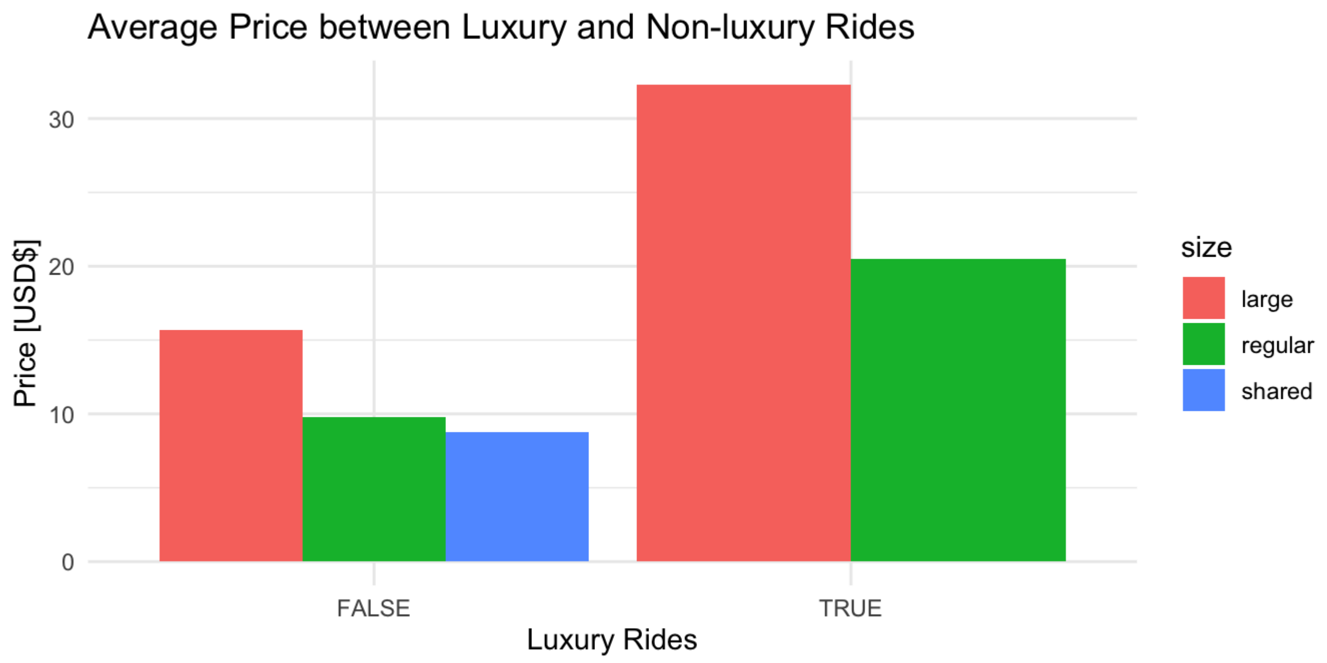
```
ggplot(tmp,aes(car_type, avg_price, fill=company)) +
  geom_bar(stat='identity') +
  labs(title = 'Average Price across Ride Types by company',
       y = 'Price [USD$]',
       x = 'Ride Type') +
  theme(aspect.ratio = 1/2)
```



Considering that fact that are significant price variations (as shown in the figure above), we will try to make a categorical variable that signifies whether the size is shared, normal, or XL and whether or not the ride is luxury or not.

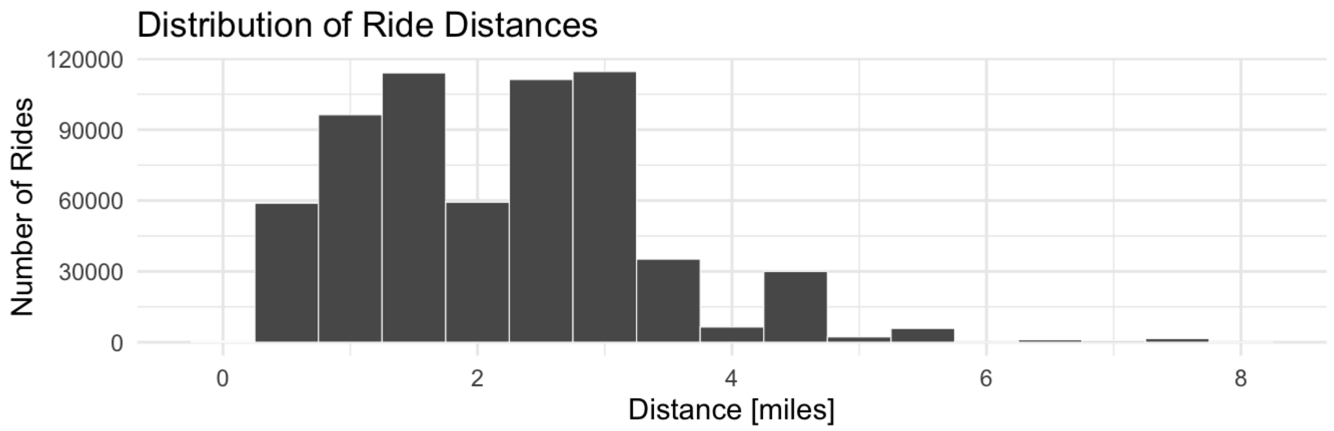
This is what the average prices look like:

```r
tmp <- cab[,.(
  count = .N,
  median_price = median(price),
  avg_price = mean(price)),

  by=.(company,size,luxury)][order(-avg_price)]

tmp[, luxury := luxury == 1]

ggplot(tmp, aes(luxury, avg_price, fill=size)) +
  geom_col(position = position_dodge()) +
  labs(title = 'Average Price between Luxury and Non-luxury Rides',
       y = 'Price [USD$]',
       x = 'Luxury Rides') +
  theme(aspect.ratio=1/2)
```

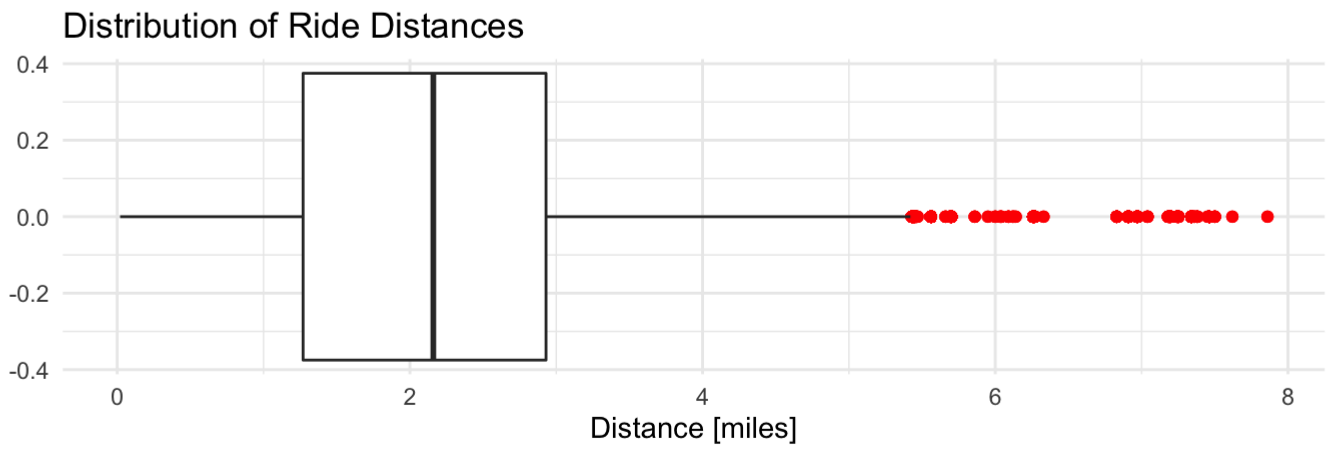


## Distance

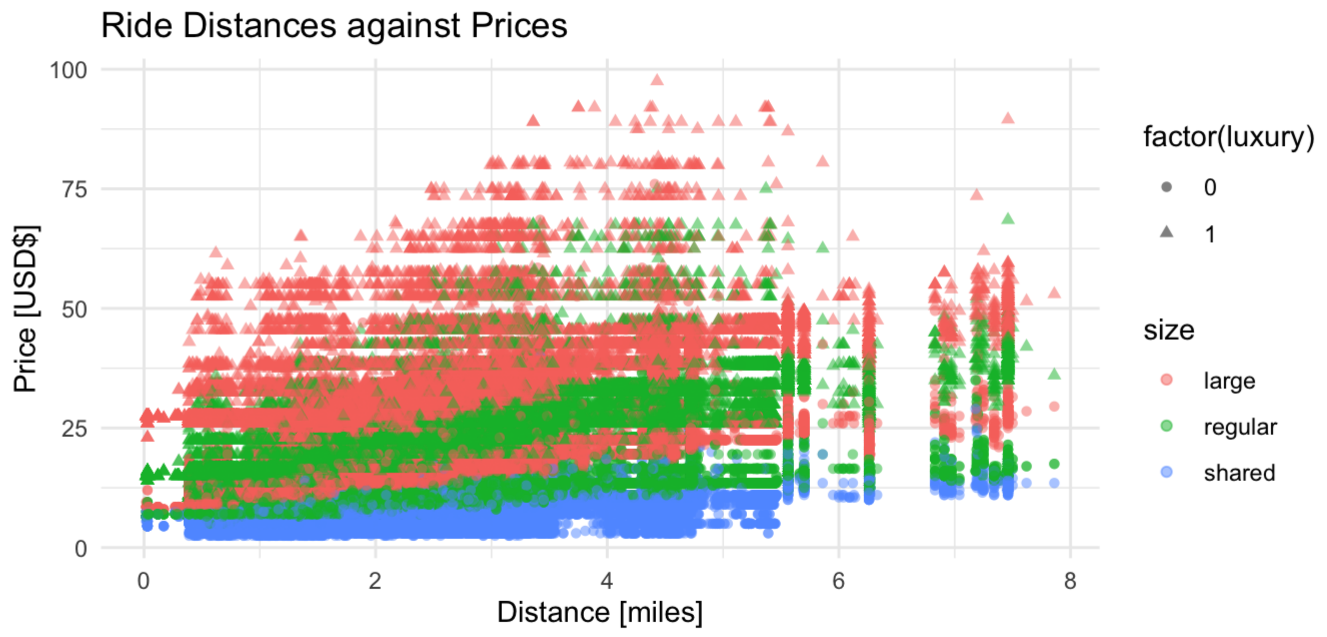
```
ggplot(cab, aes(distance)) +
  geom_histogram(binwidth=0.5,
                 color='white',
                 size=0.2) +
  labs(title = 'Distribution of Ride Distances',
       x = 'Distance [miles]',
       y = 'Number of Rides') +
  theme(aspect.ratio = 1/4)
```



```
ggplot(cab, aes(distance)) +
  geom_boxplot(outlier.color = 'red',
               outlier.alpha = 0.5) +
  labs(title = 'Distribution of Ride Distances',
       x = 'Distance [miles]') +
  theme(aspect.ratio=5/20)
```



```
ggplot(cab, aes(distance, price)) +
  geom_point(alpha=0.5, aes(
    color = size,
    shape = factor(luxury)
    )) +
  labs(title="Ride Distances against Prices",
       x = "Distance [miles]",
       y = "Price [USD$]") +
  theme(aspect.ratio = 1/2)
```


Ride Distances against Prices
Price [USD$]
100
75
50
25
0
0
2
4
6
8
Distance [miles]
factor(luxury)
0
1
size
large
regular
shared

```
str(cab)
```

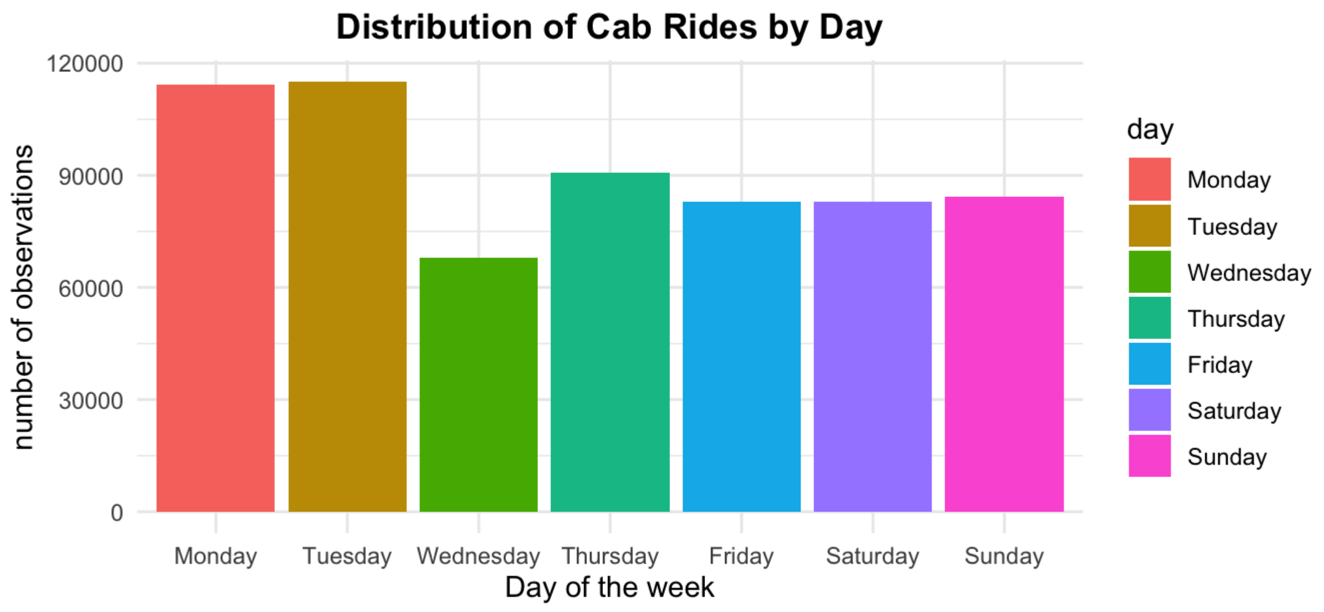
```
## Classes 'data.table' and 'data.frame':   637905 obs. of  20 variables:
##  $ id              : chr  "ef4771c2-c88d-4730-aaf7-a95751e9d27e" "00ea74ea-2c49-416c-bfc5-f78
77025f6eb" "8682f9bf-5cc0-4dfc-b8fe-4e22070d1684" "edfc7f44-97e1-48cd-930c-e4fe20e88ac8" ...
##  $ distance        : num  3.03 1.3 2.71 2.43 2.71 2.19 3.05 2.19 2.19 2.22 ...
##  $ company         : chr  "Lyft" "Uber" "Uber" "Lyft" ...
##  $ destination     : chr  "Theatre District" "Theatre District" "Fenway" "Beacon Hill" ...
##  $ price           : num  34 18.5 19.5 10.5 32 8 10.5 13 17.5 7 ...
##  $ surge_multiplier: num  1 1 1 1 1 1 1 1 1 1 ...
##  $ type            : chr  "Lux Black XL" "Black" "UberX" "Lyft" ...
##  $ datetime        : chr  "2018-11-26 03:40:46.318" "2018-11-26 03:40:46.319" "2018-11-26 03:
40:46.320" "2018-11-26 03:40:46.320" ...
##  $ temp            : num  41.1 40.9 40.8 40.8 40.8 ...
##  $ clouds          : num  0.86 0.87 0.87 0.89 0.87 0.87 0.89 0.87 0.87 0.87 ...
##  $ pressure        : num  1014 1014 1014 1014 1014 ...
##  $ rain            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ humidity        : num  0.92 0.93 0.93 0.93 0.93 0.92 0.92 0.92 0.92 0.93 ...
##  $ wind            : num  1.36 1.6 1.55 1.36 1.55 1.5 1.43 1.5 1.5 1.55 ...
##  $ origin          : chr  "Boston University" "South Station" "Theatre District" "Northeaster
n University" ...
##  $ day             : chr  "Monday" "Monday" "Monday" "Monday" ...
##  $ hour            : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ time_stamp      :integer64 1543203646318 1543203646319 1543203646320 1543203646320 1543203
646320 1543203646358 1543203646358 1543203646358 ...
##  $ luxury          : int  1 1 0 0 0 0 0 0 1 0 ...
##  $ size            : chr  "large" "regular" "regular" "regular" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

## Time and Day

```
cab[,.N, by='day'][order(N,decreasing = TRUE)]
```

```
##          day      N
## 1:   Tuesday 115081
## 2:    Monday 114228
## 3:  Thursday  90707
## 4:    Sunday  84175
## 5:  Saturday  83004
## 6:    Friday  82874
## 7: Wednesday  67836
```

```
cab$day <- factor(cab$day, levels= c( "Monday",
    "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday","Sunday"))

palette <- c(automatic= '#377EB8',manual= 'E41A1C')

ggplot(cab,aes(day,fill=day)) +
  geom_bar() + theme_minimal()+theme(plot.title = element_text(hjust = 0.5, lineheight = 0.8, fa
ce = "bold"))+
      labs(
       x="Day of the week",
       y="number of observations",
       title="Distribution of Cab Rides by Day")+
       theme(aspect.ratio = 1/2)
```

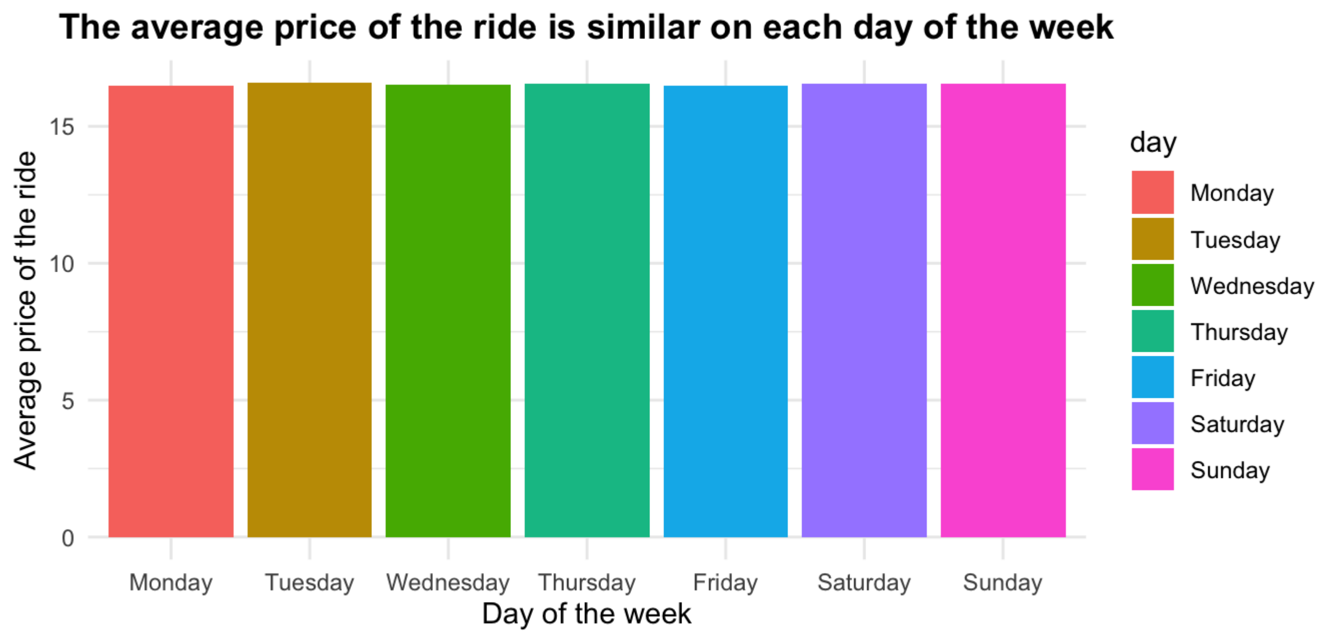


Since this is a simulated data set, the distribution of rides according to the day of the week doesn't really matter.

```
# What is the average price of the ride in each day of the week?

#summary statistics
cab[,average_price:= mean(price),by='day']  #[order(average_price,decreasing = TRUE)]
cab[,.(average_price= mean(price)),by='day']
```

```
##          day average_price
## 1:    Monday      16.49297
## 2:   Tuesday      16.59053
## 3: Wednesday      16.52642
## 4:  Thursday      16.57413
## 5:    Friday      16.48938
## 6:  Saturday      16.56665
## 7:    Sunday      16.57170
```

```
# bar plot
ggplot(cab,aes(x= day,y= price,fill=day)) +
  geom_bar(stat='summary', fun='mean') + theme_minimal()+theme(plot.title = element_text(hjust =
0.5, lineheight = 0.8, face = "bold"))+
      labs(
       x="Day of the week",
       y="Average price of the ride",
       title="The average price of the ride is similar on each day of the week")+
       theme(aspect.ratio = 1/2)
```

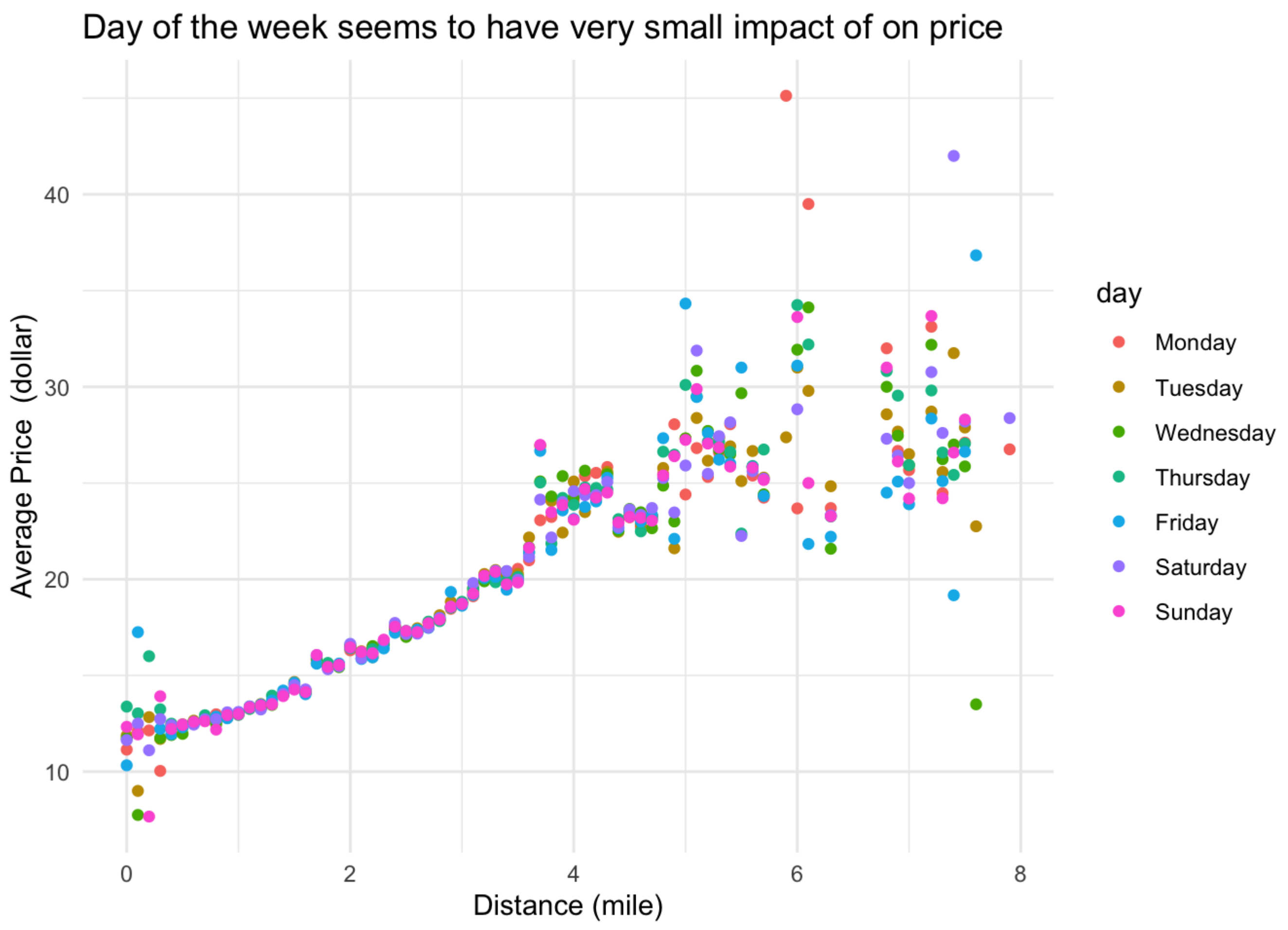


Based on the averages, it seems that there is no difference in price on any given day. There seems to be no correlation between ride prices against the day of the week.

```
# Create a scatterplot to further examine the relationship among price,distance, and the day of
 the week

dist_p <- cab  %>%
  mutate(dist = round(cab$distance,digit = 1))  %>%
  select(price,dist,day) %>%
  group_by(dist,day)  %>%
  summarise(avg_p = mean(price))
```

```
## `summarise()` has grouped output by 'dist'. You can override using the `.groups` argument.
```

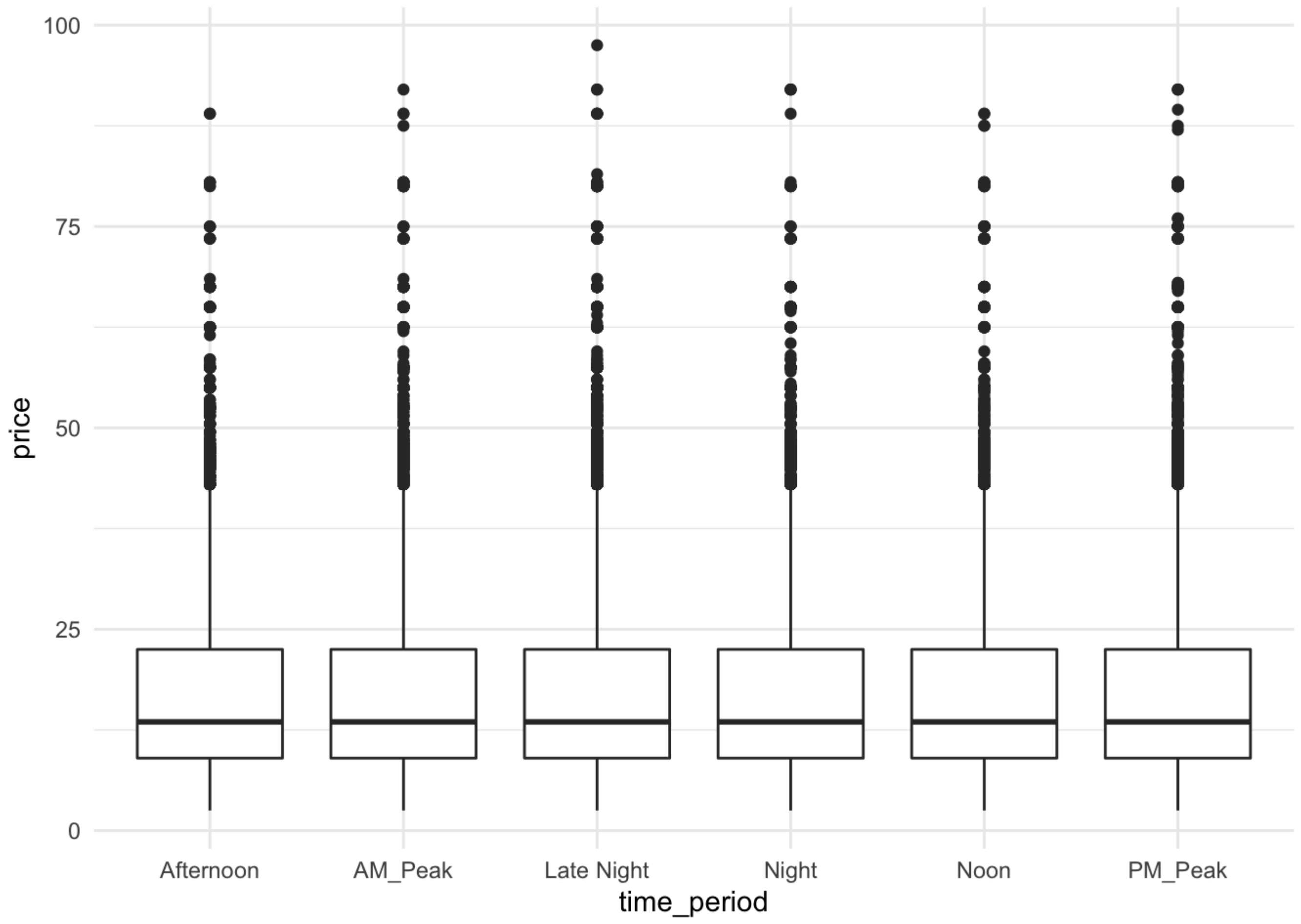
```
ggplot(dist_p, aes(x=dist,y=avg_p,color=day))+
  geom_point() +
  labs(
       x="Distance (mile)",
       y="Average Price  (dollar)",
       title='Day of the week seems to have very small impact of on price' )
```



##

Time of Day

```
# create labels to differentiate time intervals

ride_hour <- cab %>%
  rownames_to_column() %>%
  mutate(time_period = case_when(( hour < 6 & hour >= 0)~ "Late Night", ( hour >= 6 & hour <= 10
) ~"AM_Peak", (hour > 10 & hour <= 13 )~"Noon", (hour > 13 & hour < 17)~"Afternoon",(hour >= 17
 & hour <= 20)~ "PM_Peak", (hour > 20 & hour <=23 )~"Night"))

#  boxplot visualizaiton
ggplot(ride_hour, aes(x=time_period, y=price)) +
  geom_boxplot()
```



```
ride_hour[,.(median = median(price),
             average = mean(price),
             max = max(price)),
             by='time_period']
```

```
##    time_period median  average  max
## 1:  Late Night   13.5 16.55214 97.5
## 2:     AM_Peak   13.5 16.52410 92.0
## 3:        Noon   13.5 16.53113 89.0
## 4:   Afternoon   13.5 16.52943 89.0
## 5:     PM_Peak   13.5 16.56859 92.0
## 6:       Night   13.5 16.56417 92.0
```

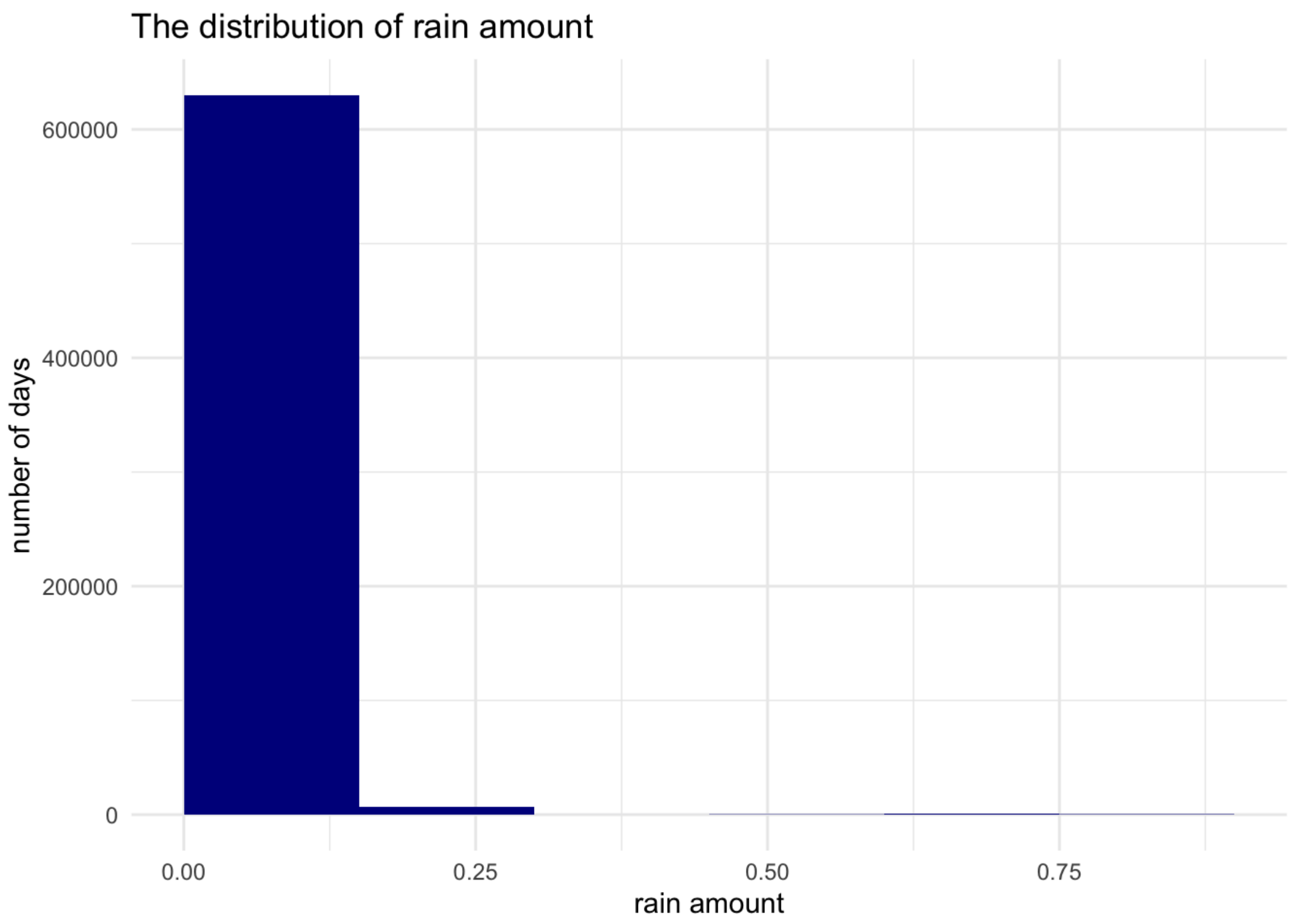
No variance in price among time of day.

## Rain

```
summary(cab$rain)
```

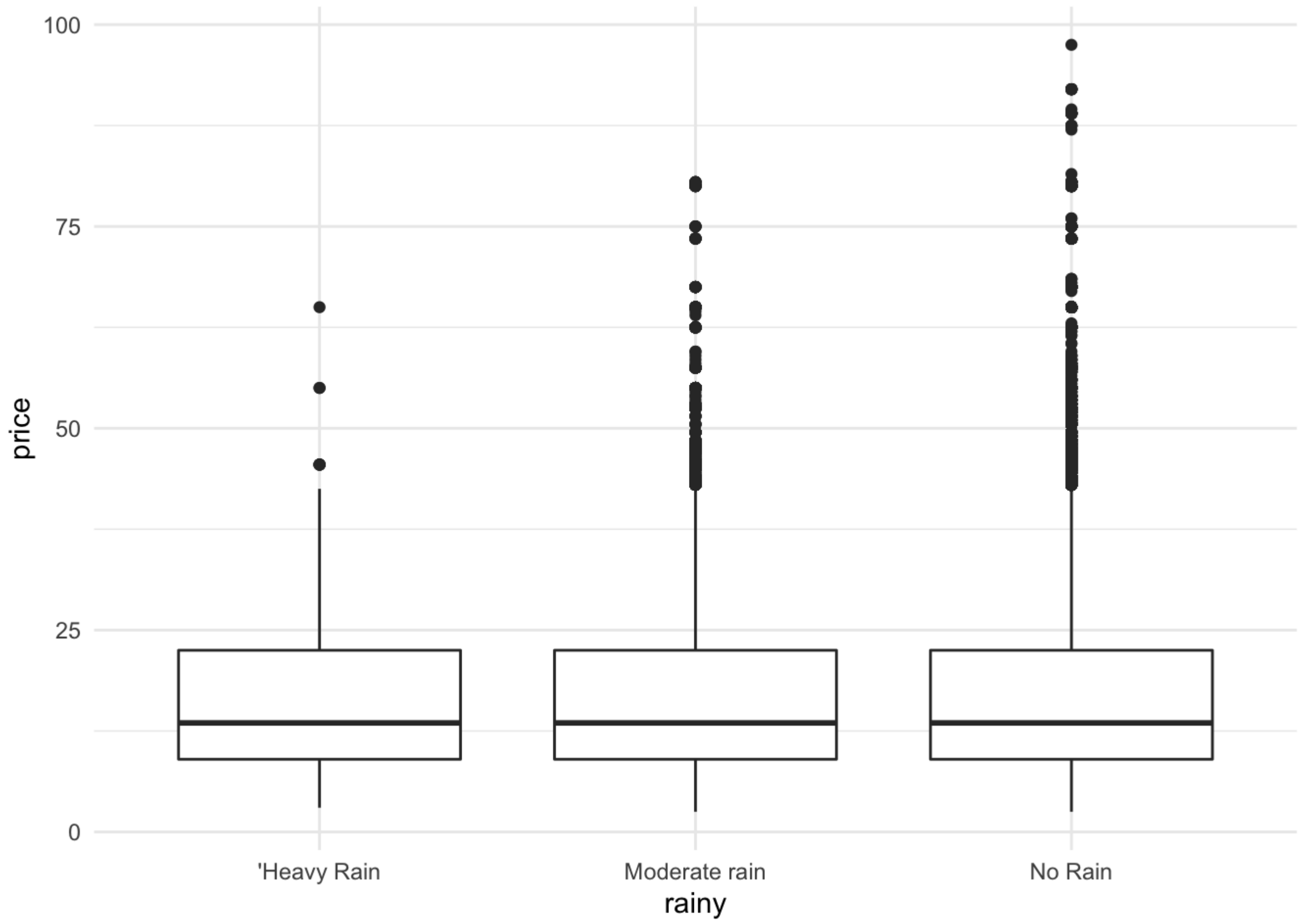
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
## 0.000000 0.000000 0.000000 0.006757 0.000000 0.780700
```

```
options(scipen=10000)

# Bucketing rain values into 3 buckets based on amount of rain
ride_rainy <- cab %>%
  rownames_to_column() %>%
  mutate(rainy = case_when(( rain == 0)~"No Rain", (rain >0 & rain < 0.5 ) ~ "Moderate rain",(ra
in > 0.5)~"'Heavy Rain"))


ggplot(cab,aes(x= rain))+
  geom_histogram(boundary=0, binwidth = 0.15,fill='darkblue') +
    labs(
       x="rain amount",
       y="number of days",
       title='The distribution of rain amount')
```



Most days are not rainy!

```
# create a boxplot to understand the price distribution among days with different level of rain
 amounts

ggplot(ride_rainy, aes(x=rainy, y=price)) +
  geom_boxplot()
```

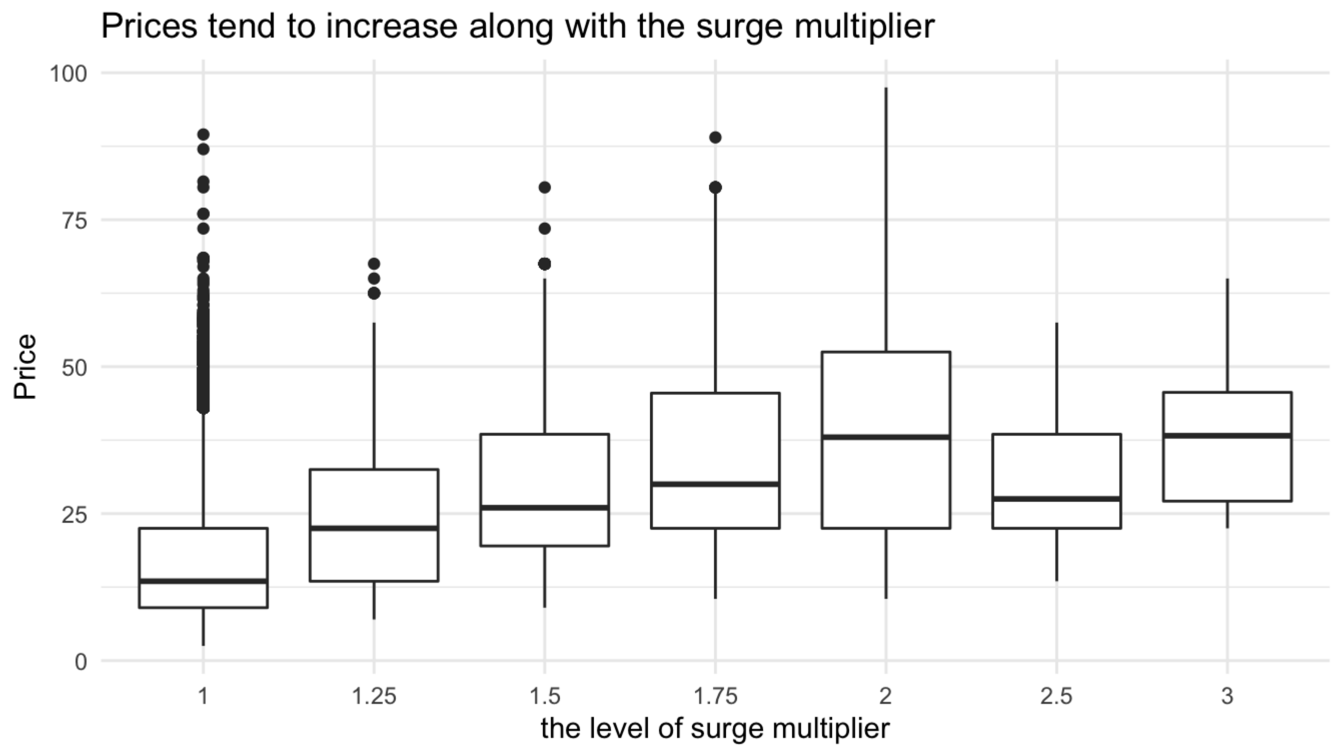


No

price variance based on buckets. Rain seems to have negligible relationship to price.

## Surge Multiplier

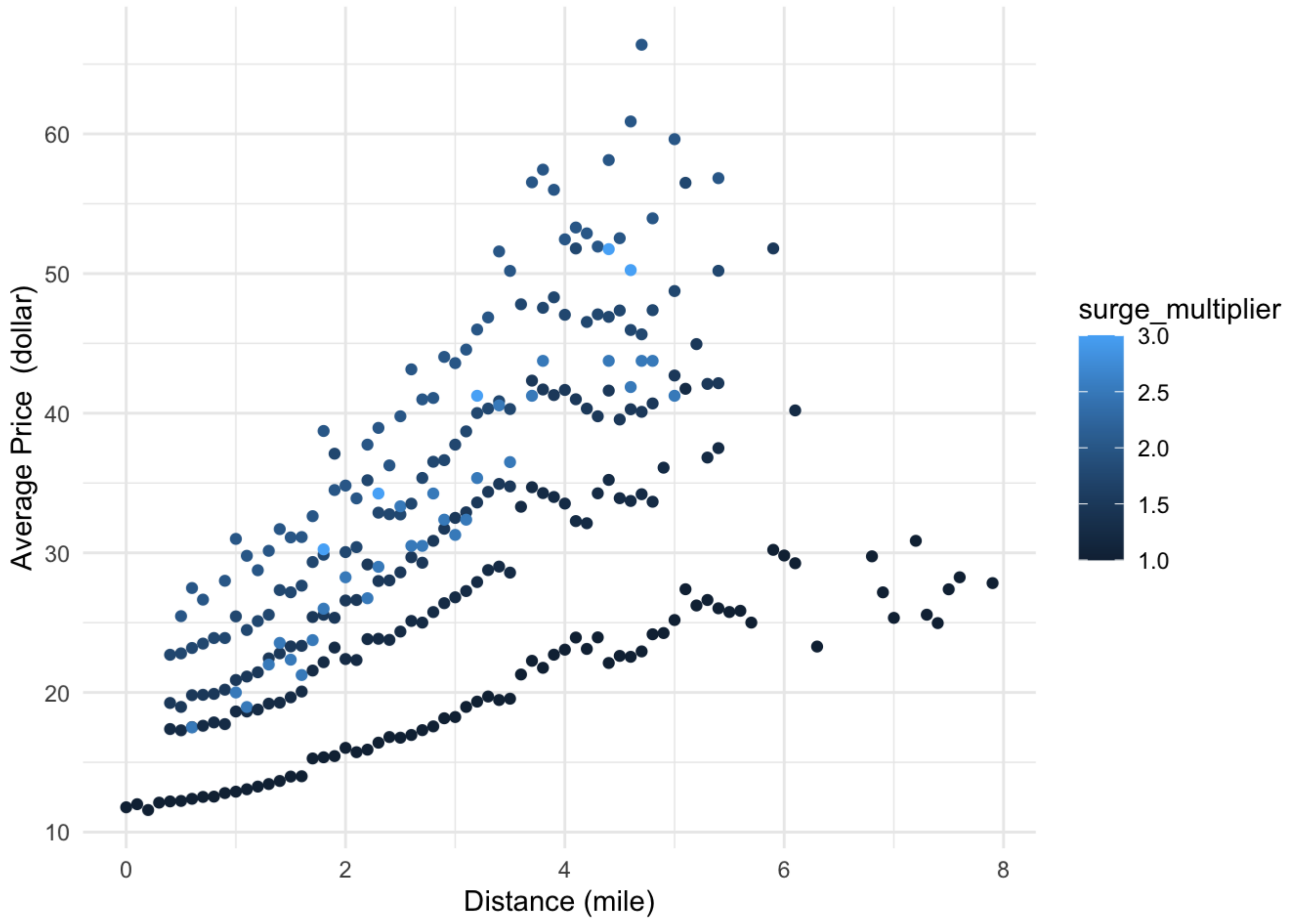
```
# Look at the relationship between price and surge multiplier by a boxplot

#further confirm the point that surge multiplier is a factor of the price, and thus it is inappr
opriate to use it as a predictor in machine learning

ggplot(cab, aes(x=factor(surge_multiplier,exclude = NA), y=price)) +
  geom_boxplot() +
    labs(
       x="the level of surge multiplier",
       y="Price",
       title="Prices tend to increase along with the surge multiplier")+
       theme(aspect.ratio = 1/2)
```



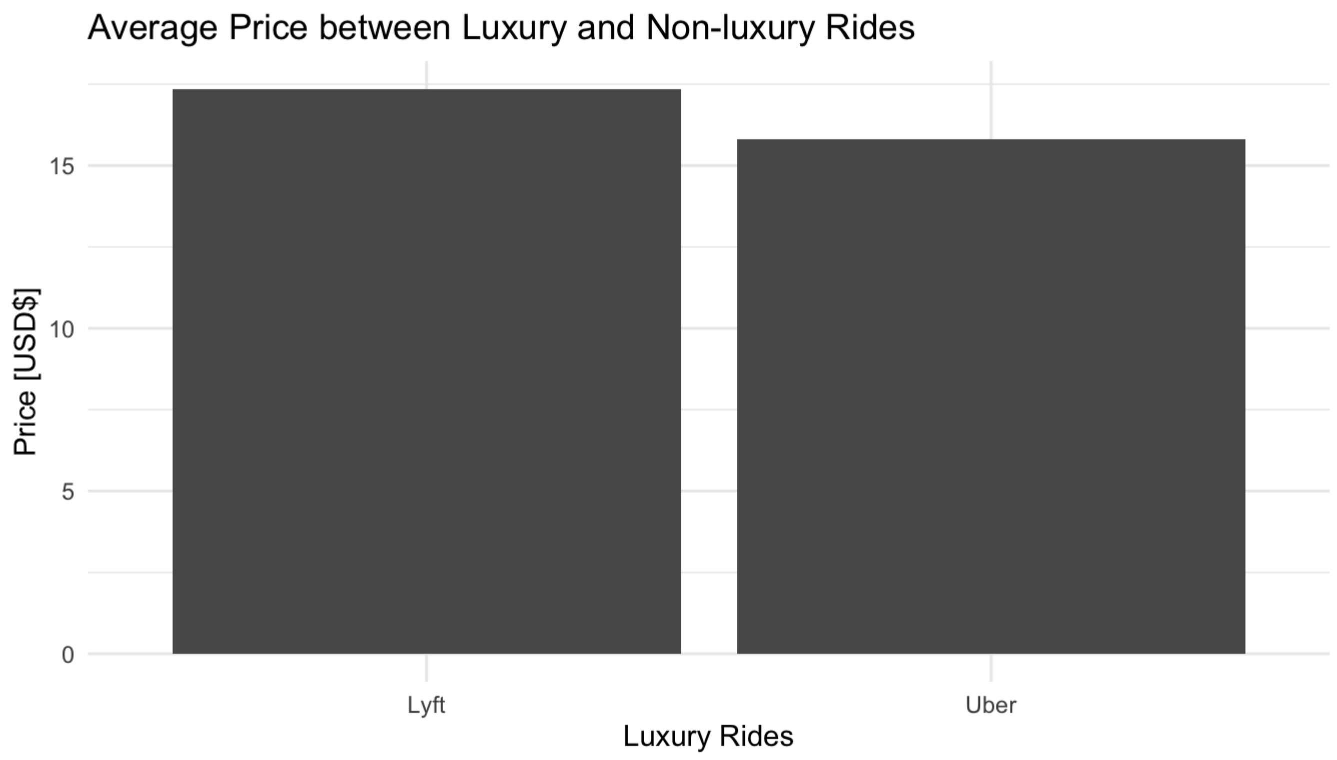
```
# Look at the relationship between price, distance and surge multiplier by a scatterplot

#further confirm the point that surge multiplier is a factor of the price, and thus it is inappr
opriate to use it as a predictor in machine learning

dist_p <- cab  %>%
  mutate(dist = round(cab$distance,digit = 1))  %>%
  select(price,dist,surge_multiplier) %>%
  group_by(dist,surge_multiplier)  %>%
  summarise(avg_p = mean(price))
```

```
## `summarise()` has grouped output by 'dist'. You can override using the `.groups` argument.
```

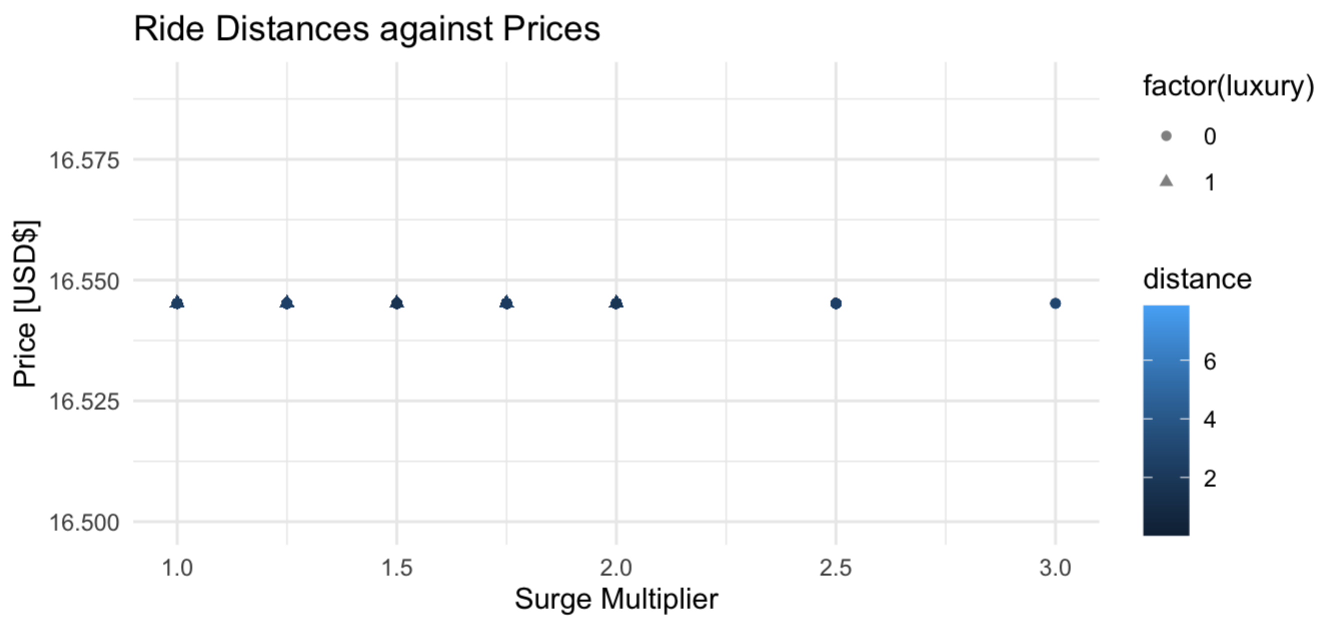
```
ggplot(dist_p, aes(x=dist,y=avg_p,color=surge_multiplier))+
  geom_point() +
  labs(
      x="Distance (mile)",
      y="Average Price  (dollar)")
```



```
tmp <- cab[,.(avg_price = mean(price)), by = company]

ggplot(tmp, aes(company, avg_price)) +
  geom_col() +
  labs(title = 'Average Price between Luxury and Non-luxury Rides',
       y = 'Price [USD$]',
       x = 'Luxury Rides') +
  theme(aspect.ratio=1/2)
```

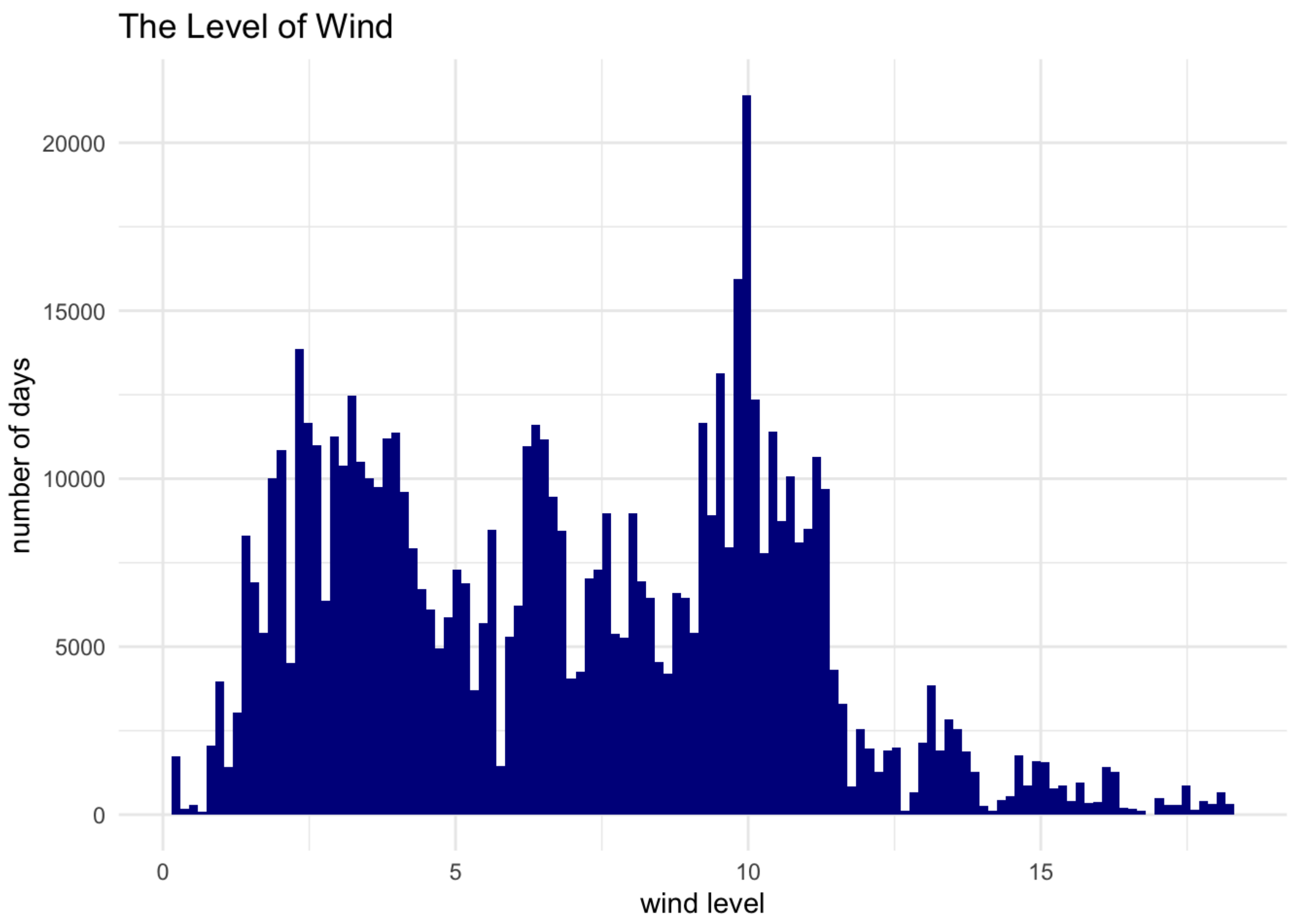


```
ggplot(cab, aes(surge_multiplier, mean(price))) +
  geom_point(alpha=0.5, aes(
    color = distance,
    shape = factor(luxury)
    )) +
  labs(title="Ride Distances against Prices",
       x = "Surge Multiplier",
       y = "Price [USD$]") +
  theme(aspect.ratio = 1/2)
```

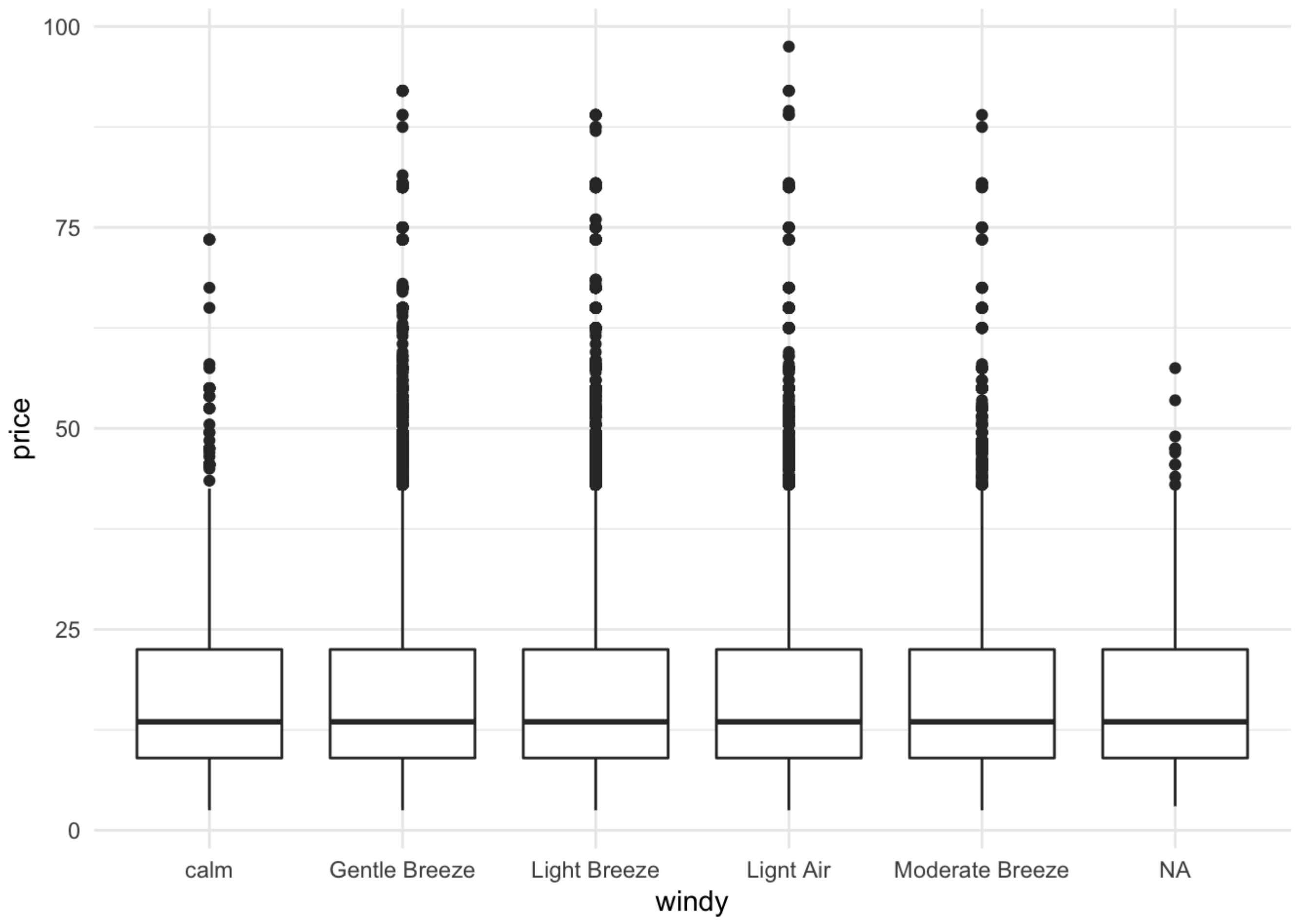


## Wind

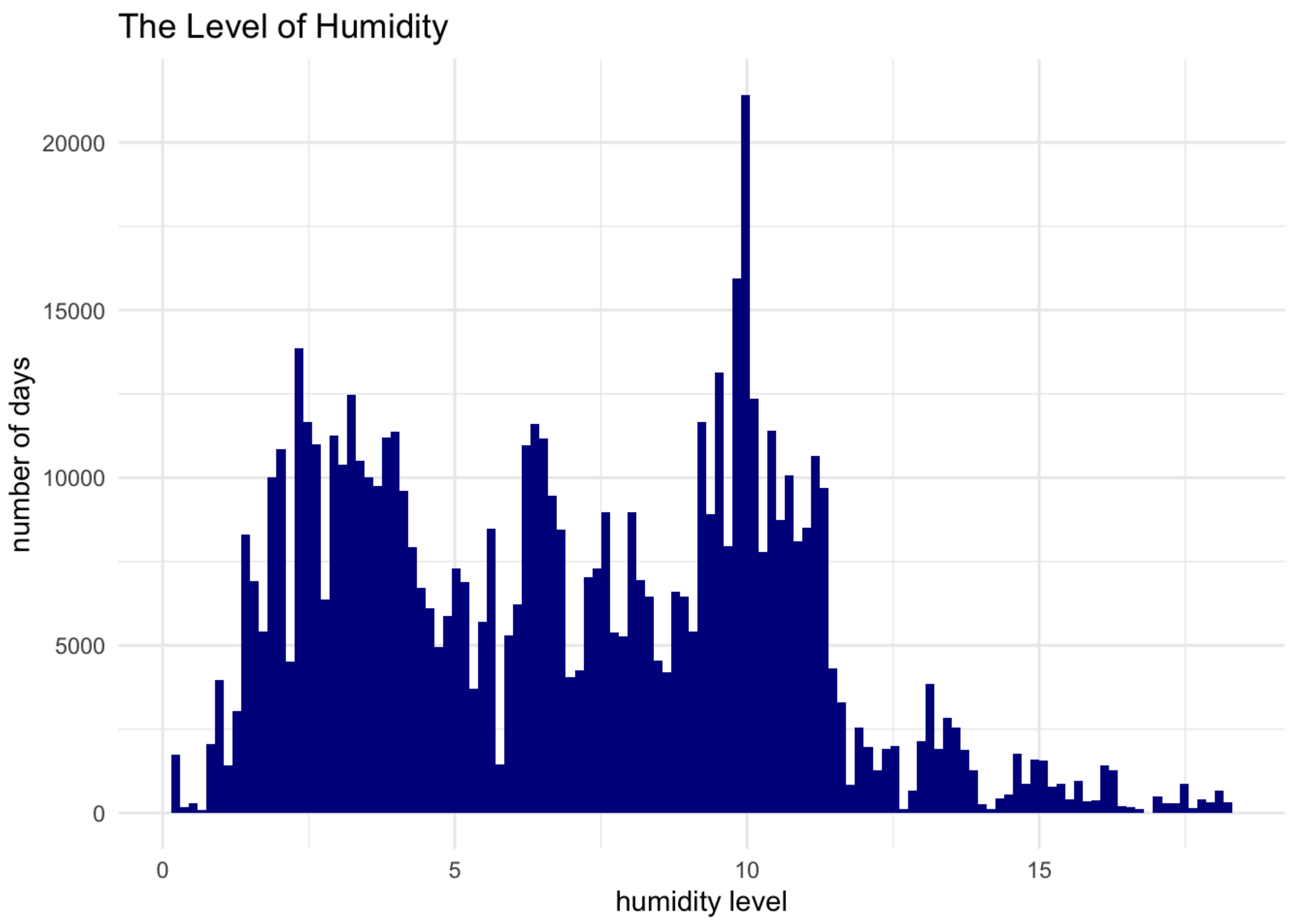
```
ride_windy<- cab %>%
  rownames_to_column() %>%
  mutate(windy = case_when((wind>=0 & wind<1)~"calm",(wind>=1 & wind<3)~"Lignt Air",(wind>=3& wi
nd<7)~"Light Breeze",(wind>=7 & wind<12)~"Gentle Breeze",(wind>=12 & wind<=18)~"Moderate Breeze"
))

ggplot(cab,aes(x=wind))+
  geom_histogram(boundary=0,binwidth = 0.15,fill='darkblue')+
  labs(
    x='wind level',
    y='number of days',
    title='The Level of Wind'
  )
```



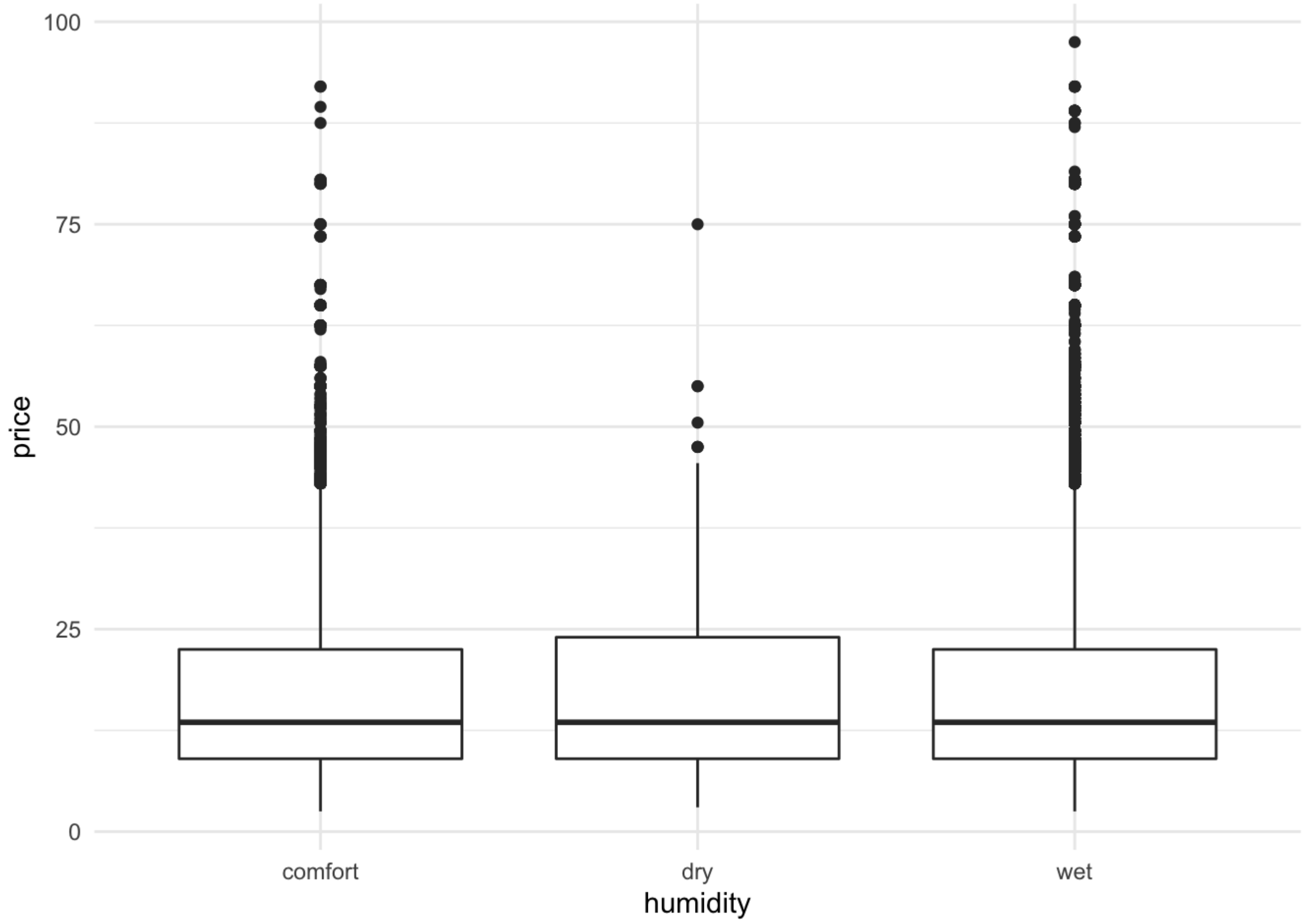
```
ggplot(ride_windy,aes(x=windy,y=price))+
  geom_boxplot()
```



```
ride_humidity<-cab %>%
  rownames_to_column() %>%
  mutate(humidity = case_when((humidity<=0.45)~"dry",(humidity>0.45 & humidity<0.6)~"comfort",(h
umidity>=0.6)~"wet"))

ggplot(cab,aes(x=wind))+
  geom_histogram(boundary=0,binwidth = 0.15,fill='darkblue')+
  labs(
    x='humidity level',
    y='number of days',
    title='The Level of Humidity'
  )
```



```
ggplot(ride_humidity,aes(x=humidity,y=price))+
  geom_boxplot()
```



## Linear Model Development

```
data <- fread('~/Google Drive/Shared drives/BA810 - Team 8A/data/clean_dum_cab.csv')
data <- subset( data, select = -id )
head(data)
```

```
##    distance price surge_multiplier  temp clouds pressure rain humidity wind
## 1:     3.03  34.0                1 41.07   0.86  1014.39    0     0.92 1.36
## 2:     1.30  18.5                1 40.86   0.87  1014.39    0     0.93 1.60
## 3:     2.71  19.5                1 40.80   0.87  1014.39    0     0.93 1.55
## 4:     2.43  10.5                1 40.81   0.89  1014.35    0     0.93 1.36
## 5:     2.71  32.0                1 40.80   0.87  1014.39    0     0.93 1.55
## 6:     2.19   8.0                1 41.02   0.87  1014.39    0     0.92 1.50
##    luxury company_Uber day_Monday day_Saturday day_Sunday day_Thursday
## 1:      1            0          1            0          0            0
## 2:      1            1          1            0          0            0
## 3:      0            1          1            0          0            0
## 4:      0            0          1            0          0            0
## 5:      0            1          1            0          0            0
## 6:      0            1          1            0          0            0
##    day_Tuesday day_Wednesday hour_01 hour_02 hour_03 hour_04 hour_05 hour_06
## 1:           0             0       0       0       1       0       0       0
## 2:           0             0       0       0       1       0       0       0
## 3:           0             0       0       0       1       0       0       0
## 4:           0             0       0       0       1       0       0       0
## 5:           0             0       0       0       1       0       0       0
## 6:           0             0       0       0       1       0       0       0
##    hour_07 hour_08 hour_09 hour_10 hour_11 hour_12 hour_13 hour_14 hour_15
## 1:       0       0       0       0       0       0       0       0       0
## 2:       0       0       0       0       0       0       0       0       0
## 3:       0       0       0       0       0       0       0       0       0
## 4:       0       0       0       0       0       0       0       0       0
## 5:       0       0       0       0       0       0       0       0       0
## 6:       0       0       0       0       0       0       0       0       0
##    hour_16 hour_17 hour_18 hour_19 hour_20 hour_21 hour_22 hour_23 size_regular
## 1:       0       0       0       0       0       0       0       0            0
## 2:       0       0       0       0       0       0       0       0            1
## 3:       0       0       0       0       0       0       0       0            1
## 4:       0       0       0       0       0       0       0       0            1
## 5:       0       0       0       0       0       0       0       0            0
## 6:       0       0       0       0       0       0       0       0            1
##    size_shared
## 1:           0
## 2:           0
## 3:           0
## 4:           0
## 5:           0
## 6:           0
```

```
rsq <- function (actual, predictions) cor(actual, predictions) ^ 2
```

```
# train test split
set.seed(810)
test_index <- sample(nrow(data), (nrow(data)*0.2)) # 80-20 split
data.test <- data[test_index]
data.train <- data[!test_index]

y.train <- data.train$price
y.test <- data.test$price
```

```
fit.lm1 <- lm(price~.-surge_multiplier,data=data.train)

yhat.train <- predict(fit.lm1)
mse.train <- mean((y.train - yhat.train)^2)

yhat.test <- predict(fit.lm1, data.test)
mse.test <- mean((y.test - yhat.test)^2)

mse.train
```

```
## [1] 12.04343
```

```
mse.test
```

```
## [1] 12.13551
```

```
summary(fit.lm1)
```

```
##
## Call:
## lm(formula = price ~ . - surge_multiplier, data = data.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.275  -2.059  -0.244   1.605  58.034
##
## Coefficients:
##                 Estimate Std. Error  t value            Pr(>|t|)
## (Intercept)   11.6091287  0.8918947   13.016 <0.0000000000000002 ***
## distance       2.8389599  0.0042766  663.841 <0.0000000000000002 ***
## temp          -0.0015414  0.0012366   -1.246              0.2126
## clouds        -0.0068687  0.0219221   -0.313              0.7540
## pressure      -0.0010162  0.0008332   -1.220              0.2226
## rain          -0.1544434  0.1413346   -1.093              0.2745
## humidity      -0.0901787  0.0698692   -1.291              0.1968
## wind           0.0026900  0.0020170    1.334              0.1823
## luxury        12.8667546  0.0108231 1188.827 <0.0000000000000002 ***
## company_Uber   0.5694978  0.0098888   57.590 <0.0000000000000002 ***
## day_Monday     0.0136423  0.0225512    0.605              0.5452
## day_Saturday   0.0412823  0.0210141    1.965              0.0495 *
## day_Sunday    -0.0056045  0.0211762   -0.265              0.7913
## day_Thursday  -0.0203086  0.0219977   -0.923              0.3559
## day_Tuesday   -0.0006005  0.0268491   -0.022              0.9822
## day_Wednesday -0.0402010  0.0315165   -1.276              0.2021
## hour_01       -0.0455677  0.0330971   -1.377              0.1686
## hour_02        0.0105907  0.0329648    0.321              0.7480
## hour_03       -0.0300100  0.0332306   -0.903              0.3665
## hour_04       -0.0443747  0.0331112   -1.340              0.1802
## hour_05       -0.0146953  0.0344083   -0.427              0.6693
## hour_06       -0.0691161  0.0336624   -2.053              0.0401 *
## hour_07       -0.0508847  0.0347341   -1.465              0.1429
## hour_08       -0.0438177  0.0351310   -1.247              0.2123
## hour_09       -0.0378896  0.0336998   -1.124              0.2609
## hour_10       -0.0500848  0.0335240   -1.494              0.1352
## hour_11        0.0450968  0.0334593    1.348              0.1777
## hour_12       -0.0083098  0.0329385   -0.252              0.8008
## hour_13        0.0217358  0.0325729    0.667              0.5046
## hour_14       -0.0377382  0.0324703   -1.162              0.2451
## hour_15       -0.0662235  0.0324616   -2.040              0.0413 *
## hour_16       -0.0585604  0.0326945   -1.791              0.0733 .
## hour_17        0.0109671  0.0331417    0.331              0.7407
## hour_18       -0.0779638  0.0334574   -2.330              0.0198 *
## hour_19       -0.0414647  0.0344062   -1.205              0.2281
## hour_20       -0.0019465  0.0345646   -0.056              0.9551
## hour_21       -0.0015978  0.0338629   -0.047              0.9624
## hour_22       -0.0023238  0.0330936   -0.070              0.9440
## hour_23       -0.0197782  0.0321692   -0.615              0.5387
## size_regular  -8.2881512  0.0108597 -763.202 <0.0000000000000002 ***
## size_shared   -9.5161190  0.0155494 -611.991 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 3.471 on 510283 degrees of freedom
## Multiple R-squared:  0.8615, Adjusted R-squared:  0.8615
## F-statistic: 7.938e+04 on 40 and 510283 DF,  p-value: < 0.00000000000000022
```

```
fit.lmdist <- lm(price~distance,data=data.train)

yhat.train <- predict(fit.lmdist)
mse.train <- mean((y.train - yhat.train)^2)

yhat.test <- predict(fit.lmdist, data.test)
mse.test <- mean((y.test - yhat.test)^2)

mse.train
```

```
## [1] 76.57733
```

```
mse.test
```

```
## [1] 76.65661
```

```
summary(fit.lmdist)
```

```
##
## Call:
## lm(formula = price ~ distance, data = data.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -22.639  -6.946  -1.681   4.916  71.017
##
## Coefficients:
##             Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 10.33513    0.02660   388.6 <0.0000000000000002 ***
## distance     2.83937    0.01078   263.3 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.751 on 510322 degrees of freedom
## Multiple R-squared:  0.1196, Adjusted R-squared:  0.1196
## F-statistic: 6.935e+04 on 1 and 510322 DF,  p-value: < 0.00000000000000022
```

```
fit.lm2 <- lm(price~distance + luxury + size_regular + size_shared,data=data.train)

yhat.train <- predict(fit.lm2)
mse.train <- mean((y.train - yhat.train)^2)

yhat.test <- predict(fit.lm2, data.test)
mse.test <- mean((y.test - yhat.test)^2)

mse.train
```

```
## [1] 12.12313
```

```
mse.test
```

```
## [1] 12.22217
```

```
summary(fit.lm2)
```

```
##
## Call:
## lm(formula = price ~ distance + luxury + size_regular + size_shared,
##     data = data.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.525  -2.025  -0.272   1.623  57.806
##
## Coefficients:
##              Estimate Std. Error t value            Pr(>|t|)
## (Intercept)  10.79384    0.01371   787.4 <0.0000000000000002 ***
## distance      2.83924    0.00429   661.8 <0.0000000000000002 ***
## luxury       12.75333    0.01068  1194.5 <0.0000000000000002 ***
## size_regular -8.28892    0.01090  -760.8 <0.0000000000000002 ***
## size_shared  -9.57255    0.01557  -614.9 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.482 on 510319 degrees of freedom
## Multiple R-squared:  0.8606, Adjusted R-squared:  0.8606
## F-statistic: 7.878e+05 on 4 and 510319 DF,  p-value: < 0.00000000000000022
```

```
fit.lm3 <- lm(price~distance + luxury + size_regular + size_shared + company_Uber,data=data.trai
n)

yhat.train <- predict(fit.lm3)
mse.train <- mean((y.train - yhat.train)^2)

yhat.test <- predict(fit.lm3, data.test)
mse.test <- mean((y.test - yhat.test)^2)

mse.train
```

```
## [1] 12.04492
```

```
mse.test
```

```
## [1] 12.13467
```

```
summary(fit.lm3)
```

```
##
## Call:
## lm(formula = price ~ distance + luxury + size_regular + size_shared +
##     company_Uber, data = data.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.228  -2.061  -0.240   1.607  58.045
##
## Coefficients:
##               Estimate Std. Error t value            Pr(>|t|)
## (Intercept)  10.442794   0.014963  697.92 <0.0000000000000002 ***
## distance      2.838866   0.004276  663.86 <0.0000000000000002 ***
## luxury       12.866644   0.010823 1188.81 <0.0000000000000002 ***
## size_regular -8.288058   0.010860 -763.19 <0.0000000000000002 ***
## size_shared  -9.516043   0.015550 -611.98 <0.0000000000000002 ***
## company_Uber  0.569206   0.009888   57.56 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.471 on 510318 degrees of freedom
## Multiple R-squared:  0.8615, Adjusted R-squared:  0.8615
## F-statistic: 6.35e+05 on 5 and 510318 DF,  p-value: < 0.00000000000000022
```

```
y.train.surge <- data.train$surge_multiplier
y.test.surge <- data.test$surge_multiplier

fit.lmsurge <- lm(surge_multiplier~.-price,data=data.train)

yhat.train <- predict(fit.lmsurge)
mse.train <- mean((y.train.surge - yhat.train)^2)

yhat.test <- predict(fit.lmsurge, data.test)
mse.test <- mean((y.test.surge - yhat.test)^2)

mse.train
```

```
## [1] 0.008878552
```

```
mse.test
```

```
## [1] 0.008532009
```

```
summary(fit.lmsurge)
```

```
##
## Call:
## lm(formula = surge_multiplier ~ . - price, data = data.train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.04696 -0.03293 -0.00616 -0.00128  1.96709
##
## Coefficients:
##                  Estimate   Std. Error  t value             Pr(>|t|)
## (Intercept)    1.086591510  0.024216404   44.870 < 0.0000000000000002 ***
## distance       0.002206601  0.000116116   19.003 < 0.0000000000000002 ***
## temp          -0.000097206  0.000033576   -2.895             0.003791 **
## clouds        -0.000508464  0.000595222   -0.854             0.392971
## pressure      -0.000052772  0.000022622   -2.333             0.019660 *
## rain          -0.001133755  0.003837467   -0.295             0.767655
## humidity       0.000438101  0.001897063    0.231             0.817364
## wind          -0.000028935  0.000054766   -0.528             0.597263
## luxury         0.000650757  0.000293864    2.214             0.026796 *
## company_Uber  -0.031382938  0.000268496 -116.884 < 0.0000000000000002 ***
## day_Monday     0.000359199  0.000612301    0.587             0.557448
## day_Saturday   0.000814831  0.000570567    1.428             0.153262
## day_Sunday     0.000339980  0.000574969    0.591             0.554318
## day_Thursday  -0.000257938  0.000597274   -0.432             0.665845
## day_Tuesday    0.000427725  0.000728996    0.587             0.557384
## day_Wednesday -0.000706462  0.000855725   -0.826             0.409048
## hour_01        0.000455025  0.000898639    0.506             0.612612
## hour_02        0.001018899  0.000895048    1.138             0.254965
## hour_03       -0.000560774  0.000902267   -0.622             0.534260
## hour_04       -0.000638868  0.000899024   -0.711             0.477317
## hour_05       -0.000372595  0.000934241   -0.399             0.690025
## hour_06       -0.000968639  0.000913990   -1.060             0.289240
## hour_07       -0.001309836  0.000943089   -1.389             0.164870
## hour_08       -0.000194225  0.000953863   -0.204             0.838651
## hour_09       -0.000048750  0.000915006   -0.053             0.957510
## hour_10       -0.001485820  0.000910231   -1.632             0.102605
## hour_11        0.000901145  0.000908476    0.992             0.321232
## hour_12       -0.000546583  0.000894334   -0.611             0.541093
## hour_13        0.003092464  0.000884408    3.497             0.000471 ***
## hour_14       -0.000486162  0.000881622   -0.551             0.581332
## hour_15       -0.000565864  0.000881386   -0.642             0.520863
## hour_16       -0.000851515  0.000887711   -0.959             0.337445
## hour_17       -0.000593995  0.000899852   -0.660             0.509188
## hour_18        0.000250886  0.000908423    0.276             0.782411
## hour_19       -0.000881612  0.000934184   -0.944             0.345311
## hour_20        0.001643051  0.000938486    1.751             0.079990 .
## hour_21       -0.000005441  0.000919433   -0.006             0.995279
## hour_22        0.000879474  0.000898546    0.979             0.327692
## hour_23       -0.000100821  0.000873448   -0.115             0.908105
## size_regular  -0.000131001  0.000294859   -0.444             0.656837
## size_shared   -0.017930649  0.000422193  -42.470 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.09423 on 510283 degrees of freedom
## Multiple R-squared:  0.0328, Adjusted R-squared:  0.03272
## F-statistic: 432.6 on 40 and 510283 DF,  p-value: < 0.00000000000000022
```

# Lasso Regression

```
f <- as.formula(price ~ .)
x.train <- model.matrix(price~.-surge_multiplier, data.train)[,-1]
x.test <- model.matrix(price~.-surge_multiplier, data.test)[,-1]

fit.lasso <- cv.glmnet(x.train, y.train, alpha = 1, nfolds = 10)
fit.ridge <- cv.glmnet(x.train, y.train, alpha = 0, nfolds = 10)
```

```
# train
yhat.train.lasso <- predict(fit.lasso, x.train, s = fit.lasso$lambda.min)
mse.train.lasso <- mean((y.train - yhat.train.lasso)^2)
mse.train.lasso
```

```
## [1] 12.04854
```

```
# test
yhat.test.lasso <- predict(fit.lasso, x.test, s = fit.lasso$lambda.min)
mse.test.lasso <- mean((y.test - yhat.test.lasso)^2)
mse.test.lasso
```

```
## [1] 12.13753
```

```
coef(fit.lasso)
```

```
## 41 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)   10.5978714
## distance       2.7461901
## temp           .
## clouds         .
## pressure       .
## rain           .
## humidity       .
## wind           .
## luxury        12.6641518
## company_Uber   0.3247694
## day_Monday     .
## day_Saturday   .
## day_Sunday     .
## day_Thursday   .
## day_Tuesday    .
## day_Wednesday  .
## hour_01        .
## hour_02        .
## hour_03        .
## hour_04        .
## hour_05        .
## hour_06        .
## hour_07        .
## hour_08        .
## hour_09        .
## hour_10        .
## hour_11        .
## hour_12        .
## hour_13        .
## hour_14        .
## hour_15        .
## hour_16        .
## hour_17        .
## hour_18        .
## hour_19        .
## hour_20        .
## hour_21        .
## hour_22        .
## hour_23        .
## size_regular  -7.9078942
## size_shared   -9.1052035
```

# Ridge Regression

```
# train
yhat.train.ridge <- predict(fit.lasso, x.train, s = fit.lasso$lambda.min)
mse.train.ridge <- mean((y.train - yhat.train.ridge)^2)
mse.train.ridge
```

```
## [1] 12.04854
```

```
# test
yhat.test.ridge <- predict(fit.ridge, x.test, s = fit.ridge$lambda.min)
mse.test.ridge <- mean((y.test - yhat.test.ridge)^2)
mse.test.ridge
```

```
## [1] 12.55845
```

```
coef(fit.ridge)
```

```
## 41 x 1 sparse Matrix of class "dgCMatrix"
##                           1
## (Intercept)   11.3325763813
## distance       2.6246896785
## temp          -0.0009815372
## clouds        -0.0001338819
## pressure      -0.0005101227
## rain          -0.1505219135
## humidity      -0.0698138807
## wind           0.0018247211
## luxury        11.9338033392
## company_Uber   0.3839817320
## day_Monday     0.0118054296
## day_Saturday   0.0331468975
## day_Sunday    -0.0063140459
## day_Thursday  -0.0040157478
## day_Tuesday    0.0167211056
## day_Wednesday -0.0222941497
## hour_01       -0.0269675606
## hour_02        0.0268567989
## hour_03       -0.0100584415
## hour_04       -0.0198104541
## hour_05       -0.0011648767
## hour_06       -0.0434785455
## hour_07       -0.0325072897
## hour_08       -0.0189032799
## hour_09       -0.0208227647
## hour_10       -0.0317876412
## hour_11        0.0561040946
## hour_12        0.0048883363
## hour_13        0.0332942805
## hour_14       -0.0164228883
## hour_15       -0.0463173233
## hour_16       -0.0401387023
## hour_17        0.0273499397
## hour_18       -0.0539629855
## hour_19       -0.0226628453
## hour_20        0.0153293207
## hour_21        0.0163227753
## hour_22        0.0167115928
## hour_23       -0.0056625105
## size_regular  -7.2587524619
## size_shared   -8.6548641687
```

# Random Forest

```
f1 <- as.formula(price ~ distance + luxury + size + company)
```

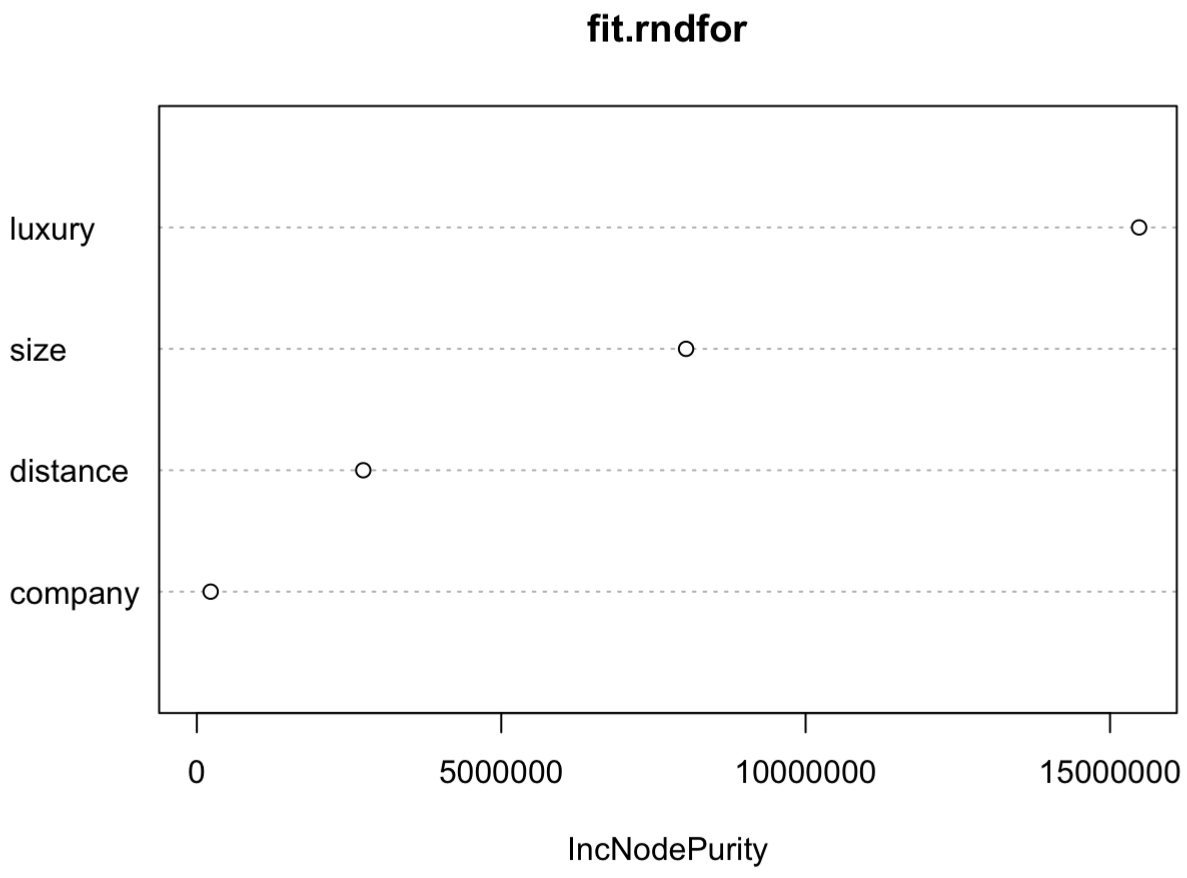
Unlike linear regression models, random forest and boosted trees do not require dumified data to run an analysis; some models even perorm better on non-dummy data. Therefore, we will use non-dummy data for the following models.

```
dd.test <- cab[test_index]
dd.train <- cab[!test_index]
```

```
x.train <- model.matrix(f1, dd.train)[, -1]
y.train <- data.train$price

x.test <- model.matrix(f1, dd.test)[, -1]
y.test <- data.test$price
```

```
fit.rndfor <- randomForest(f1,
dd.train,
ntree=200,
do.trace=F)
```

```
varImpPlot(fit.rndfor)
```



```
# TRAIN
yhat.rndfor <- predict(fit.rndfor, dd.train)
mse.tree <- mean((yhat.rndfor - y.train) ^ 2)
print(mse.tree)
```

```
## [1] 18.66631
```

```
# TEST
yhat_t.rndfor <- predict(fit.rndfor, dd.test)
mse_t.tree <- mean((yhat_t.rndfor- y.test)^2)
mse_t.tree
```

```
## [1] 18.62814
```

## Boosted Forest

```
f_dum <- as.formula(price ~ distance + luxury + size_regular + size_shared + company_Uber)

fit.train.btree <- gbm(f_dum,
data = data.train,
distribution = "gaussian",
n.trees = 500,
shrinkage = 0.1,
interaction.depth = 2
)

yhat.btree <- predict(fit.train.btree, data.train, n.trees = 500)
mse.train.btree <- mean((yhat.btree - y.train) ^ 2)
print(mse.train.btree)
```

```
## [1] 8.292962
```

```
yhat.test.btree <- predict(fit.train.btree, data.test, n.trees = 500)
mse.test.btree <- mean((yhat.test.btree - y.test) ^ 2)
print(mse.test.btree)
```

```
## [1] 8.36394
```