

EDA for Final Project: Uber/Lyft Ride Price Prediction

Yuanbiao Wang, Hao Wang, Rain Wu, Dean Huang

Introduction

Brief recap about our proposal

The core question of this project is to identify the pricing model of Uber / Lyft rides, and get an intuition about how to get cheap rides in Boston. The main tools for such analysis is regression.

The dataset used across this project is Uber and Lyft Dataset Boston, MA. The predictors we selected and cared about for the regression analysis includes:

- Distance of the ride
- Weather condition, including a categorical variable for the weather, and other statistics like temperature, humidity, visibility, etc.
- Pick up / Destination location
- Ride type: Uber or Lyft; more specific information about the vehicle, like Lyft XL, UberX, etc.

The interesting question includes: is Uber cheaper than Lyft or vice versa? What's the pricing model between distance and price? Since the data is around seasonal holidays, do time and location affect the price?

A preview of what did in this milestone

Specifically, we did the following things:

- Subsampled the original dataset to create a scale that is processible by R, and compared the distribution of predictor variables to confirm that the distributions won't be off the original ones.
- Created visualizations and analysis without modeling to explore how time, location and weather conditions affect the price, and preprocess them in a proper way for future modeling.
- Built several models to perform regression analysis, including linear, polynomial regression using only the distance, and linear regression with interaction terms between distance and ride type (Uber and Lyft). We used these models to analyze the pricing model and try to answer the first two questions.

What we will do next

We will be focusing on building a predictive model that utilizes all of the predictor variables we cared about. After that, we will try to interpret them with visuals and formal tests, and we will also use cross validation and RMSE metric to properly evaluate the predictive capability of such models.

In the end, we will attempt to answer all the questions we proposed and provide the Bostonians a practical ride suggestion supported by data.

Data Explorations

Downsample Distribution

Since the original dataset is too big to process in R, we downsample the dataset to a smaller dataset. However, we'd like to make sure the distribution of the smaller dataaset is similar to the original dataset.

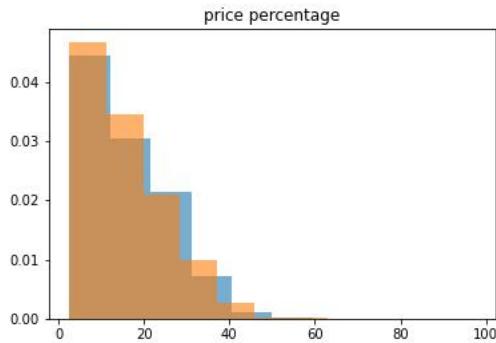


Figure 1: price_percentage

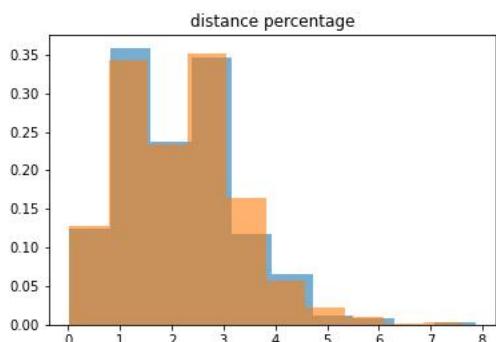


Figure 2: distance_percentage

It seems that the downsampling distribution is pretty similar to the original one besides the price distribution, which is a little bit off. We might want to adjust this in the future.

Evaluation of number of orders within the same range

It's intuitive for Uber/Lyft to use supply and demand to price the rides. Therefore, the price of the rides might be much higher if there are a lot of demands. We did some data preprocessing in python by calculating the number of rides within certain time intervals (15 mins here). If there are a lot of rides in the same time range, we expect the price of the rides will be higher.

```
# read the data
rideshare = read.csv("data/sampled_rideshare.csv")
slide_window_df = read.csv("./data/rides_slide_window.csv")
hist(slide_window_df$number_of_orders_within_15_mins)
```

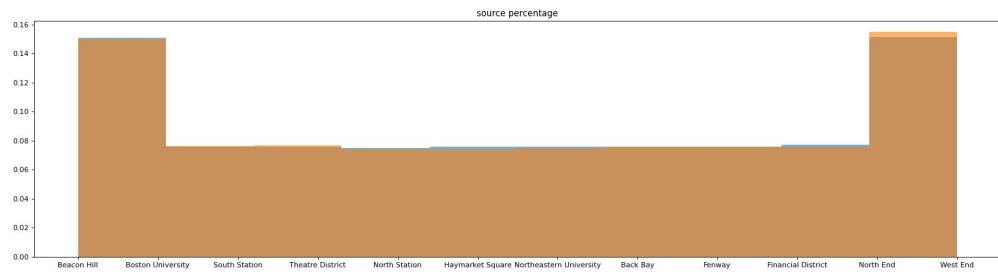


Figure 3: source_percentage

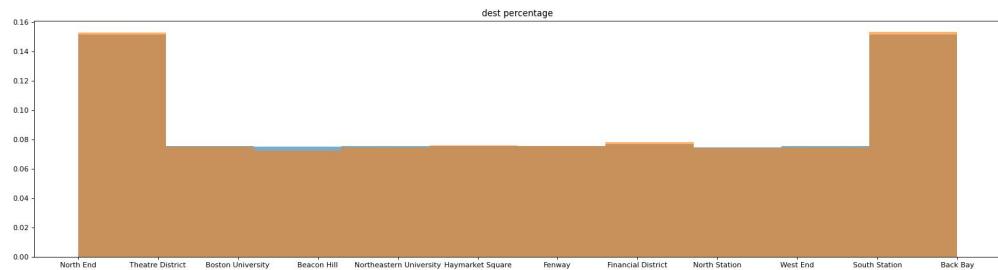


Figure 4: dest_percentage

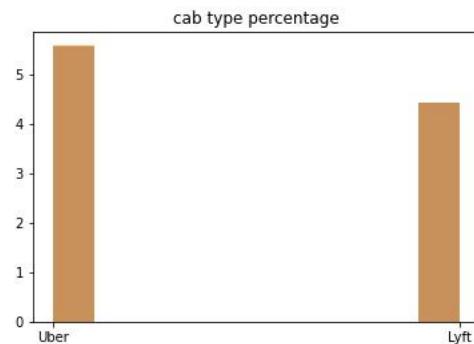


Figure 5: cab_type_percentage

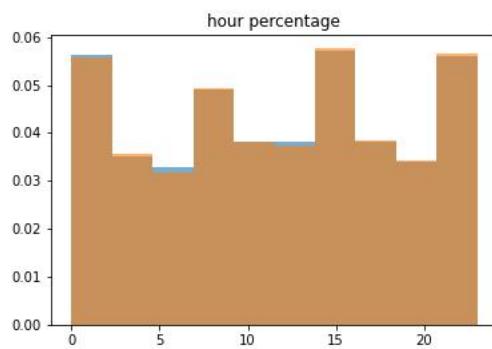
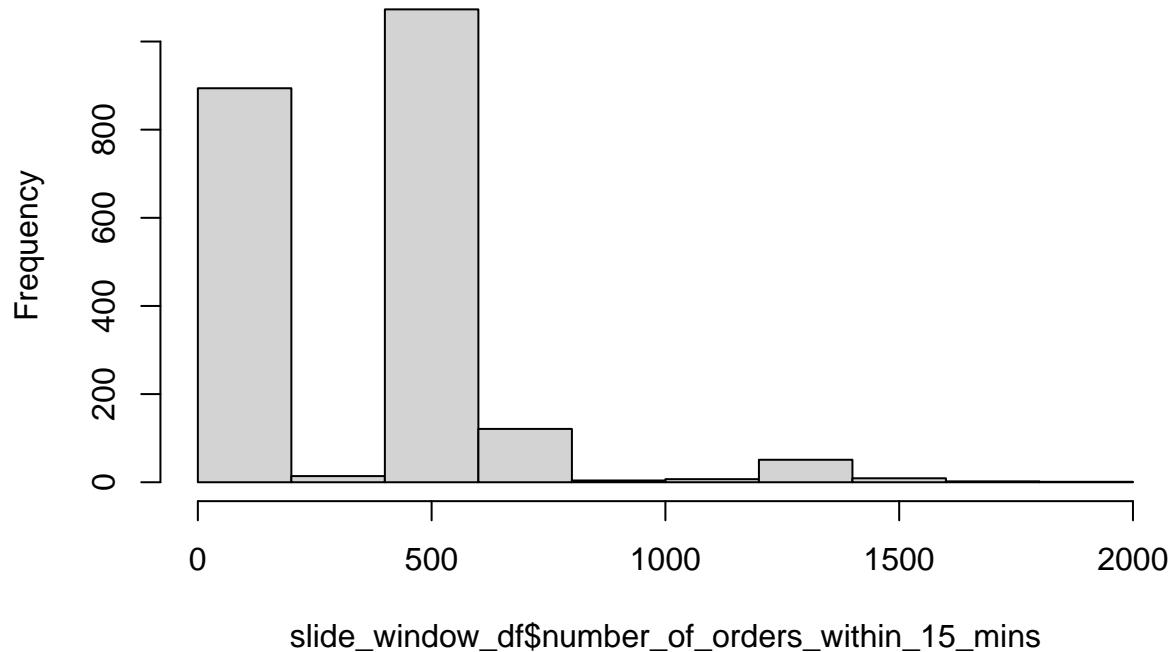


Figure 6: hour_percentage

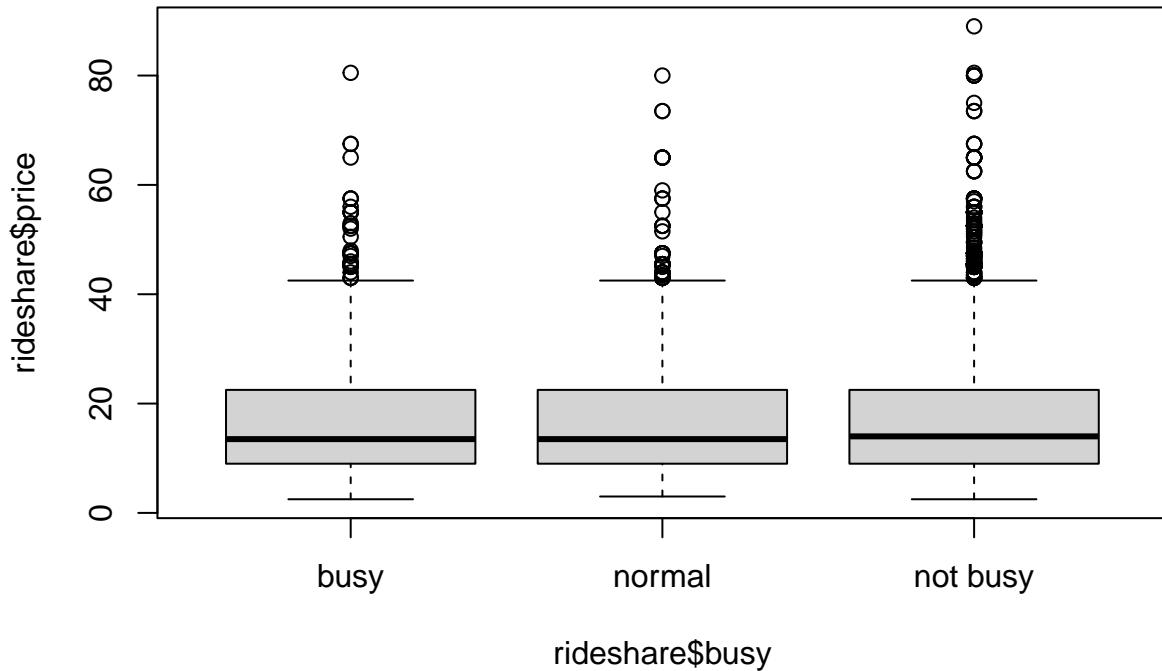
Histogram of slide_window_df\$number_of_orders_within_15_mins



Based on the histogram, it seems that the number of rides is not very continuous. However, we could certainly see that there are certain time ranges (peak hours) with a lot of more rides. Since the value is not very continuous, it might be helpful to convert them into categorical variables based on some time ranges or use a decision tree model in the future.

```
rideshare$busy = as.factor(  
  ifelse(rideshare$number_of_orders_within_15_mins < 500,  
    "not busy",  
    ifelse(rideshare$number_of_orders_within_15_mins < 1000,  
      "normal",  
      "busy")))
```

```
boxplot(rideshare$price ~ rideshare$busy)
```



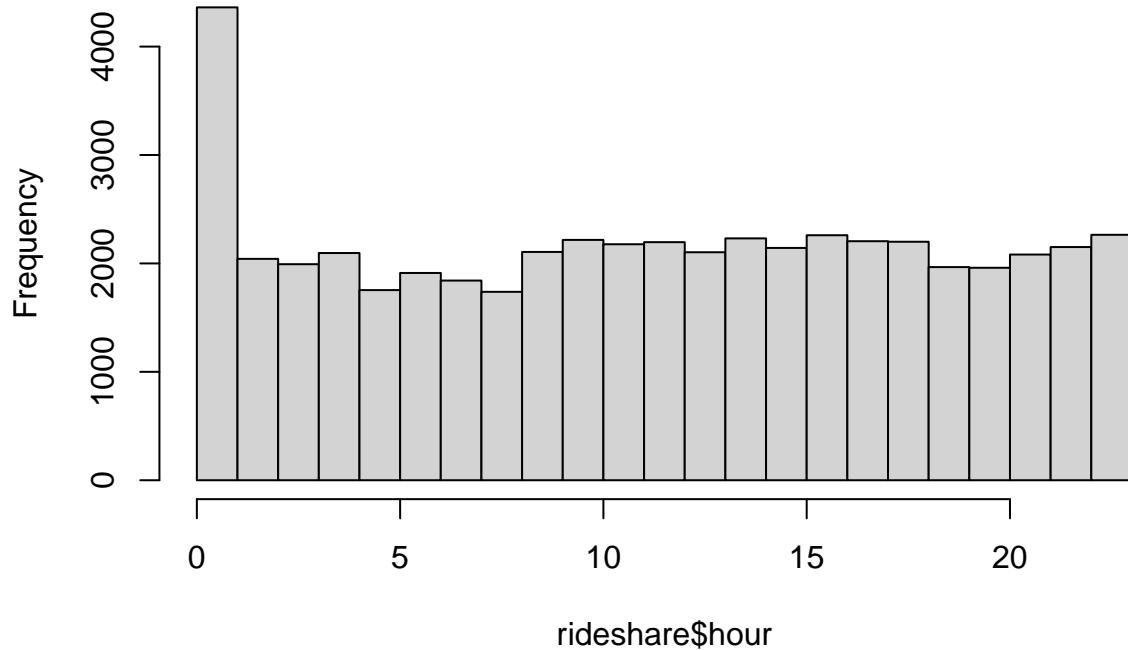
```
summary(aov(rideshare$price ~ rideshare$busy))
```

```
##              Df  Sum Sq Mean Sq F value Pr(>F)
## rideshare$busy     2    135   67.51    0.78  0.458
## Residuals      46036 3983021   86.52
## 3961 observations deleted due to missingness
```

Based on the boxplot and the anova result, there is no significant difference between the price and the number of orders in the same time range. This is counter-intuitive. It might be due to the data generation process. The data provider might presample some data to make each hour contain similar amount of orders. Since it does not represent the true distribution, then our calculation might be misled.

```
hist(rideshare$hour)
```

Histogram of rideshare\$hour



Preprocessing of data: Categorical or Numerical

1 MonthDayHour

Since the data comes solely from the span of two months, it makes sense to simply use the first day of Nov. as the baseline and the days as numerical in the range [1,61] and simply ignore the month column.

```
n_sample = dim(rideshare)
for(i in 1:50000){
  if(rideshare[i,"month"] == "12"){
    rideshare[i,"day"] = rideshare[i,"day"] + 30
  }
}
```

For the hours columns, it makes sense to create another categorical variable that divides the hours into different periods of the day

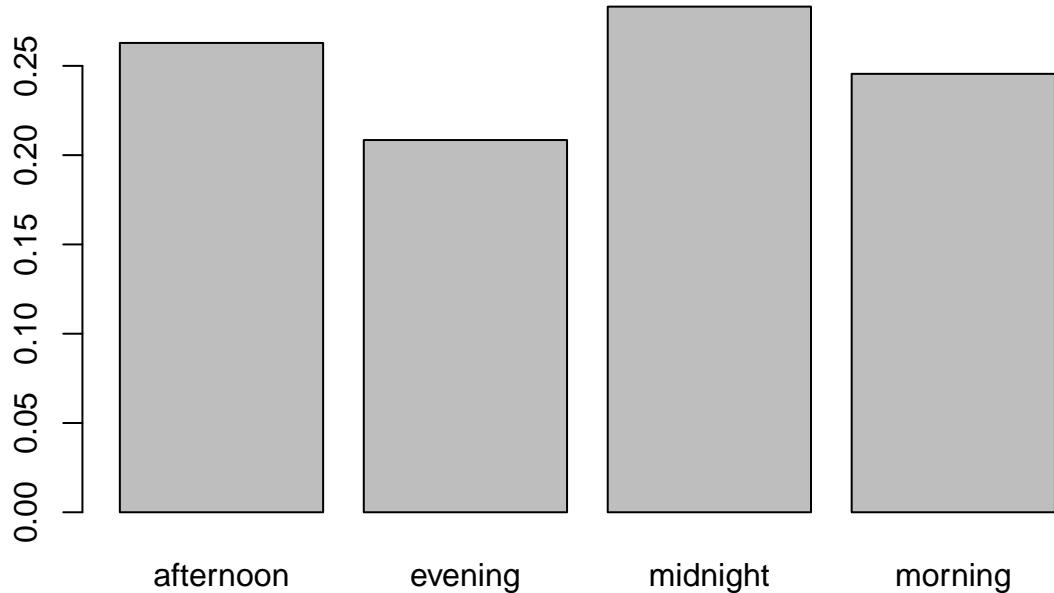
```
tmp = rep(0,50000)
for(i in 1:50000){
  if((rideshare[i,"hour"] > 6) && (rideshare[i,"hour"] <= 12)){
    tmp[i]="morning"
  }
  if((rideshare[i,"hour"] > 12) && (rideshare[i,"hour"] <= 18)){
    tmp[i]="afternoon"
  }
}
```

```

}
if((rideshare[i,"hour"] > 18) && (rideshare[i,"hour"] <= 23)){
  tmp[i]="evening"
}
if((rideshare[i,"hour"] >= 0) && (rideshare[i,"hour"] <= 6)){
  tmp[i]="midnight"
}
}
rideshare$period_of_day = tmp
barplot(prop.table(table(rideshare$period_of_day)))
library("ggpubr")

```

Loading required package: ggplot2

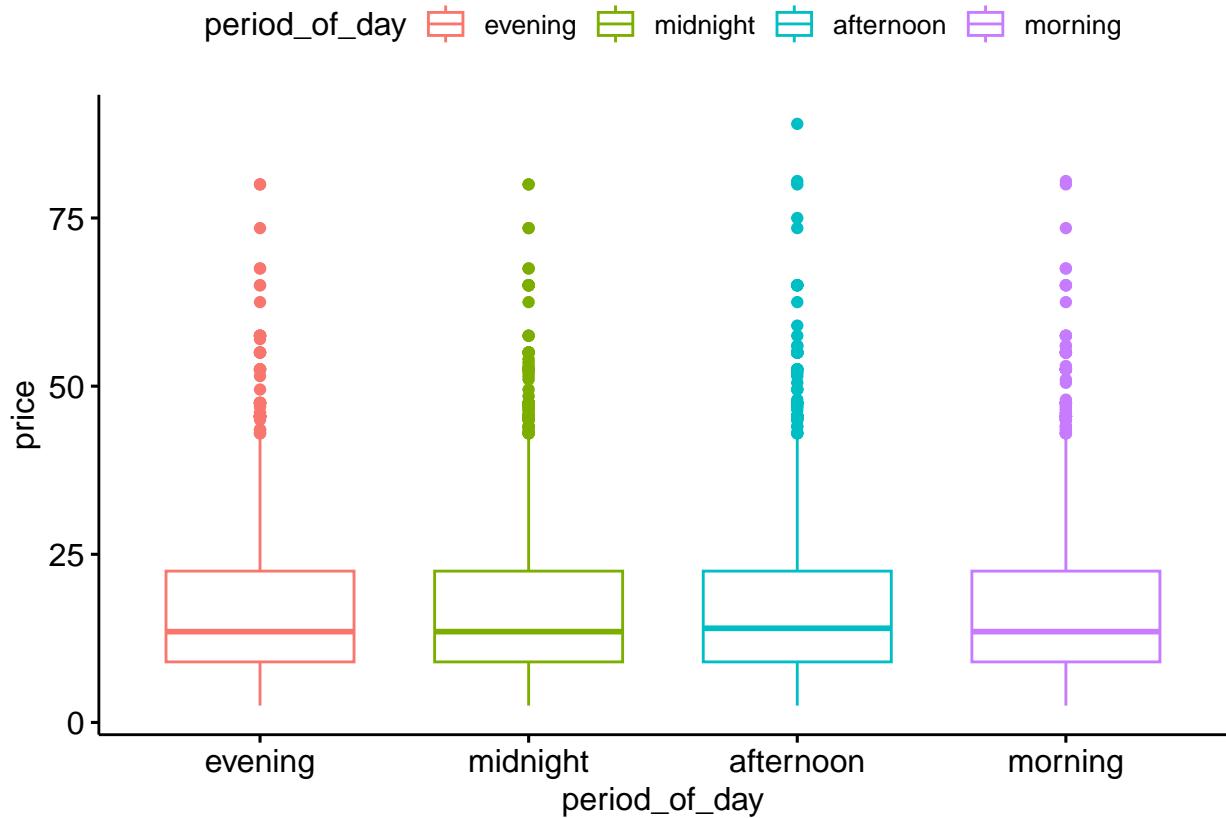


```

ggboxplot(rideshare, x = "period_of_day", y = "price",
          color = "period_of_day",
          ylab = "price", xlab = "period_of_day")

```

Warning: Removed 3961 rows containing non-finite values ('stat_boxplot()').



And we see from the plot that the proportion is close to being even among the categories, which also justifies our categorization choice.

```

morning_data = na.omit(rideshare$price[(rideshare$period_of_day == "morning")])
afternoon_data = na.omit(rideshare$price[(rideshare$period_of_day == "afternoon")])
evening_data = na.omit(rideshare$price[(rideshare$period_of_day == "evening")])
midnight_data = na.omit(rideshare$price[(rideshare$period_of_day == "midnight")])
mean_morning = mean(na.omit(rideshare$price[(rideshare$period_of_day == "morning")]))
mean_afternoon = mean(na.omit(rideshare$price[(rideshare$period_of_day == "afternoon")]))
mean_evening = mean(na.omit(rideshare$price[(rideshare$period_of_day == "evening")]))
mean_midnight = mean(na.omit(rideshare$price[(rideshare$period_of_day == "midnight")]))

mean_df = data.frame(morning = mean_morning,
                      afternoon = mean_afternoon,
                      evening = mean_evening,
                      midnight = mean_midnight)

# Pairwise t-test for the means
t.test(morning_data, afternoon_data)

##
## Welch Two Sample t-test
##
## data: morning_data and afternoon_data
## t = -0.054946, df = 23331, p-value = 0.9562
## alternative hypothesis: true difference in means is not equal to 0

```

```

## 95 percent confidence interval:
## -0.2446809 0.2313368
## sample estimates:
## mean of x mean of y
## 16.58600 16.59267

t.test(morning_data, evening_data)

##
## Welch Two Sample t-test
##
## data: morning_data and evening_data
## t = 0.75625, df = 20262, p-value = 0.4495
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -0.1562115 0.3524752
## sample estimates:
## mean of x mean of y
## 16.58600 16.48786

t.test(morning_data, midnight_data)

##
## Welch Two Sample t-test
##
## data: morning_data and midnight_data
## t = 0.76612, df = 23859, p-value = 0.4436
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -0.1425729 0.3255426
## sample estimates:
## mean of x mean of y
## 16.58600 16.49451

t.test(afternoon_data, evening_data)

##
## Welch Two Sample t-test
##
## data: afternoon_data and evening_data
## t = 0.82178, df = 20416, p-value = 0.4112
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -0.1451710 0.3547788
## sample estimates:
## mean of x mean of y
## 16.59267 16.48786

t.test(afternoon_data, midnight_data)

##

```

```

## Welch Two Sample t-test
##
## data: afternoon_data and midnight_data
## t = 0.83904, df = 25020, p-value = 0.4015
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -0.1311459 0.3274597
## sample estimates:
## mean of x mean of y
## 16.59267 16.49451

t.test(evening_data, midnight_data)

##
## Welch Two Sample t-test
##
## data: evening_data and midnight_data
## t = -0.052915, df = 20475, p-value = 0.9578
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -0.2528630 0.2395691
## sample estimates:
## mean of x mean of y
## 16.48786 16.49451

res.ftest <- aov(price ~ period_of_day, data = rideshare)

res.ftest

## Call:
## aov(formula = price ~ period_of_day, data = rideshare)
##
## Terms:
##          period_of_day Residuals
## Sum of Squares      110   3983046
## Deg. of Freedom       3     46035
##
## Residual standard error: 9.301727
## Estimated effects may be unbalanced
## 3961 observations deleted due to missingness

```

Baseline Model

Visualization

In this section we will aim to build a very simple model to analyze the relationship between ride distance and ride price. As is stated earlier, we didn't use the entire dataset because it is too slow to process them all in R. So we randomly sampled 10% of the observations and use the downsampled version in the subsequent model building and analysis. We will start with some simple visualizations to explore the relationship between ride price and ride distance.

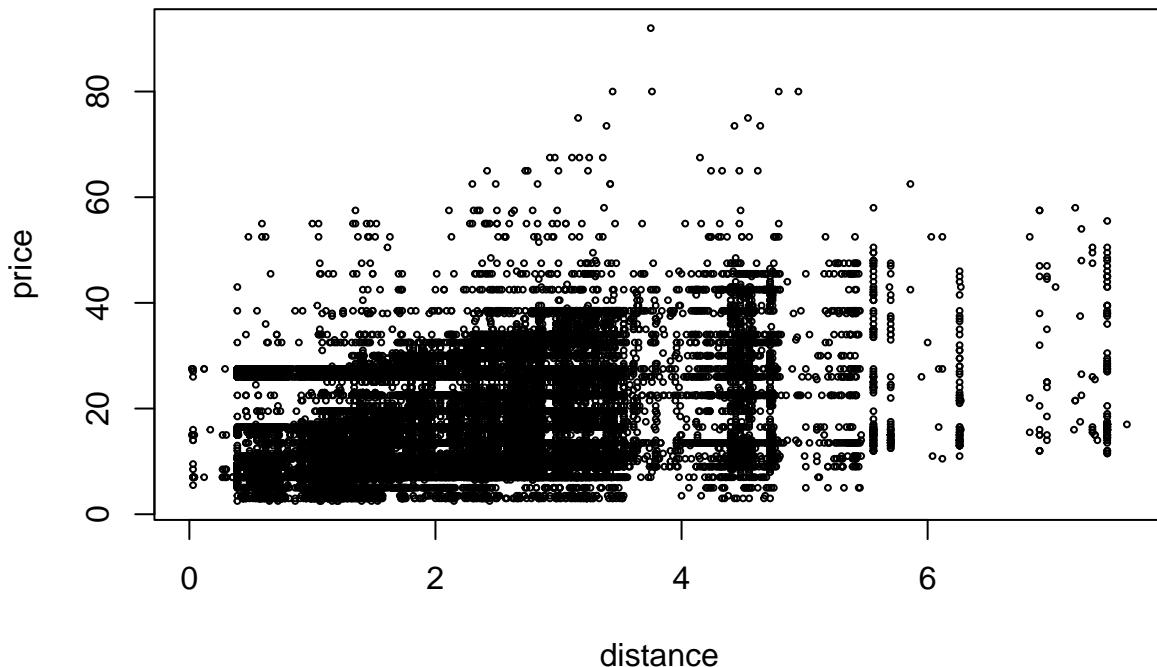
```

# drop rows with price being NA
rideshare = rideshare[is.na(rideshare$price) == F & is.na(rideshare$cab_type) == F, ]

# train / test split
rideshare.train = rideshare[1:40000, ]
rideshare.test = rideshare[40001:45984, ]

# scatter plot for ride price vs ride distance
plot(price ~ distance, data=rideshare.train, cex=0.4)

```



From the scatter plot we can see there is a positive relationship between ride distance and price, although we can see that the variability of the response at the same predictor value is fairly big, and we can expect that the R^2 for a simple linear model would not be very ideal.

Simple Linear model

The next step is to actually fit a simple linear regression model and check the assumptions for simple linear regression. We call it `lm.simple`.

```

RMSE = function(y, yhat) {
  return (mean((y - yhat) ^ 2))
}

lm.simple = lm(price ~ distance, data=rideshare.train)

```

```

# model summary
summary(lm.simple)

## 
## Call:
## lm(formula = price ~ distance, data = rideshare.train)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -20.607  -6.986  -1.697   4.819  71.138 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 10.39595   0.09549 108.87 <2e-16 ***
## distance     2.79105   0.03878  71.98 <2e-16 ***  
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 8.772 on 39998 degrees of freedom
## Multiple R-squared:  0.1147, Adjusted R-squared:  0.1146 
## F-statistic:  5180 on 1 and 39998 DF,  p-value: < 2.2e-16 

# RMSE, which can also be calculated via residual standard error and df
y.pred.train = predict(lm.simple)
y.pred.test = predict(lm.simple, newdata=rideshare.test)
lm.simple.train.rmse = RMSE(rideshare.train$price, y.pred.train)
lm.simple.test.rmse = RMSE(rideshare.test$price, y.pred.test)

# print out the RMSE result
data.frame(trainRMSE=lm.simple.train.rmse, testRMSE=lm.simple.test.rmse)

##    trainRMSE testRMSE
## 1    76.9443      NA

```

From the point estimates we can make the rough interpretation: first, the intercept is 10.36, indicating that a “starting price” would be around 10.36 dollar, even if it’s a super short ride; second, the slope for the *distance* is 2.83 meaning that every 1 mile would be estimated to cost you 2.82 dollar more in general cases. The *p*-value for the slope is less than 2×10^{-16} , so we can say that the association is very very significant.

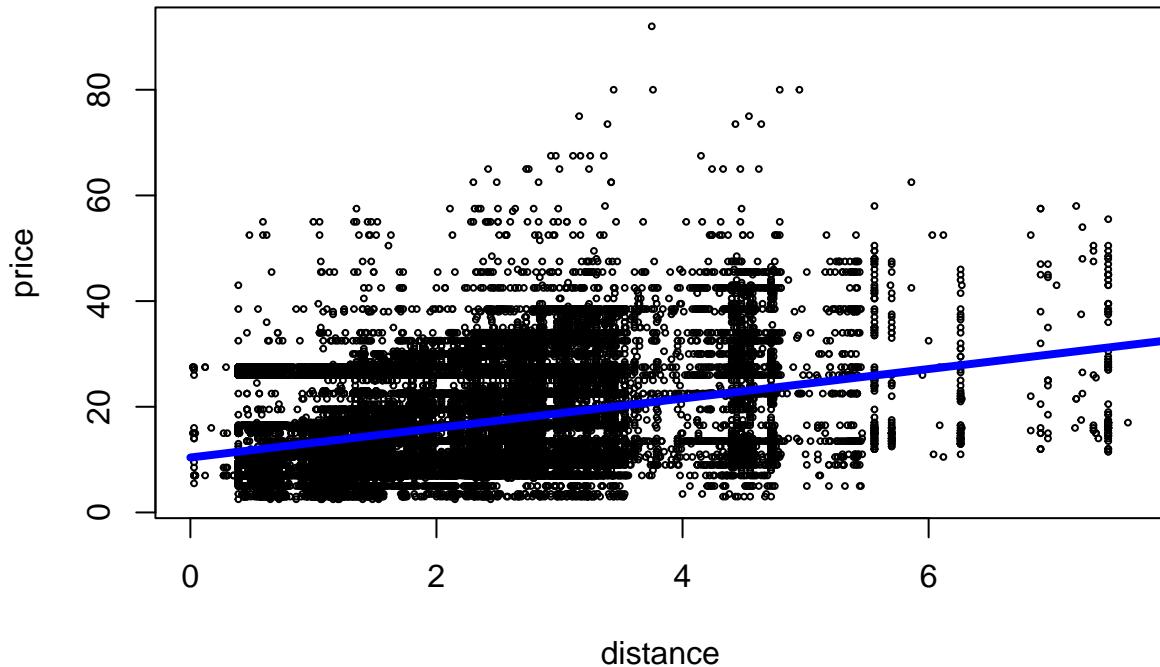
R^2 is roughly 0.1189. It’s rather distant from 1. It means that nearly 90% of the variability is due the residual errors. We get a pretty high RMSE. This is not surprising as we expect more factors aside from sheer distance would affect the price as we learned they are considered into the pricing model. Also, we notice that the test RMSE is not significantly higher than that of the training, so the model is not overfitting at all (which is also within our expectation, as the model is rather simple and has a very low variance).

Further more we plot the prediction line to see if it matches the scatter plot we created before.

```

x.plot = seq(0, 8, 0.1)
y.pred.plot = predict(lm.simple, newdata=data.frame(distance=x.plot))
plot(price ~ distance, data=rideshare.train, cex=0.4)
lines(x.plot, y.pred.plot, col="blue", lwd=4)

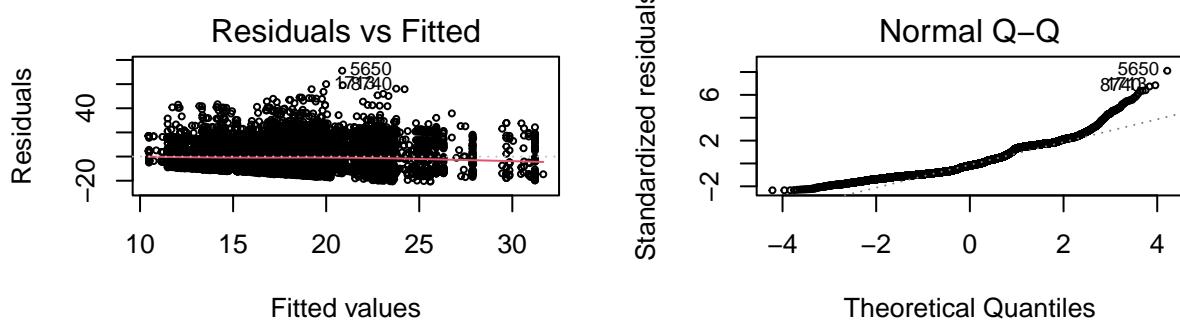
```



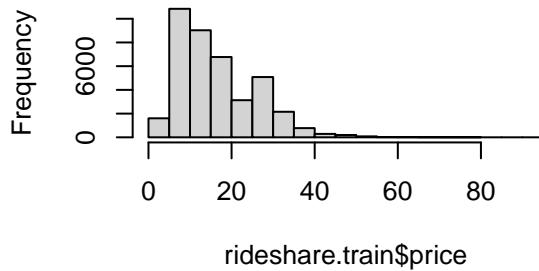
Next we check the assumptions for this linear model and see if we would need any transformations to relieve the skewness issue.

```
# residual and QQ plot
par(mfrow=c(2, 2))
plot(lm.simple, which=1:2, cex=0.5)

# y distribution plot
hist(rideshare.train$price)
```



Histogram of rideshare.train\$price



According to the residual plot, the linearity and constant variance assumption are all poorly conformed. There are a lot more positive residuals than negative ones, so `lm.simple` is underestimating the prices. The variability of residuals seem greater when the predicted prices are within range [18, 23]. From the QQ plot and the histogram we can see that the normality is also not met. The data looks quite right-skew, and slightly bi-modal.

Polynomial Regression

Since the linear model has a relatively not so satisfying performance, and the linearity is not well met, we think it might be worth a try to use polynomial model. We want to start with a quadratic model

```
lm.quadratic = lm(price ~ poly(distance, 2, raw=T), data=rideshare.train)
```

```
# quadratic model summary
summary(lm.quadratic)
```

```
##
## Call:
## lm(formula = price ~ poly(distance, 2, raw = T), data = rideshare.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -20.399  -7.002  -1.706   4.817  71.121 
##
## Coefficients:
```

```

##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)           10.06018   0.15335 65.601 < 2e-16 ***
## poly(distance, 2, raw = T)1  3.11317   0.12147 25.628 < 2e-16 ***
## poly(distance, 2, raw = T)2 -0.06082   0.02174 -2.798  0.00514 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.771 on 39997 degrees of freedom
## Multiple R-squared:  0.1148, Adjusted R-squared:  0.1148
## F-statistic:  2595 on 2 and 39997 DF,  p-value: < 2.2e-16

# F-test
anova(lm.simple, lm.quadratic)

```

```

## Analysis of Variance Table
##
## Model 1: price ~ distance
## Model 2: price ~ poly(distance, 2, raw = T)
##   Res.Df   RSS Df Sum of Sq    F   Pr(>F)
## 1 39998 30777772
## 2 39997 3077170  1     602.36 7.8295 0.005142 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

From the p -values from both the t -test of the quadratic term and the F -test between `lm.simple` and `lm.quadratic` we can see that the added quadratic term is not of much help, so we will not further more consider it.

Uber or Lyft?

Here we want to research into a very intriguing question: is Uber and Lyft exhibiting different pricing models? To answer this question, we will experiment with another linear model incorporating the type of the ride into the consideration. We call this model `lm.brand`. It considers the `distance` and `cab_type`, as well as their interaction

```

lm.brand = lm(price ~ (distance + cab_type) ^ 2, data=rideshare.train)
summary(lm.brand)

```

```

##
## Call:
## lm(formula = price ~ (distance + cab_type)^2, data = rideshare.train)
##
## Residuals:
##      Min      1Q      Median      3Q      Max 
## -23.135  -6.627  -1.608   4.764  69.509 
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)           10.04103   0.14245 70.488 < 2e-16 ***
## distance              3.31992   0.05812 57.122 < 2e-16 ***
## cab_typeUber          0.54321   0.19113  2.842  0.00448 **  
## distance:cab_typeUber -0.95523   0.07767 -12.298 < 2e-16 ***

```

```

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.722 on 39996 degrees of freedom
## Multiple R-squared:  0.1249, Adjusted R-squared:  0.1248
## F-statistic:  1902 on 3 and 39996 DF,  p-value: < 2.2e-16

# compare this to the simple model
anova(lm.simple, lm.brand)

## Analysis of Variance Table
##
## Model 1: price ~ distance
## Model 2: price ~ (distance + cab_type)^2
##   Res.Df   RSS Df Sum of Sq    F    Pr(>F)
## 1 39998 3077772
## 2 39996 3042355  2      35417 232.8 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# RMSEs
y.pred.brand.train = predict(lm.brand)
y.pred.brand.test = predict(lm.brand, newdata=rideshare.test)
lm.brand.train.rmse = RMSE(rideshare.train$price, y.pred.brand.train)
lm.brand.test.rmse = RMSE(rideshare.test$price, y.pred.brand.test)
data.frame(trainRMSE=lm.brand.train.rmse, testRMSE=lm.brand.test.rmse)

##   trainRMSE testRMSE
## 1 76.05888     NA

```

From the summary of the model we can derive some useful insights. When `cab_type` is `Lyft`, the regression formula is

$$\text{price} = 10.28 + 3.25 \times \text{distance}$$

; While if `cab_type` is `Uber`, the regression formula is

$$\text{price} = 10.33 + 2.51 \times \text{distance}$$

. This suggests that `Lyft` have a slightly lower starting price but would have a significantly higher fare rate w.r.t the distance compared to Uber rides. The pricing model for these two companies are in fact quite different. The *t*-test result supports this claim: the coefficients for `cab_typeUber` is not significant with a *p*-value as big as 0.791, while the coefficients for `distance:cab_typeUber` has a *p*-value less than 2×10^{-16} , indicating extreme significance. So we would have the impression that in Boston during 2018 in the winter, `Lyft` would be more expensive generally if we are only considering the distance.

To explore whether this is an improvement compared to the naive linear regression, we perform an *F*-test and calculated the RMSEs. The *p*-value for the *F*-test is less than 2.2×10^{-16} , suggesting a significant improve. Also both train & test RMSEs are better compared to `lm.simple`.

Next, we will provide some visualizations to support this intuition. We will plot the scattered points as well as the prediction line, but in different colors to separate these two cab brands: pink for `Lyft`, and Orange for `Uber`.

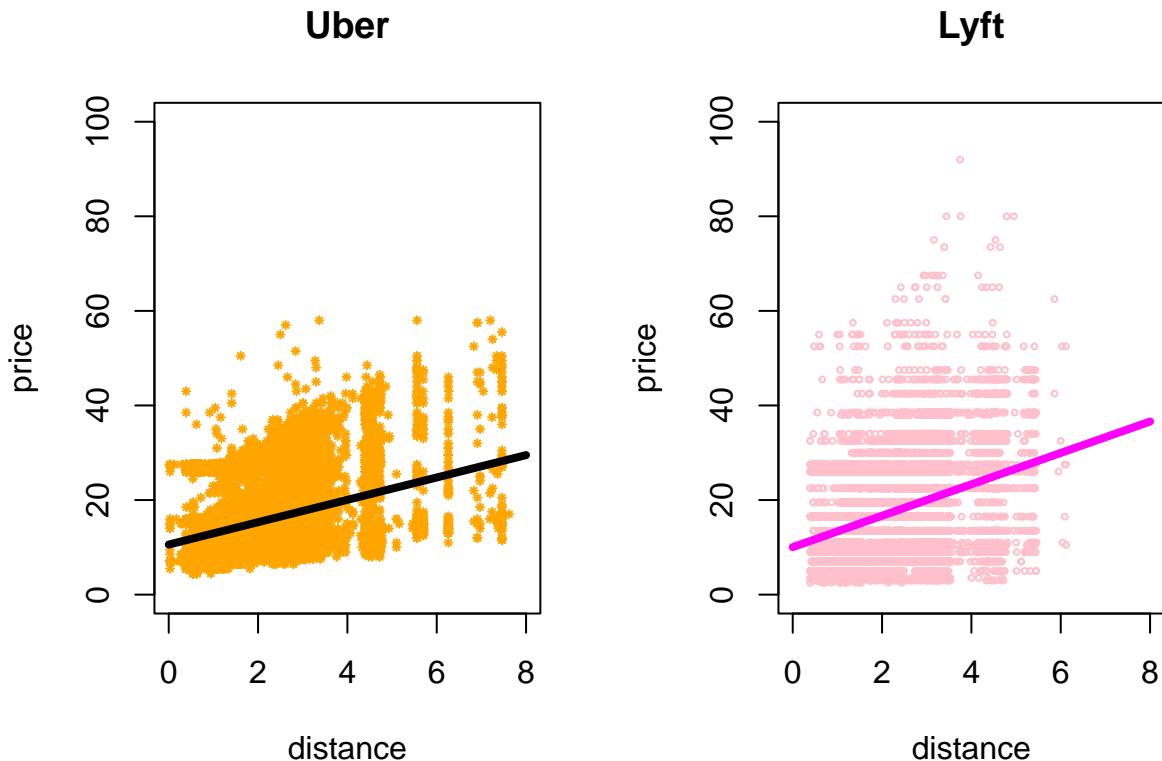
```

par(mfrow=c(1,2))
rideshare.train.uber = rideshare.train[rideshare.train$cab_type=="Uber", ]
rideshare.train.lyft = rideshare.train[rideshare.train$cab_type=="Lyft", ]

# Uber plot
plot(price ~ distance, data=rideshare.train.uber, col="orange",
      cex=0.4, pch=8, xlim=c(0, 8), ylim=c(0, 100),
      main="Uber")
y.pred.plot.uber = predict(lm.brand,
                           newdata=data.frame(distance=x.plot, cab_type=rep("Uber",
                           length(x.plot))))
lines(x.plot, y.pred.plot.uber, col="black", lwd=4)

# Lyft plot
plot(price ~ distance, data=rideshare.train.lyft, col="pink",
      cex=0.4,
      xlim=c(0, 8),
      ylim=c(0, 100), main="Lyft")
y.pred.plot.lyft = predict(lm.brand,
                           newdata=data.frame(distance=x.plot, cab_type=rep("Lyft",
                           length(x.plot))))
lines(x.plot, y.pred.plot.lyft, col="Magenta", lwd=4)

```



From the prediction line we can see that Lyft has a steeper prediction line, suggesting a significant higher fare rate. However, according to the scatter plots, we can easily find that Uber ride fares have better linearity w.r.t the ride distance, and has a smaller variance in the price.