Data Visualization for Uber Interview Challenge

Xinyi Ye

2019/7/25

Contents

List of Figures

List of Tables

Import Libraries & Datasets

```
library(wordcloud) # Word Cloud
```

```
## Loading required package: RColorBrewer
```

```
library(rpart) # Decision Tree
library(caret) # Feature Selection
```

```
## Warning: package 'caret' was built under R version 3.5.2
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(rpart.plot) # Plotting Decision Tree
```

```
## Warning: package 'rpart.plot' was built under R version 3.5.2
```

```r
library(pROC) # ROC curve and AUC
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(randomForest) # Random Forest
```

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

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

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

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

```r
library(nnet) # Neural Network
library(corrplot) # For correlation plor
```

```
## corrplot 0.84 loaded
```

```r
library(oddsratio) # For odds-probability conversion
```

```
## Warning: package 'oddsratio' was built under R version 3.5.2
```

```r
library(knitr) # For knitting tables
library(kableExtra) # For knitting tables
```

```
## Warning: package 'kableExtra' was built under R version 3.5.2
```

```r
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

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

```
## The following object is masked from 'package:kableExtra':
##
##     group_rows
```

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

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

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

```r
library(devtools) # For neural network plots
```

```
## Warning: package 'devtools' was built under R version 3.5.2
## Loading required package: usethis
## Warning: package 'usethis' was built under R version 3.5.2
```

```
library(reshape)
```

```
##
## Attaching package: 'reshape'
## The following object is masked from 'package:dplyr':
##
##     rename
```

```
library(devtools)
library(plyr)
```

```
## ------------------------------------------------------------------------------
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
## ------------------------------------------------------------------------------
##
## Attaching package: 'plyr'
## The following objects are masked from 'package:reshape':
##
##     rename, round_any
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
library(NeuralNetTools) # Neural Network Plots
source_url('https://gist.githubusercontent.com/fawda123/7471137/raw/466c1474d0a505ff044412703516c34f1a4
```

```
## SHA-1 hash of file is 74c80bd5ddbc17ab3ae5ece9c0ed9beb612e87ef
```

```
urlRemote  <- "/Users/yxy/Dropbox/Xinyi Ye/Business Analytics Case Study/"
pathGithub <- "2_4_Uber Interview Challenge/Dataset/"
fileName   <- "ds_challenge_v2_1_data.csv"
uber <- read.csv(paste0(urlRemote, pathGithub, fileName))
```

Data Description

This dataset has 54681 records of Uber drivers who signed up during 1/1/2016 through 1/31/2016. It includes data fields related to the driver's signup date, location, vehicle information and first trip date. The process a new driver may complete starts with signup, through background check, vehicle registration, adding vehicle information, to finishing the first trip.

The background check and vehicle registration are optional after signup, hence there are 21875 missing records of background check date and 41547 missing records of vehicle registration. The period for background check is from 1/1/2016 to 3/25/2016, and the period for vehicle registration is from 1/1/2016 to 1/1/3/26. There are only 13223 records (41458 missing records) of vehicle information (make, model, year) since some drivers

might not have finished the registration process. Moreover, only 6137 drivers finished their first trip from 1/4/2016 to 2/29/2016.

```
# Change the time to the appropriate format for summary.
uber$signup_date <- as.Date(format(as.Date(uber$signup_date,
                                           format="%m/%d/%Y"), "20%y/%m/%d"))
uber$bgc_date <- as.Date(format(as.Date(uber$bgc_date,
                                        format="%m/%d/%Y"), "20%y/%m/%d"))
uber$vehicle_added_date <- as.Date(format(as.Date(uber$vehicle_added_date,
                                                  format="%m/%d/%Y"), "20%y/%m/%d"))
uber$first_completed_date <- as.Date(format(as.Date(uber$first_completed_date,
                                                    format="%m/%d/%Y"), "20%y/%m/%d"))
summary(uber)
```

```
##        id          city_name           signup_os      signup_channel
##  Min.   :    1   Berton :20117              : 6857   Organic :13427
##  1st Qu.:13671   Strark :29557   android web:14944   Paid    :23938
##  Median :27341   Wrouver: 5007   ios web    :16632   Referral:17316
##  Mean   :27341                   mac        : 5824
##  3rd Qu.:41011                   other      : 3648
##  Max.   :54681                   windows    : 6776
##
##   signup_date            bgc_date          vehicle_added_date
##  Min.   :2016-01-01   Min.   :2016-01-01   Min.   :2016-01-01
##  1st Qu.:2016-01-07   1st Qu.:2016-01-16   1st Qu.:2016-01-18
##  Median :2016-01-14   Median :2016-01-25   Median :2016-01-28
##  Mean   :2016-01-14   Mean   :2016-01-25   Mean   :2016-01-29
##  3rd Qu.:2016-01-22   3rd Qu.:2016-02-02   3rd Qu.:2016-02-10
##  Max.   :2016-01-30   Max.   :2016-03-25   Max.   :2016-03-26
##                       NA's   :21785        NA's   :41547
##   vehicle_make   vehicle_model    vehicle_year   first_completed_date
##          :41458          :41458   Min.   :   0   Min.   :2016-01-04
##  Toyota : 3219   Civic  :  689   1st Qu.:2008   1st Qu.:2016-01-18
##  Honda  : 1845   Corolla:  688   Median :2013   Median :2016-01-27
##  Nissan : 1311   Camry  :  683   Mean   :2011   Mean   :2016-01-26
##  Ford   :  778   Accord :  595   3rd Qu.:2015   3rd Qu.:2016-02-04
##  Hyundai:  677   Prius V:  522   Max.   :2017   Max.   :2016-02-29
##  (Other): 5393   (Other):10046   NA's   :41458   NA's   :48544
```

There are in a total of 11 variables, 5 categorical and 6 numeric.

```
length(unique(uber$id))
```

```
## [1] 54681
```

Every id is unique.

Pre-Analysis

This section consists of the pre-analyses of 3 parts: Driver Registration, Vehicle Registration/Information and First Trip Completion.

```
# Drivers who NEVER completed their first ride
uber.not.comp <- uber[which(is.na(uber$first_completed_date)), ]
# Drivers who completed their first ride
```

```
uber.comp <- uber[which(!is.na(uber$first_completed_date)), ]
uber$complete <- 0
uber$complete[which(!is.na(uber$first_completed_date))] <- 1
```

```
#  Create a subset with drivers only with registered vehicle
uber_v <- uber[which(!is.na(uber$vehicle_added_date)), ]
uber_v <- uber[which(uber_v$vehicle_year != 0), ]
#summary(uber_v)
```

```
# Create subsets by whether a driver with rigistered vehicle completed the first trip or not
uber_v.not.comp <- uber_v[which(is.na(uber_v$first_completed_date)), ]
uber_v.comp <- uber_v[which(!is.na(uber_v$first_completed_date)), ]
uber_v$complete <- 0
uber_v$complete[which(!is.na(uber_v$first_completed_date))] <- 1
```

Driver Registration

Signup Source

What is the most popular signup source?

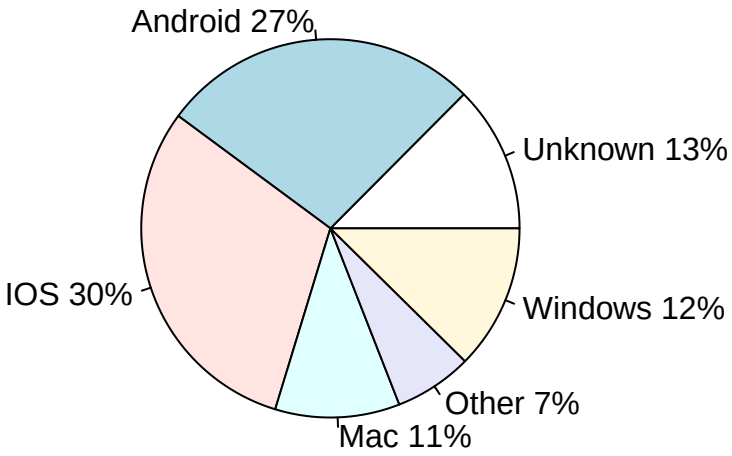


Figure 1: Distribution of Signup Sources

[Interpretation]

Figure 1 shows that the two mobile sources - IOS and Andriod take up 30% and 27%, respectively. In a total, they account for 57% of the signups. As for PC source, Windows and Mac take up 12% and 11%, respectively. The total percentege of unknown and other sources are 20%.

[Implication]

The drivers are most likely to sign up through mobile devices. The reason might be that Uber app is run on cell phones, so users tend to sign up with mobile devices. Nearly 1/4 of users chose to sign up trough websites on PC, which also makes up an important component of the signup source.

Signup Channel

What is the most common signup channel?

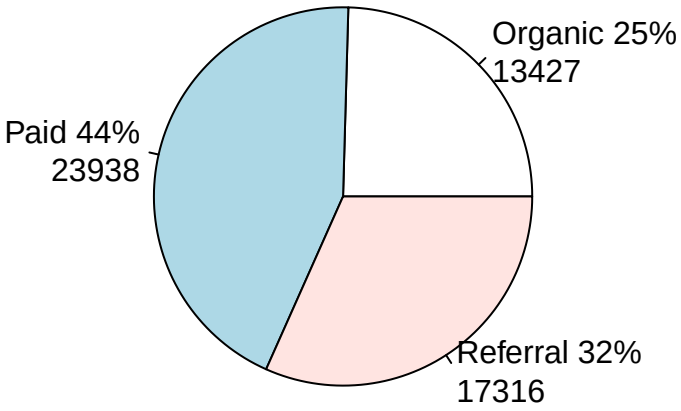


Figure 2: Distribution of Signup Channels

[**Interpretation**]

Figure 2 shows that 44% drivers are paid, 32% are through referral and only 25% are organic.

[**Implication**]

The organic channel is the least effective way when it comes to attracting new drivers. Most drivers signed up either through referrals or were simply paid. In both way, they can get rewards. Signup bonus is very important for new drivers.

City

Which city are the drivers from?

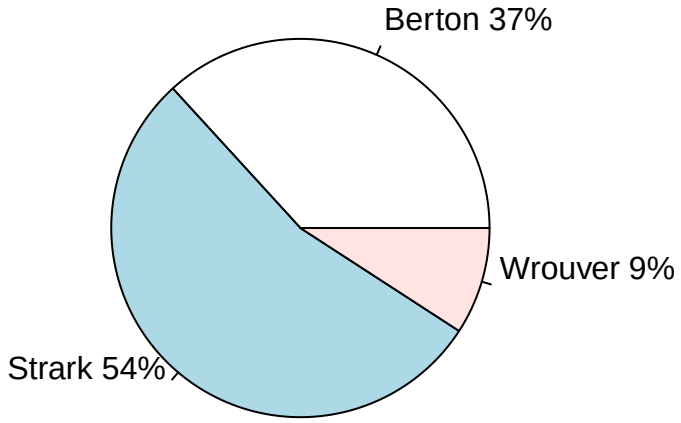


Figure 3: Distribution of Location

[**Interpretation**]

Figure 3 shows that 54% of the drivers are from Strark, 37% are from Berton and only 9% are from Wrouver.

[**Implication**]

More information about the cities is needed.

Summary

The most popular signup source/device is mobile devices including IOS and Andriod, which takes up 57% of the whole signup number. Then second popular source is websites from Mac or Windows, which takes up to 23%. 44% of new drivers signed up through paid promotion channel, 32% though referral, 25% though organic. Half of the drivers come from Strark, 37% from Berton and 9% from Wrouver.

Vehicle Registration/Information

Vehicle Condition

How old are the cars?

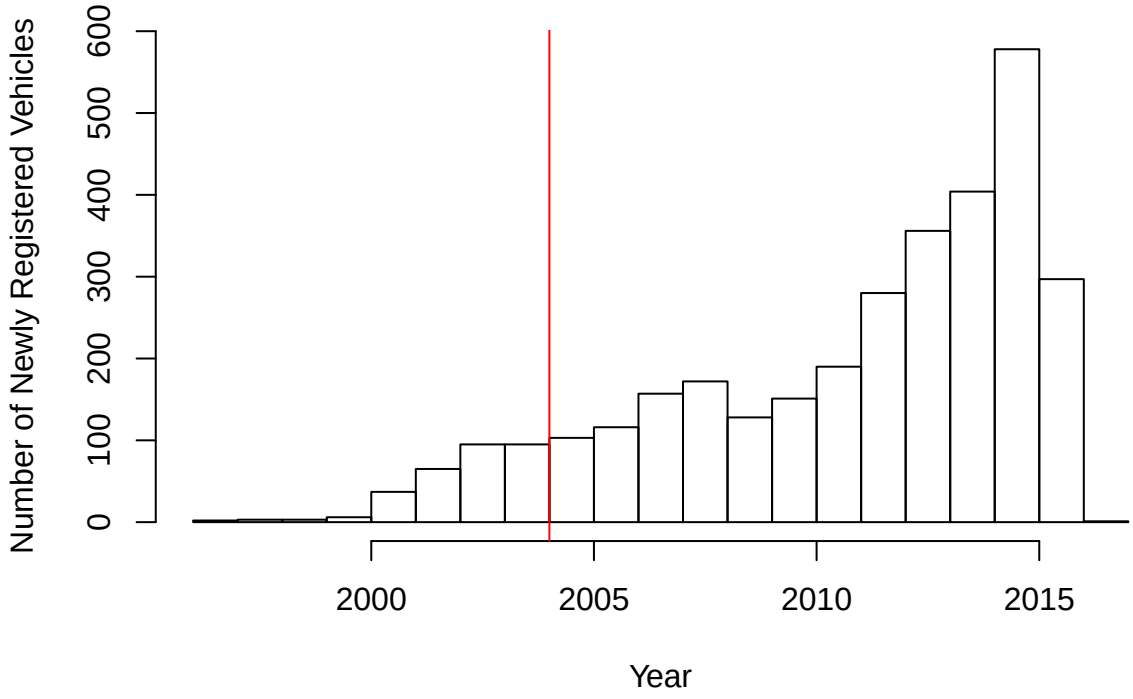


Figure 4: Distribution of Age of Vehicles

[Interpretation]

Figure 4 shows that most vehicles are made between 2011 and 2016.The oldest vehicles are over 15 years old while the newest are made in 2016. The histogram above is heavily skewed to the right side, which indicates that the mojority of vehicles are new.

[Implication]

In terms of years of a vehicle, the condition of most vehicles is satisfying.

```
# Add new features: complete, new
# 4: new
# 3: semi-new
# 2: semi-old
# 1: old
# 0: not acceptable
uber_v$new[which(!is.na(uber_v$vehicle_year))] <- "0" # do not meet Uber's requirements
uber_v$new[which(uber_v$vehicle_year >= 2004 &
```

```
                    uber_v$vehicle_year <= 2007)] <- "1" # 10-13 years
uber_v$new[which(uber_v$vehicle_year > 2007 &
                    uber_v$vehicle_year <= 2010)] <- "2" # 7-9 years
uber_v$new[which(uber_v$vehicle_year > 2010 &
                    uber_v$vehicle_year <= 2013)] <- "3" # 4-6 years
uber_v$new[which(uber_v$vehicle_year > 2013)] <- "4" # 0-3 years
table(uber_v$new)
```

```
##
##    0    1    2    3    4
##  211  471  451  826 1280
```

```
uber_v.comp$new[which(!is.na(uber_v.comp$vehicle_year))] <- "0" # do not meet Uber's requirements
uber_v.comp$new[which(uber_v.comp$vehicle_year >= 2004 &
                    uber_v.comp$vehicle_year <= 2007)] <- "1" # 10-13 years
uber_v.comp$new[which(uber_v.comp$vehicle_year > 2007 &
                    uber_v.comp$vehicle_year <= 2010)] <- "2" # 7-9 years
uber_v.comp$new[which(uber_v.comp$vehicle_year > 2010 &
                    uber_v.comp$vehicle_year <= 2013)] <- "3" # 4-6 years
uber_v.comp$new[which(uber_v.comp$vehicle_year >2013)] <- "4" # 0-3 years
table(uber_v.comp$new)
```

```
##
##   0   1   2   3   4
##  75 201 201 380 600
```

```
uber_v.not.comp$new[which(!is.na(uber_v.not.comp$vehicle_year))] <- "0" # do not meet Uber's requiremen
uber_v.not.comp$new[which(uber_v.not.comp$vehicle_year >= 2004 &
                    uber_v.not.comp$vehicle_year <= 2007)] <- "1" # 10-13 years
uber_v.not.comp$new[which(uber_v.not.comp$vehicle_year > 2007 &
                    uber_v.not.comp$vehicle_year <= 2010)] <- "2" # 7-9 years
uber_v.not.comp$new[which(uber_v.not.comp$vehicle_year > 2010 &
                    uber_v.not.comp$vehicle_year <= 2013)] <- "3" # 4-6 years
uber_v.not.comp$new[which(uber_v.not.comp$vehicle_year >2013)] <- "4" # 0-3 years
table(uber_v.not.comp$new)
```

```
##
##   0   1   2   3   4
## 136 270 250 446 680
```

```
uber$new <- NA
uber$new[which(!is.na(uber$vehicle_year))] <- "0" # do not meet Uber's requirements
uber$new[which(uber$vehicle_year >= 2004 &
                    uber$vehicle_year <= 2007)] <- "1" # 10-13 years
uber$new[which(uber$vehicle_year > 2007 &
                    uber$vehicle_year <= 2010)] <- "2" # 7-9 years
uber$new[which(uber$vehicle_year > 2010 &
                    uber$vehicle_year <= 2013)] <- "3" # 4-6 years
uber$new[which(uber$vehicle_year >2013)] <- "4" # 0-3 years
table(uber$new)
```

```
##
##    0    1    2    3    4
##  838 2080 1817 3275 5213
```

[Interpretation]

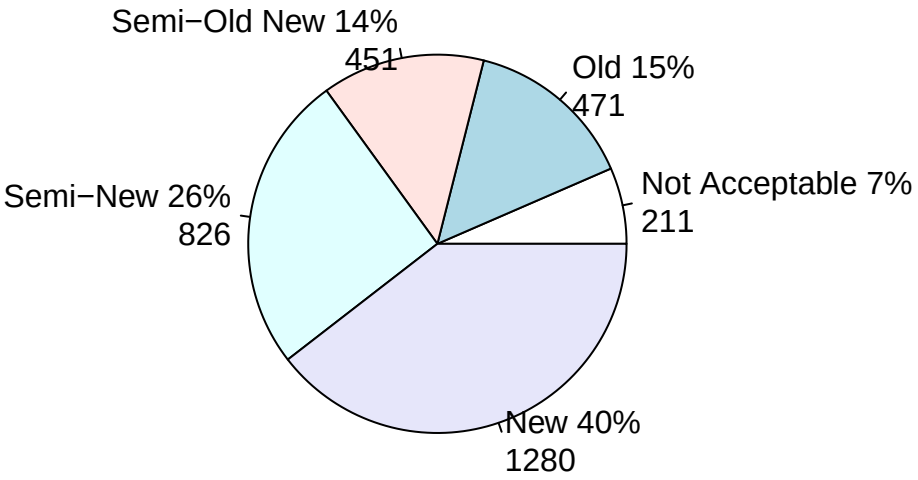


Figure 5: Distribution of Car Conditions

The cars are binned into 5 categories according to the years. 0-3 years are new, 4-6 years are semi-new, 7-9 years are semi-old, 10-13 years are old, and since using vehicles over 13 years for Uber would break the law in most states [1], they are considered unacceptable. Figure 5 shows that 40% of the vehicles are new, 26% are semi-new, 14% are semi-old and 15% are old. However, there are still 17% unacceptable vehicles registered.

[**Implication**]

66% of vehicles can be considered as in good condition, which is good news to passengers. However, the life expectancy or longevity of a Uber vehicle should be shorter than family cars considering the frequent usage. Therefore, for the vehicles over 6 years, which take up to 34% of the total number, Uber should provide more careful inspection. And the 7% of cars over 13 years should not even be added to the platform.

Vehicle Makes & Models

Which makes are most popular among drivers?

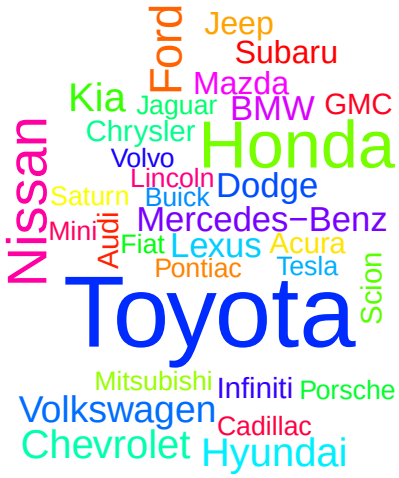


Figure 6: Wordcloud of Vehicle Makes

##

[1] https://rideshareapps.com/uber-vehicle-requirements-for-2019/

```
##     Toyota      Honda     Nissan       Ford    Hyundai  Chevrolet
##        788        448        315        215        166        160
##        Kia Volkswagen      Lexus        BMW
##        143        120         94         93
```

[Interpretation]

Figure 6 shows that the three top 10 brands are Toyota, Honda, Nissan, Ford, Hyundai, Chevrolet, Kia, Volkswagen, Lexus, BMW.

[Implication]

Interestingly, the 3 most popular makes are all from Japan, which are famous for economy cars. Lexus and BMW are the only 2 non-economy car makes.

Which models are most popular among drivers?

Figure 7: Wordcloud of Vehicle Models

```
##
##   Camry   Civic Corolla  Accord Prius V  Altima   Prius  Sentra  Fusion
##     188     166     152     143     136     103      85      69      65
## Elantra
##      62
```

[Interpretation]

Figure 7 shows that the 10 most popular car models are Camry, Civic, Corolla, Accord, Prius V, Altima, Prius, Sentra, Fusion, Elantra.

[Implication]

These cars are not only popular in Uber, but also among college students and can be seen everywhere, which indicates that cost-effectiveness and practicality are very essential for present and future Uber drivers.

Summary

Most cars are less than 6 years old. In terms of years, 40% vehicles can be considered as new, and 26% are semi-new. There are still 7% that are made before 2004, hence these vehicles should not be approved during registration. As to car makes and models, Japanese brands are really popular. The top 3 brands are all from Japan, which are Toyota, Honda and Nissan, which are all famous for economy and compact models. The top three models are Camry, Civic and Corolla, which are all economy cars.

First Trip Completion

Drivers

Drivers from which signup sources are most likely to finish their first ride?

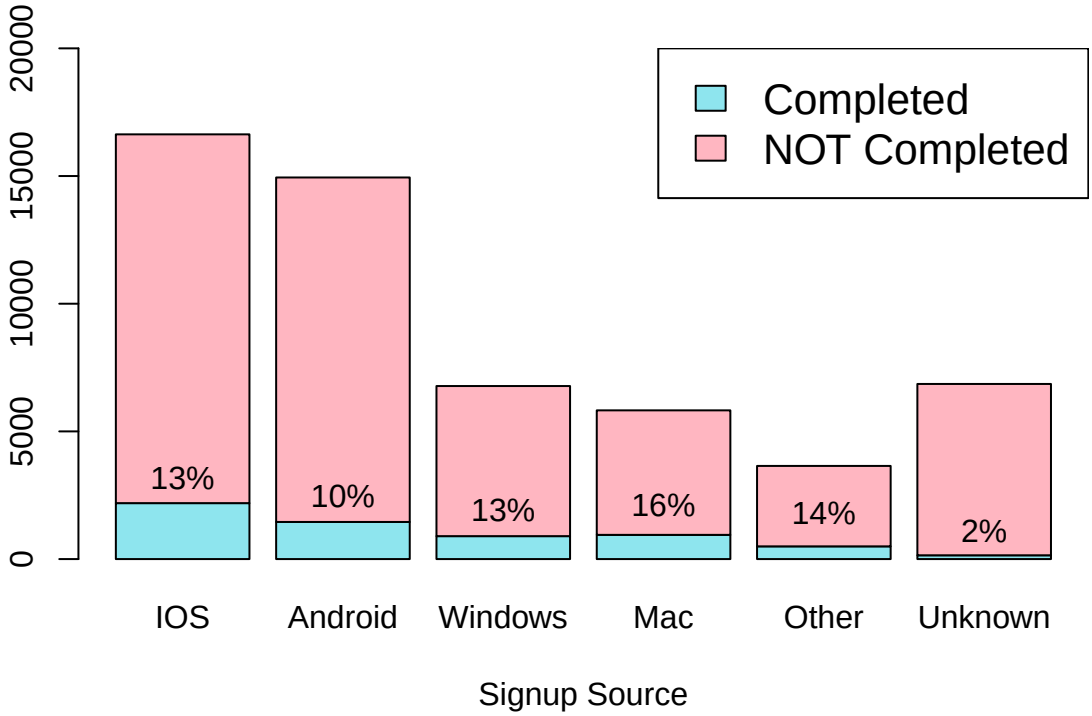


Figure 8: First Trip Compeletion Rate vs. Signup Source

[**Interpretation**]

Figure 8 shows that the completion rate is the highest at PC sources. That of Mac users is 16%, and of Windows users is 13%. For mobile sources, the conversion rates are slightly lower, which is 12% for Andriod users, and 13% for IOS users. Drivers signed up from other sources also have a relatively high conversion rate of 14%.

[**Implication**]

Even though the completion rate of mobile signup drivers are slightly lower than that of PC signup drivers, the difference in actual numbers can be big. Since IOS and Andriod take up 57% of the total users, the number of drivers who completed the first trip from these two sources is way larger than that of drivers signed up from PC.

The possible reason for the slightly higher completion rate for PC uses is that they signed up instantly when they finished researching about Uber with PC. Therefore, there is a great possibility to transfer users with this potential to sign up on mobile devices.

Drivers from which signup channels are most likely to finish their first ride?

[**Interpretation**]

Figure 9 shows that 9% percent of organic drivers , 6% of paid drivers and 20% of refered drivers finished their first trip.

[**Implication**]

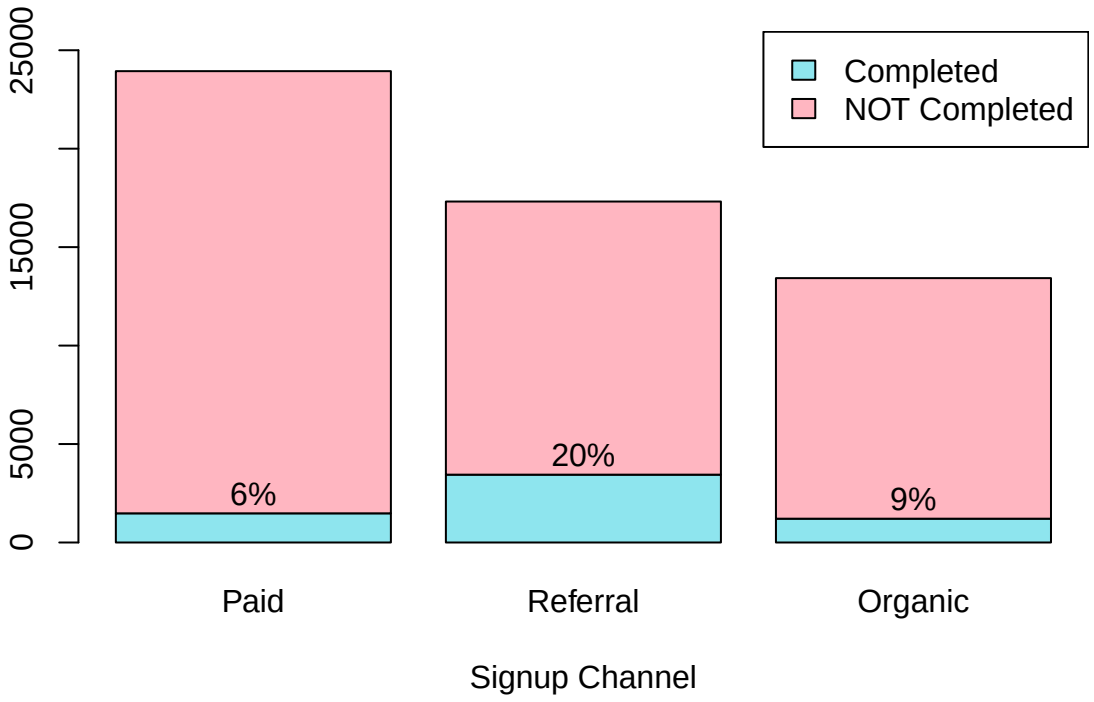


Figure 9: First Trip Compeletion Rate vs. Signup Channel

Drivers signed up through referrals has the highest value in both population and percentage of drivers who completed their first trip. In contrast, paid drivers has the highest number of signup but lowest number of first trip completion. Since signup bonus is important, the complany may want to invest more on referral bonus rather than paid promotions.

Drivers from which city have the highest first drive completion rate?

[**Interpretation**]

Figure 10 shows that Berton drivers have a completion rate of 12%, and Strark users have 11%, Wrouver drivers have 9%.

[**Implication**]

The completion rates for 3 cities are very close. Therefore, location may not play an important role when determine the completion rate.

Vehicles

Drivers with which vehicle conditions are most likely to finish the first trip?

[**Interpretation**]

Figure 11 shows that except unacceptable cars, the completion rates are very close. Among them, the newest have 47%, the semi-new have 46%, the semi-old have 45%, the oldest have 43%. The completion rate for unacceptable vehicles are 36%.

[**Implication**]

Suprisingly, this figure tells us that once the information of vehicle was complete, there are almost half the chance that the driver would complete their first trip. The reason may be that this group of drivers might be

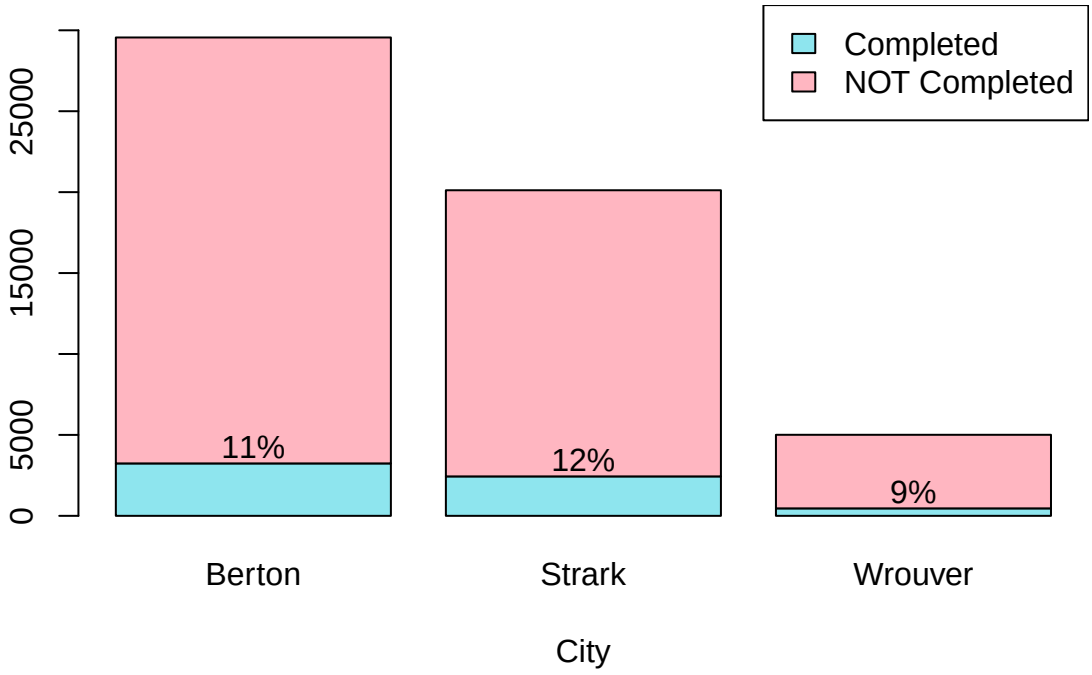


Figure 10: First Trip Completion Rate vs. Driver's Location

more determined to take a chance in Uber. But those unacceptable vehicles should not have a 36% completion rate.

Drivers with which kind of car are most likely to finish their first trip?

```
# Reference: https://www.quora.com/At-what-price-ranges-is-a-
#            car-considered-an-entry-level-mid-tier-and-luxury
# Price reference: https://www.truecar.com/prices-new/
# Price reference: https://www.autoblog.com/
# C: Entry, around $12000.
# B: Mid level, low $20's to mid $30's, possibly lower with incentives.
# A: Near luxury, low $40s to mid $50s.
# S: Luxury, high 40's to infinity.
uber_v$tier[which(!is.na(uber_v$vehicle_make))] <- "C"
list_S <- c("Porsche", "Tesla", "Maserati", "Bentley")
list_A <- c("Cadillac", "Mercedes-Benz", "Infiniti", "Lexus", "Audi", "BMW", "Volvo",
            "Lincoln", "Land Rover", "Jaguar")
list_B <- c("Dodge", "Acura", "GMC", "Jeep", "Subaru", "Buick", "Mazda", "Chrysler",
            "Hummer", "Mercury", "Mini", "Saab")
uber_v$tier[uber_v$vehicle_make %in% list_B] <- "B"
uber_v$tier[uber_v$vehicle_make %in% list_A] <- "A"
uber_v$tier[uber_v$vehicle_make %in% list_S] <- "S"
table(uber_v$tier)
```

```
##
##     A     B     C     S
##   393   392 12339     6
```

```
uber_v.comp$tier[which(!is.na(uber_v.comp$vehicle_make))] <- "C"
uber_v.comp$tier[uber_v.comp$vehicle_make %in% list_B] <- "B"
```

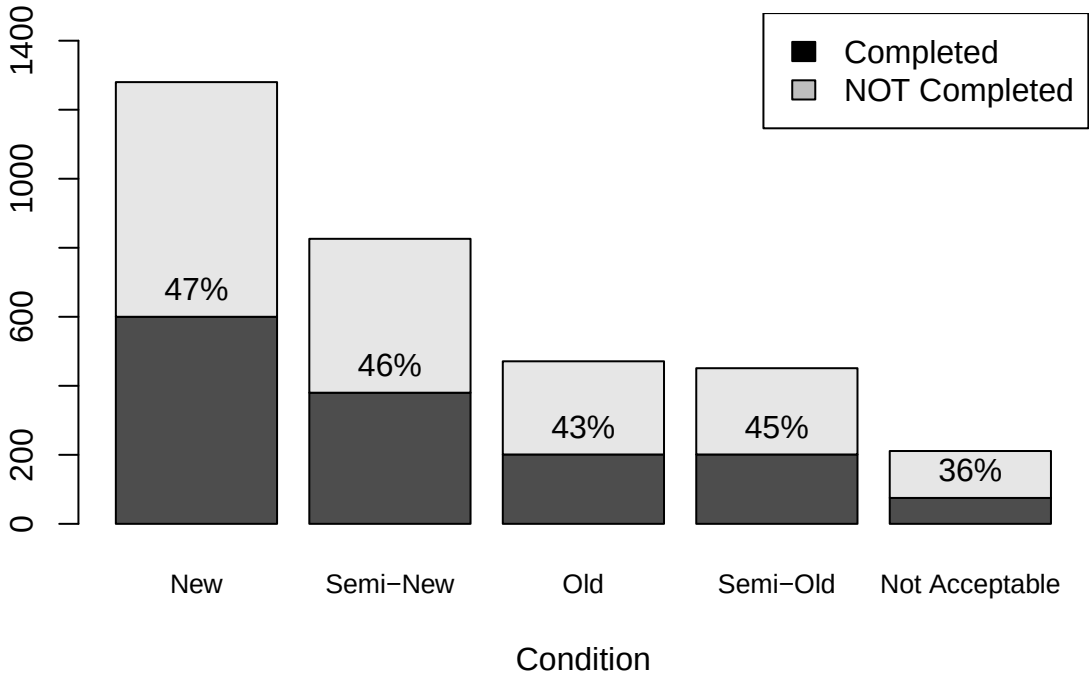


Figure 11: First Trip Completion Rate vs. Vehicle Condition

```
uber_v.comp$tier[uber_v.comp$vehicle_make %in% list_A] <- "A"
uber_v.comp$tier[uber_v.comp$vehicle_make %in% list_S] <- "S"
table(uber_v.comp$tier)
```

```
##
##    A    B    C    S
##  174  171 1184    2
```

```
uber_v.not.comp$tier[which(!is.na(uber_v.not.comp$vehicle_make))] <- "C"
uber_v.not.comp$tier[uber_v.not.comp$vehicle_make %in% list_B] <- "B"
uber_v.not.comp$tier[uber_v.not.comp$vehicle_make %in% list_A] <- "A"
uber_v.not.comp$tier[uber_v.not.comp$vehicle_make %in% list_S] <- "S"
table(uber_v.not.comp$tier)
```

```
##
##     A     B     C     S
##   219   221 11155     4
```

```
uber$tier[which(!is.na(uber$vehicle_make))] <- "C"
uber$tier[uber$vehicle_make %in% list_B] <- "B"
uber$tier[uber$vehicle_make %in% list_A] <- "A"
uber$tier[uber$vehicle_make %in% list_S] <- "S"
table(uber$tier)
```

```
##
##     A     B     C     S
##  1552  1679 51423    27
```

[Interpretation]

Figure 12 shows that the completion rate of Tier A and B cars are both 44%, for S the number is 33%, while for C the number is 10%.

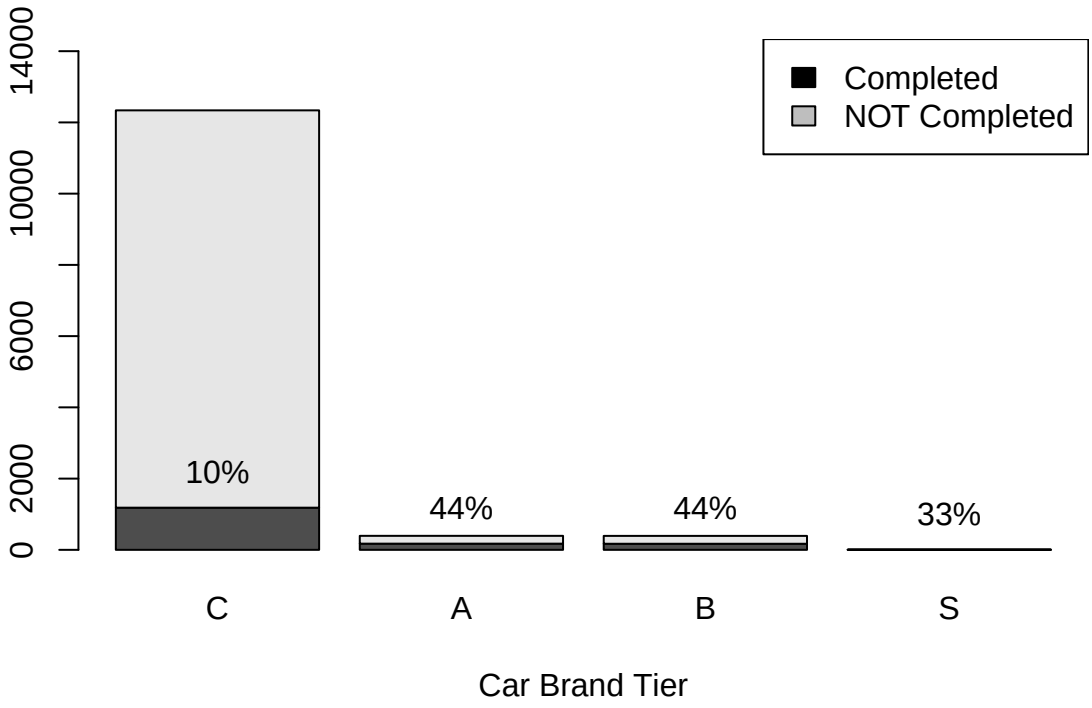


Figure 12: First Trip Completion Rate vs. Car Maker Tiers

[**Implication**]

Even though the completion rate of Tier C drivers is the lowest, and way lower that those of other tiers, it still has the highest actual number because most vehicles belong to this tier. It is worth trying to increase the completion rate with Tier C drivers.

Trend and Intervals Between Events

Trend of Daily Signup Drivers and Completion Rate

[**Interpretation**]

Figure 13 shows that he number of signup users fluctuate thourgh the whole month from 300 to 600 with an average of 437 new users everyday.

[**Implication**]

Suprisingly, no matter how the numbers of the signup change, the numbers of drivers who finished the first trip do not vary much and are stable around 60 each day.

Intervals Between Signup And First Trip Completion

[**Interpretation**]

Figure 14 shows that intervals between signup and first trip for most users are between 2-7 days. The longest interval can be up to 30 days.

[**Implication**]

Most drivers finished their first trip within a week of signing up.

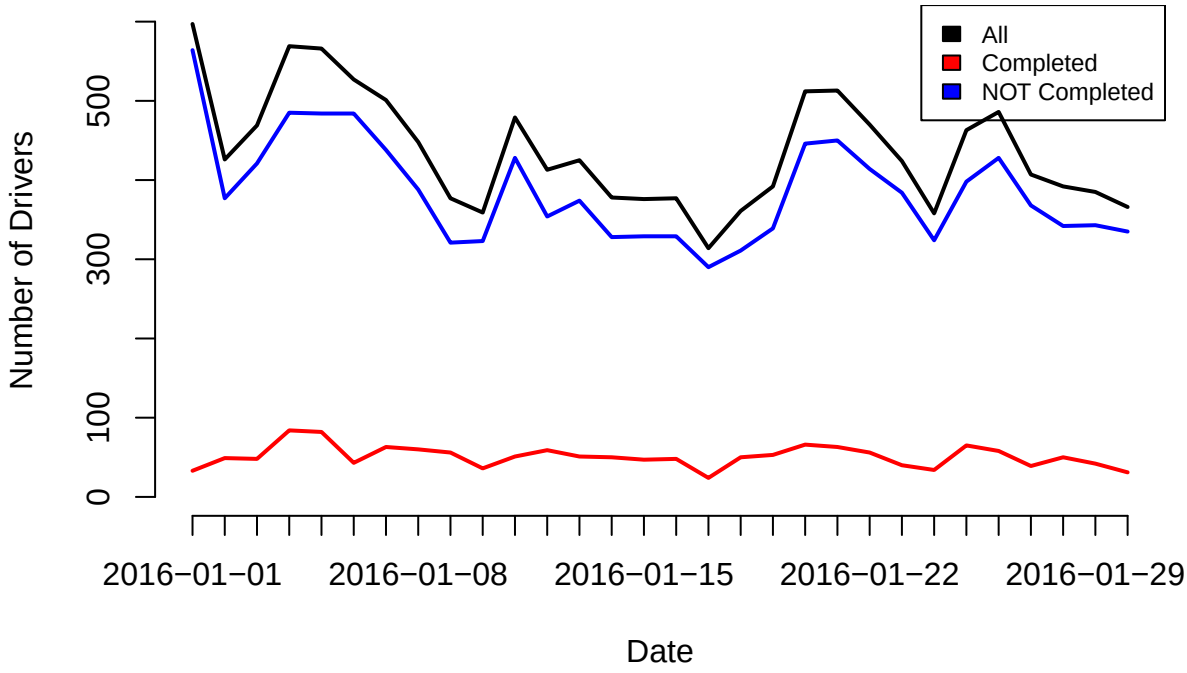


Figure 13: Number of Signup Drivers vs. Time

Summary

The completion rates of all signup source excepte unknown are between 10%-16%, which are close. The interesting part comes from the completion rate of each signup channel: even though most customers signed up though paid promotions, it has the lowest completion rate of only 6%; on the contrary, 32% drivers signed up thourgh referral channel, 20% of them finished their first trips. In terms of car conditions, the completion rate are between 43%-47% except unacceptable cars. Vehicles from brand tier A and B, which are determined by the average price of models of the brand, have the highest completion rate of 44%. Tier C drivers has the highest actual number of complete trips, but the completion rate is only 10%.

Also, the daily completion number is about 60 throughout the whole signup period (the whole Janurary). Most drivers would finish their first trip within a week if they had decided to do so.

Pre-Analysis Summary

According to the pre-analysis section, the most popular signup source is mobile devices (IOS and Andriod) with a total percentage of 57%, while the second most popular source, PC, only takes up 23%. The first trip completion rate is similar among those sources, between 10% to 16%. The most popular signup channel is paid promotions, which accounts for 44%. In comparison, organic and referral channels have slightly lower percentages, which are 25% and 32%, respectively. However, referral channel has the highest completion rate of 20%, comparing to 9% for organic channel and 6% for paid channel. And also, 54% of the drivers are from Strark, 37% are from Berton and 9% are from Wrouver. The completion rates of cities are all between 9%-12%.

Moreover, the age of vehicles are mostly between 0-6 years. In terms of age, 40% vehicles can be considered as new, and 26% can be considered as semi-new. However, in most states, vehicles made before 2004 cannot be used for Uber, hence there are still 7% of unacceptable vehicles. As to the completion rate, except for the unacceptable vehicles, the rates are all very close, varying from 43% to 47%. The top 3 popular makes are Toyota, Honda and Nissan, which are all famous for economy vehicles. The top 3 popular models are Camry, Civic and Crolla, belonging to Toyota, Honda and Toyota, respectively.

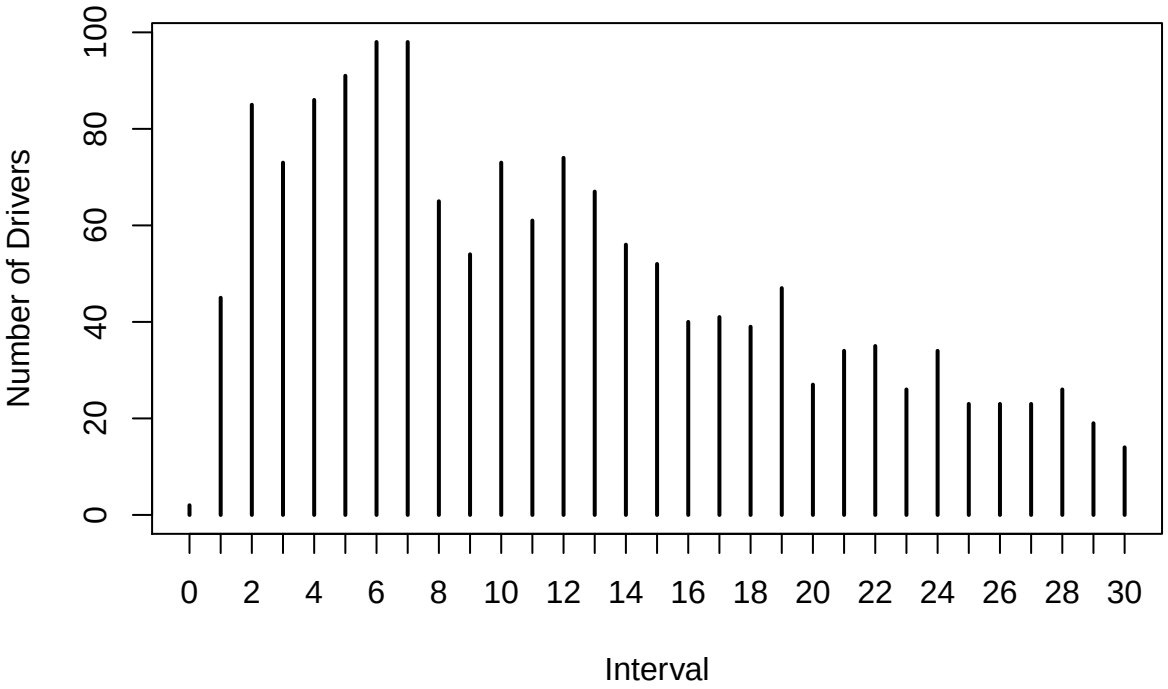


Figure 14: Number of Drivers vs. Intervals Between Signup And First Trip Completion

Also, the completion rates are close in terms of signup source, city, and vehicle conditions. The observable differences come in sighup channels and car brand tiers. 44% of drivers chose to sign up through paid channel but only 6% of then finished the first trip; however, 32% drivers who signed uo through referral channel have a completion rate of 20%. Similarly, vehicles from tier A and B only take up 6% of the total number, but both 44% completion rates. 94% of tier C drivers has a completion rate of 10%.

The daily signup number flactuates between 300 to 600, but the number of first trip completion are more stable around 60. Also, first trips usually happen within a week of signup.

Analysis

Correlation Analysis

Signup Channel vs. Car Brand Tiers

```
tbl <- table(uber_v$signup_channel, uber_v$tier)
tbl
```

```
##
##                A    B    C    S
##   Organic    101   88 3105    1
##   Paid       117  120 5474    1
##   Referral   175  184 3760    4
```

```
chisq.test(tbl, correct=F)
```

```
##
##  Pearson's Chi-squared test
```

```
##
## data:  tbl
## X-squared = 93.918, df = 6, p-value < 2.2e-16
```

[**Interpretation**]

Chi-Square test is used here to analyze the correlation between signup channel and car brand tiers. The p-value is close to 0.

[**Implcation**]

The null hypothesis is that groups of drivers categorized by signup channels and car brand tiers are independent. Since the p-value is less than the .05 significance level, we can reject the null hypothesis and say that these groups are correlated.

Interval Study

Signup Channel vs. Interval

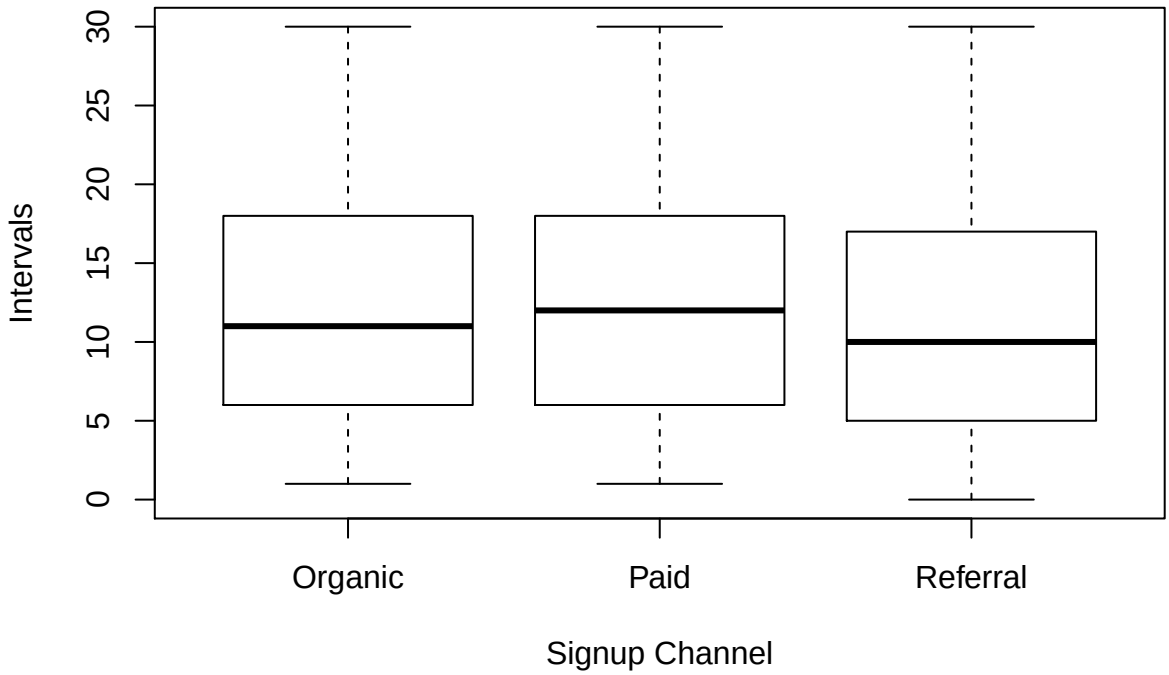


Figure 15: Boxplot of Signup Channel vs. Interval

[**Interpretation**]

Figure 15 the first and third quartiles of orgnic and paid channels are very close, which are 4 and 18, respectively. The averages are close as well, around 12, even though that of paid channel is slightly higher. Referred drivers have the lowest quartiles and average, and are close to those of the other two channels.

[**Implication**]

Drivers referred to Uber have the relatively shortest intervals, 1 or 2 days earlier than the other methods.

Car Tier vs. Interval

[**Interpretation**]

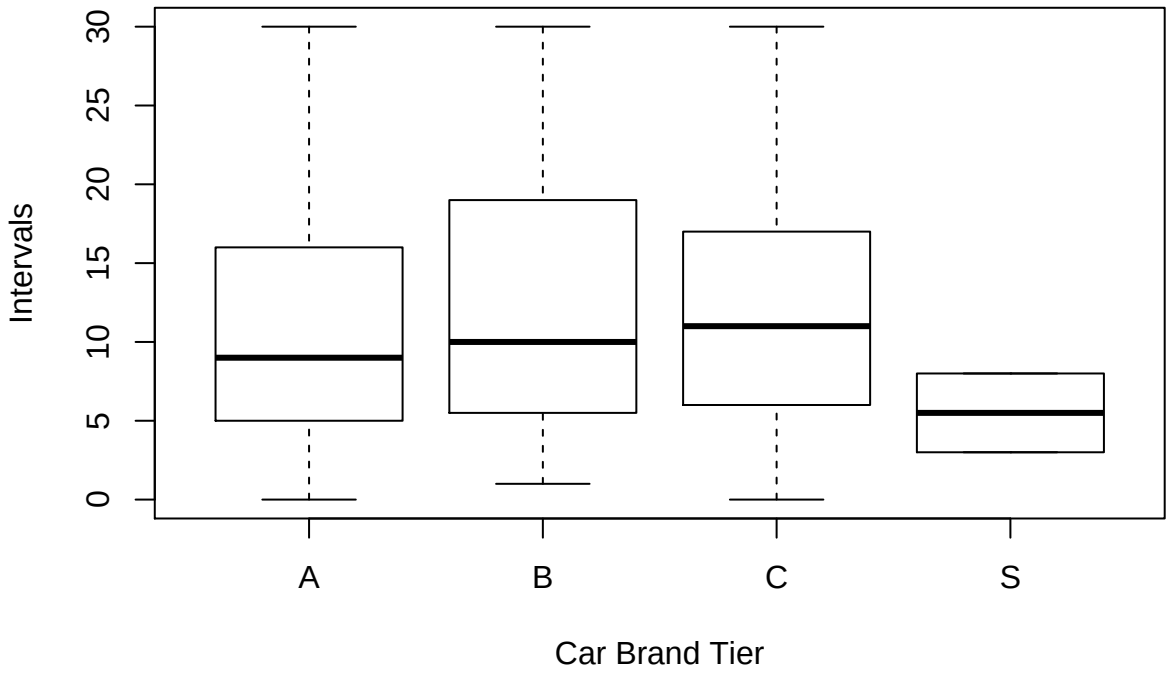


Figure 16: Boxplot of Car Brand Tier vs. Interval

Figure 16 shows that the drivers with tier S vehicles have the shortest average intervals about 7 days. Tier A, B and C drivers all have higher average intervals as well as quartiles. Their averages are around 11 days, and the interquartile range is about 10-12 days.

[**Implication**]

The sample size for tier S driver is 2, so that it may not represent the real situation. Tier C drivers has the longest average intervals comparing to A and B, however the difference is not large.

All Variables

```
# Create a variable for intervals in dataset Uber
uber$time.diff <- uber$first_completed_date - uber$signup_date
```

```
uber$signup_date.d <- (as.POSIXct(uber$signup_date) -
                         as.POSIXct(as.Date("2016/01/01")))/86400
uber$vehicle_added_date.d <- (as.POSIXct(uber$vehicle_added_date) -
                                as.POSIXct(as.Date("2016/01/01")))/86400
uber$bgc_date.d <- (as.POSIXct(uber$bgc_date) -
                      as.POSIXct(as.Date("2016/01/01")))/86400

uber$signup_v <- as.numeric(uber$vehicle_added_date.d - uber$signup_date.d)
uber$signup_bgc <- as.numeric(uber$bgc_date.d - - uber$signup_date.d)
uber$v_bgc <- as.numeric(uber$bgc_date.d - uber$vehicle_added_date.d)

uber$tier <- as.factor(uber$tier)
uber$new <- as.factor(uber$new)
```

```
# Create a new dataset with categories in numbers for correlation matrix
uber.num <- uber
```

```r
uber.num$city_name <- as.numeric(factor(uber$city_name)) - 1
uber.num$signup_os <- as.numeric(factor(uber$signup_os)) - 1
uber.num$signup_channel <- as.numeric(factor(uber$signup_channel)) - 1
uber.num$tier <- as.numeric(factor(uber$tier)) - 1
uber.num$new <- as.numeric(uber$new)
uber.num$signup_date.d <- as.numeric(uber$signup_date.d)
uber.num$bgc_date.d <- as.numeric(uber$bgc_date.d)
uber.num$vehicle_added_date.d <- as.numeric(uber$vehicle_added_date.d)
```

```r
M <- cor(uber.num[c('city_name', 'signup_os', 'signup_channel', 'tier', 'new',
               'signup_date.d', 'bgc_date.d', 'vehicle_added_date.d',
               'signup_v', 'signup_bgc', 'v_bgc')],
         use="complete.obs")
M
```

```
##                         city_name    signup_os signup_channel
## city_name             1.000000000 -0.002442366    -0.05031430
## signup_os            -0.002442366  1.000000000    -0.14799655
## signup_channel       -0.050314304 -0.147996547     1.00000000
## tier                  0.012519825 -0.014777778     0.01867590
## new                   0.065260262 -0.013290220     0.03013786
## signup_date.d         0.025943424 -0.013673246    -0.06565407
## bgc_date.d            0.020178662 -0.043392912    -0.08323698
## vehicle_added_date.d  0.012504562 -0.051479473    -0.08917715
## signup_v             -0.002355856 -0.047360366    -0.05632914
## signup_bgc            0.025063708 -0.034492101    -0.08432936
## v_bgc                 0.005828810  0.027326851     0.03690237
##                                tier          new signup_date.d   bgc_date.d
## city_name             0.0125198246  0.065260262   0.025943424  0.020178662
## signup_os            -0.0147777776 -0.013290220  -0.013673246 -0.043392912
## signup_channel        0.0186759047  0.030137863  -0.065654067 -0.083236980
## tier                  1.0000000000  0.156218487   0.002428565 -0.002472277
## new                   0.1562184874  1.000000000   0.017532603 -0.009436573
## signup_date.d         0.0024285648  0.017532603   1.000000000  0.610643705
## bgc_date.d           -0.0024722774 -0.009436573   0.610643705  1.000000000
## vehicle_added_date.d -0.0107242729 -0.015017943   0.412780666  0.787139133
## signup_v             -0.0130992222 -0.027001805  -0.165745130  0.478124219
## signup_bgc           -0.0004872153  0.001947569   0.857494528  0.931053521
## v_bgc                 0.0141872735  0.012139821   0.120674608  0.017413747
##                      vehicle_added_date.d     signup_v    signup_bgc
## city_name                     0.01250456 -0.002355856  0.0250637075
## signup_os                    -0.05147947 -0.047360366 -0.0344921008
## signup_channel               -0.08917715 -0.056329144 -0.0843293585
## tier                         -0.01072427 -0.013099222 -0.0004872153
## new                          -0.01501794 -0.027001805  0.0019475695
## signup_date.d                 0.41278067 -0.165745130  0.8574945283
## bgc_date.d                    0.78713913  0.478124219  0.9310535209
## vehicle_added_date.d          1.00000000  0.829816032  0.7015914957
## signup_v                      0.82981603  1.000000000  0.2342629160
## signup_bgc                    0.70159150  0.234262916  1.0000000000
## v_bgc                        -0.60297491 -0.726782721  0.0669162222
##                           v_bgc
## city_name            0.00582881
## signup_os            0.02732685
```

```
## signup_channel         0.03690237
## tier                   0.01418727
## new                    0.01213982
## signup_date.d          0.12067461
## bgc_date.d             0.01741375
## vehicle_added_date.d -0.60297491
## signup_v             -0.72678272
## signup_bgc             0.06691622
## v_bgc                  1.00000000
```

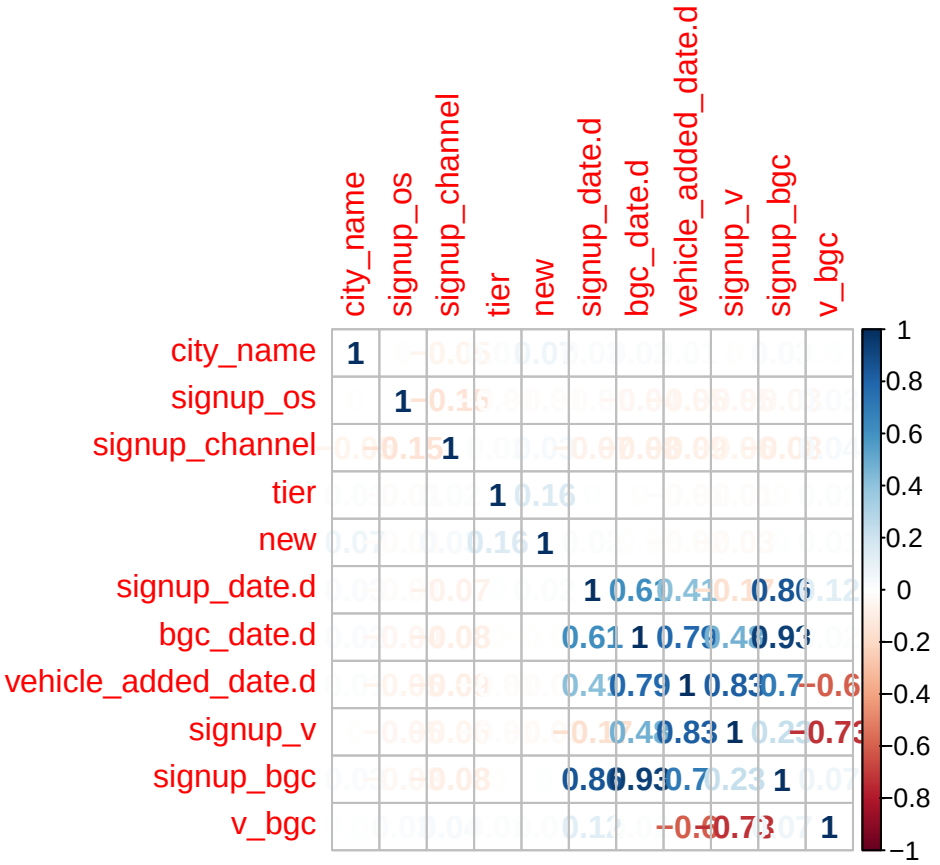


Figure 17: Correlation Map of All Variables

[Interpretation]

Figure 17 shows that the there are 2 noticeable correlations among time variables: signup date and background check date (0.61, moderate positive correlation), signup date and vehicle registration date (0.41, weak positive correlation), and background check date and vehicle registration date (0.79, strong positive correlation). Since there are many missing values in the dataset, the correlation matrix only used complete observations, which are the drivers that went through the signup to adding all the information of their vehicles.

After adding features representing the invertals between signup, background check and vehicle regitration, there are some more correlations: vehicle registration date and the intervals between vehicle registration date and background check (-0.6, moderate negative correlation), intervals between signup and vehicle registration and intervals between vehicle registration and background check (-0.73, strong negative correlation), vehicle registration date and the intervals between signup and background check (0.7, strong positive correlation), signup date and the intervals between signup date and background check date (0.86, strong positive correlation), background check date and the intervals between signup date and background check date (0.93, strong positive

correlation).

[Implication]

The result of the corrrelation plot shows the following things:// 1) Early vehicle registration date leads to longer intervals between vehicle registration date and background check.\ 2) The longer the intervals between signup and vehicle registration are, the shorter the intervals between vehicle registration and background check are.\ 3) Early vehicle regitration date leads to shorter intervals between signup and background check.\ 4) Early signup date leads to shorter intervals between signup date and background check date.\ 5) Early background check date leads to shorter intervals between signup date and background check date.\

Summary

The intervals between drivers from different signup channels do not vary much, and referral channel has the lowest average intervals. As to car brand tiers, tier C drivers has the longest average interval of about 11 days. If tier S is not taken into consideration duw to the small sample size of 2, tier A has the shortest average intervals of about 9 days. The correlation between intervals versus signup channel and car brand tier needs further analysis.

Prediction

First Trip Completion

```
## Split the whole dataset into training and test
set.seed(123)
smp_siz = floor(0.7*nrow(uber))
train_ind = sample(seq_len(nrow(uber)), size = smp_siz)
uber.train = uber[train_ind,] #creates the training dataset with row numbers stored in train_ind
uber.test= uber[-train_ind,]

uber.num <- na.roughfix(uber.num[c('city_name', 'signup_os', 'signup_channel', 'tier', 'new',
               'signup_date.d', 'bgc_date.d', 'vehicle_added_date.d',
               'signup_v', 'signup_bgc', 'v_bgc', 'complete')])
uber.num.train = uber.num[train_ind,]
uber.num.test= uber.num[-train_ind,]
```

Logistic Regression (With Dates)

```
glm.uber <- glm(complete ~ city_name + signup_os + signup_channel +
                  signup_date.d + vehicle_added_date.d + new,
                data = uber.train, family = "binomial")
summary(glm.uber)
```

```
##
## Call:
## glm(formula = complete ~ city_name + signup_os + signup_channel +
##     signup_date.d + vehicle_added_date.d + new, family = "binomial",
##     data = uber.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2880  -0.6313  -0.1134   0.7333   3.3816
##
```

Table 1: Odds Ratios of Logistic Regression Results (With Dates)

predictor	oddsratio	CI_low (2.5)	CI_high (97.5)	increment
city_nameStrark	0.875	0.779	0.983	Indicator variable
city_nameWrouver	0.669	0.542	0.827	Indicator variable
signup_osandroid web	2.699	1.975	3.708	Indicator variable
signup_osios web	2.715	1.998	3.711	Indicator variable
signup_osmac	3.797	2.742	5.289	Indicator variable
signup_osother	3.767	2.642	5.400	Indicator variable
signup_oswindows	3.789	2.727	5.294	Indicator variable
signup_channelPaid	1.033	0.884	1.206	Indicator variable
signup_channelReferral	1.905	1.654	2.195	Indicator variable
signup_date.d	1.152	1.141	1.163	Indicator variable
vehicle_added_date.d	0.853	0.847	0.859	Indicator variable
new1	1.533	1.187	1.982	Indicator variable
new2	1.431	1.102	1.859	Indicator variable
new3	1.601	1.257	2.040	Indicator variable
new4	1.842	1.459	2.328	Indicator variable

```
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)              0.151557   0.201268   0.753 0.451441
## city_nameStrark         -0.133215   0.059135  -2.253 0.024277 *
## city_nameWrouver        -0.401375   0.107547  -3.732 0.000190 ***
## signup_osandroid web     0.992711   0.160539   6.184 6.27e-10 ***
## signup_osios web         0.998642   0.157855   6.326 2.51e-10 ***
## signup_osmac             1.334177   0.167511   7.965 1.66e-15 ***
## signup_osother           1.326283   0.182302   7.275 3.46e-13 ***
## signup_oswindows         1.332001   0.169132   7.876 3.39e-15 ***
## signup_channelPaid       0.032302   0.079277   0.407 0.683674
## signup_channelReferral   0.644455   0.072295   8.914  < 2e-16 ***
## signup_date.d            0.141300   0.004712  29.986  < 2e-16 ***
## vehicle_added_date.d    -0.159257   0.003567 -44.648  < 2e-16 ***
## new1                     0.427377   0.130706   3.270 0.001076 **
## new2                     0.358222   0.133319   2.687 0.007211 **
## new3                     0.470788   0.123522   3.811 0.000138 ***
## new4                     0.611094   0.119086   5.132 2.87e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 12653.8  on 9221  degrees of freedom
## Residual deviance:  8086.4  on 9206  degrees of freedom
##   (29054 observations deleted due to missingness)
## AIC: 8118.4
##
## Number of Fisher Scoring iterations: 6
```

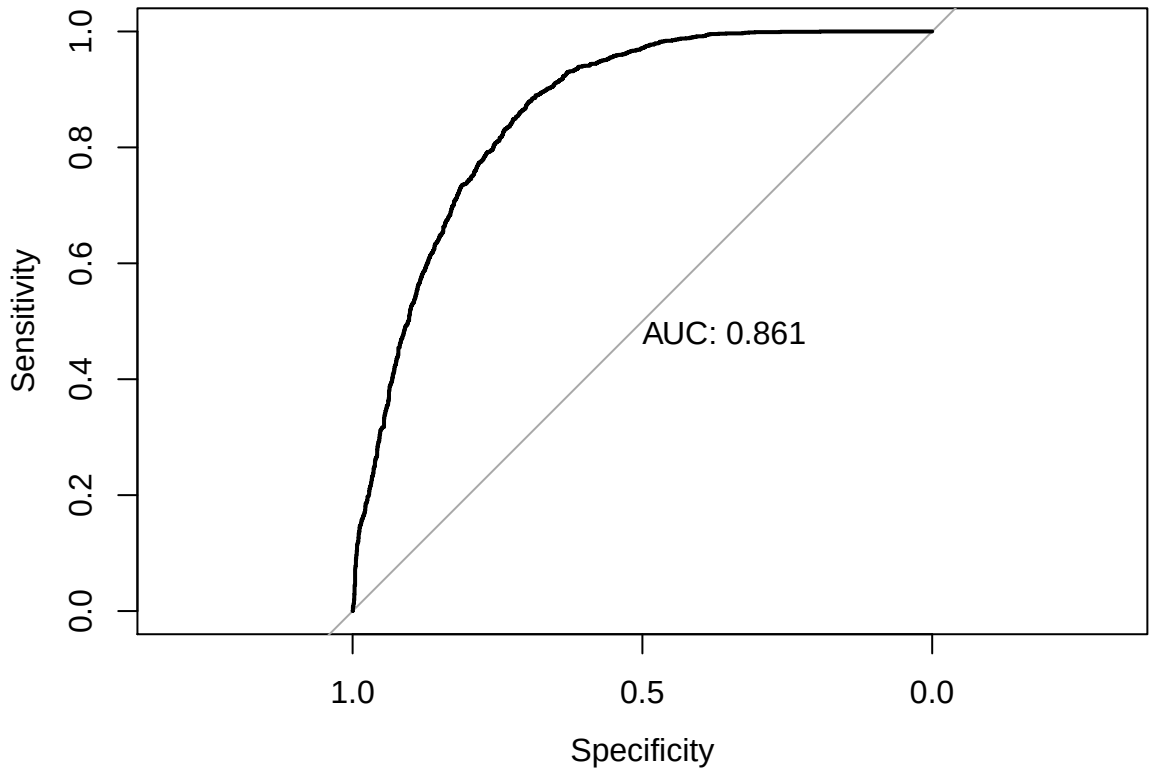


Figure 18: ROC of Logistic Regression for First Trip Completion (With Dates)

[Interpretation]

Baseline: The completion rate for drivers from Berton, sign up through unknow source and organic channel, and the vehicle condition is unacceptable.

Formula: `complete` ~ `city_name` (Berton) + `signup_os` (Unknown) + `signup_channel` (Organic) + `new` (Unacceptable) + `signup_date.d` + `vehicle_added_date.d`

The summary shows 6 important variables: city, signup channel, signup source, siangup date, vehicle added date,and vehicle condition. Increase in vehicle added date and change in the location compared to the baseline model can reduce the chances of finishing the first trip. Changes or increase in other variables compared to the baseline model can all increase the possibilities.

Figure 18 shows that the AUC score of logistic model is 0.861. 6 predictors are included in the model: city, signup source, signup channel, signup date, vehicle registration date and the vehicle condition. The variables that are not significant are removed from the model.

[Implication]

According to Table 1, comparing to the baseline model:

1) Compared to the baseline model with an unknown signup source, changing the signup source to any of the rest categories can easily double or even triple the chances.
2) Compared to the baseline model with drivers from Berton, if the location of the drivers are other cities, the chanced will be lower.
3) As long as the condition of vehicle is not acceptable, the chances will increase by 50%-80%.
4) Later vehicle registration date will decrease the possibility by 15% per day; however, later signup date will increase the chances by 15% per day.

Logistic Regression (With Intervals)

```
glm.uber.2 <- glm(complete ~ city_name + signup_os + signup_channel +
                  signup_v + signup_bgc + new ,
                data = uber.train, family = "binomial")
summary(glm.uber.2)
```

```
##
## Call:
## glm(formula = complete ~ city_name + signup_os + signup_channel +
##     signup_v + signup_bgc + new, family = "binomial", data = uber.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3028  -0.6281  -0.1071   0.7313   3.3841
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)              0.255026   0.203358   1.254 0.209814
## city_nameStrark         -0.133998   0.059691  -2.245 0.024778 *
## city_nameWrouver        -0.389324   0.108780  -3.579 0.000345 ***
## signup_osandroid web     0.936125   0.162365   5.766 8.14e-09 ***
## signup_osios web         0.953816   0.159734   5.971 2.35e-09 ***
## signup_osmac             1.312528   0.169624   7.738 1.01e-14 ***
## signup_osother           1.302294   0.184473   7.060 1.67e-12 ***
## signup_oswindows         1.284085   0.171055   7.507 6.06e-14 ***
## signup_channelPaid       0.029471   0.080101   0.368 0.712927
## signup_channelReferral   0.637990   0.072992   8.741  < 2e-16 ***
## signup_v                -0.154551   0.003633 -42.545  < 2e-16 ***
## signup_bgc              -0.009574   0.001576  -6.077 1.23e-09 ***
## new1                     0.437289   0.132103   3.310 0.000932 ***
## new2                     0.340523   0.134499   2.532 0.011348 *
## new3                     0.455385   0.124658   3.653 0.000259 ***
## new4                     0.611773   0.120263   5.087 3.64e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 12444.3  on 9043  degrees of freedom
## Residual deviance:  7945.2  on 9028  degrees of freedom
##   (29232 observations deleted due to missingness)
## AIC: 7977.2
##
## Number of Fisher Scoring iterations: 6
```

[**Interpretation**]

Baseline: The completion rate for drivers from Berton, sign up through unknow source and organic channel, and the vehicle condition is unacceptable.

Formula: `complete` ~ `city_name` (Berton) + `signup_os` (Unknown) + `signup_channel` (Organic) + `new` (Unacceptable) + `signup_v` + `signup_bgc`

The summary shows 6 important variables: city, signup channel, signup source, siangup-vehicle registration intervals, signup-background check intervals,and vehicle condition. Increase in siangup-vehicle registration intervals and signup-background check intervals can reduce the chances, and change in the location compared

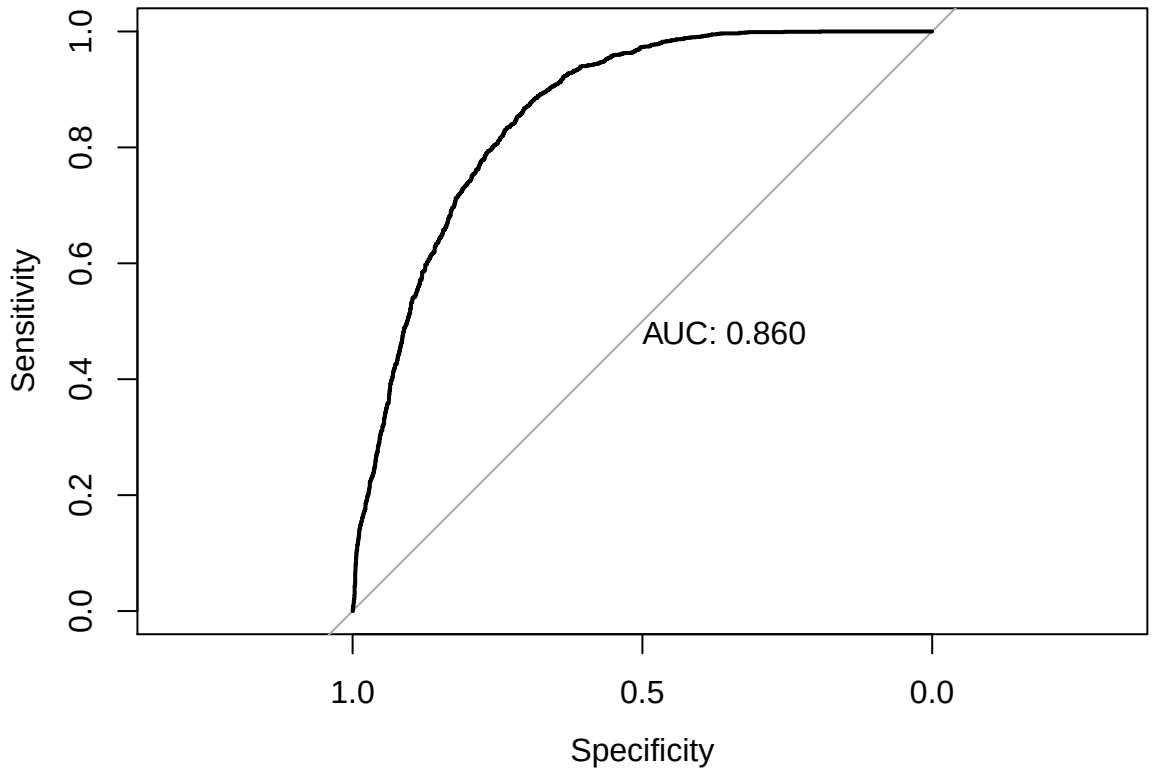


Figure 19: ROC of Logistic Regression for First Trip Completion (With Intervals)

to the baseline model can reduce the chances of finishing the first trip. Changes or increase in other variables can all increase the possibilities.

Figure 19 shows that the AUC is 0.86, 0.01 lower compared to that of the previous logistic regression model with dates.

[Implication]

Table 2 shows similar results as Table 1. In addition, when the signup-vehicle regitration intervals and signup background check intervals increase by 1, the possibility will reduce 15% or 1%, respectively.

The results are similar to the results with logistic regression model with dates. As to the 2 new features, it shows that the when the signup-vehicle registration intervals and signup-background check intervals increase by 1 day, the chances will decrease by 20%.

Summary - Logistic Regression

Compare to the baseline model, any increase in `city`, `vehicle_added_date.d` (in model with dates), `signup_v` (in model with intervals) and `signup_bgc` (in model with intervals) can reduce the chances of first trip completion. On the contrary, the increases in other variables may help increase probabilities. Among all the variables, change the signup source from unknow to any other sources can cause the greatest increase the possibilities.

Decision Tree (With Dates)

```
dt.uber <- rpart(complete ~ city_name + signup_os + signup_channel + new +
                   signup_date.d+ bgc_date.d + vehicle_added_date.d + tier,
                 data = uber.train, method = "class",
```

Table 2: Odds Ratios of Logistic Regression Results (With Intervals)

predictor	oddsratio	CI_low (2.5)	CI_high (97.5)	increment
city_nameStrark	0.875	0.778	0.983	Indicator variable
city_nameWrouver	0.678	0.547	0.839	Indicator variable
signup_osandroid web	2.550	1.860	3.516	Indicator variable
signup_osios web	2.596	1.903	3.561	Indicator variable
signup_osmac	3.716	2.671	5.196	Indicator variable
signup_osother	3.678	2.568	5.294	Indicator variable
signup_oswindows	3.611	2.589	5.065	Indicator variable
signup_channelPaid	1.030	0.880	1.205	Indicator variable
signup_channelReferral	1.893	1.641	2.184	Indicator variable
signup_v	0.857	0.851	0.863	1
signup_bgc	0.990	0.987	0.994	1
new1	1.549	1.195	2.007	Indicator variable
new2	1.406	1.080	1.830	Indicator variable
new3	1.577	1.235	2.014	Indicator variable
new4	1.844	1.457	2.334	Indicator variable

```
                control = rpart.control(cp = 0.005))
summary(dt.uber)
```

```
## Call:
## rpart(formula = complete ~ city_name + signup_os + signup_channel +
##     new + signup_date.d + bgc_date.d + vehicle_added_date.d +
##     tier, data = uber.train, method = "class", control = rpart.control(cp = 0.005))
##   n= 38276
##
##           CP nsplit rel error    xerror       xstd
## 1 0.03498233      0 1.0000000 1.0000000 0.01447222
## 2 0.01224971      2 0.9300353 0.9302709 0.01401911
## 3 0.00500000      4 0.9055359 0.9074205 0.01386541
##
## Variable importance
## vehicle_added_date.d                 tier           bgc_date.d
##                   45                   24                   23
##        signup_date.d
##                    8
##
## Node number 1: 38276 observations,    complexity param=0.03498233
##   predicted class=0  expected loss=0.110905  P(node) =1
##     class counts: 34031  4245
##    probabilities: 0.889 0.111
##   left son=2 (12848 obs) right son=3 (25428 obs)
##   Primary splits:
##       vehicle_added_date.d < 32.5 to the right, improve=1116.13700, (29054 missing)
##       tier                 splits as  RRLL,     improve= 429.59780, (0 missing)
##       bgc_date.d           < 29.5 to the right, improve= 333.24300, (15175 missing)
```

```
##       signup_channel       splits as  LLR,      improve= 275.76840, (0 missing)
##       signup_os            splits as  LRRRRR,   improve=  86.00242, (0 missing)
##   Surrogate splits:
##       bgc_date.d    < 28.5 to the right, agree=0.803, adj=0.491, (14057 split)
##       signup_date.d < 21.5 to the right, agree=0.664, adj=0.133, (14997 split)
##
## Node number 2: 12848 observations
##   predicted class=0  expected loss=0.04047323  P(node) =0.3356673
##     class counts: 12328   520
##    probabilities: 0.960 0.040
##
## Node number 3: 25428 observations,    complexity param=0.03498233
##   predicted class=0  expected loss=0.1464921  P(node) =0.6643327
##     class counts: 21703  3725
##    probabilities: 0.854 0.146
##   left son=6 (24065 obs) right son=7 (1363 obs)
##   Primary splits:
##       tier                 splits as  RRLR,     improve=616.0249, (0 missing)
##       signup_channel       splits as  LLR,      improve=280.1886, (0 missing)
##       signup_date.d        < 23.5 to the left,  improve=132.6802, (0 missing)
##       vehicle_added_date.d < 20.5 to the right, improve=125.7978, (19779 missing)
##       signup_os            splits as  LRRRRR,   improve=120.9599, (0 missing)
##
## Node number 6: 24065 observations
##   predicted class=0  expected loss=0.1202992  P(node) =0.628723
##     class counts: 21170  2895
##    probabilities: 0.880 0.120
##
## Node number 7: 1363 observations,    complexity param=0.01224971
##   predicted class=1  expected loss=0.3910492  P(node) =0.03560978
##     class counts:   533   830
##    probabilities: 0.391 0.609
##   left son=14 (863 obs) right son=15 (500 obs)
##   Primary splits:
##       vehicle_added_date.d < 16.5 to the right, improve=39.556600, (11 missing)
##       signup_os            splits as  LRRRRR,   improve=18.265720, (0 missing)
##       bgc_date.d           < 12.5 to the right, improve=13.256600, (33 missing)
##       signup_channel       splits as  LLR,      improve= 3.052765, (0 missing)
##       signup_date.d        < 16.5 to the left,  improve= 2.273050, (0 missing)
##   Surrogate splits:
##       bgc_date.d    < 14.5 to the right, agree=0.849, adj=0.586, (1 split)
##       signup_date.d < 9.5  to the right, agree=0.773, adj=0.377, (10 split)
##
## Node number 14: 863 observations,    complexity param=0.01224971
##   predicted class=1  expected loss=0.4797219  P(node) =0.02254677
##     class counts:   414   449
##    probabilities: 0.480 0.520
##   left son=28 (302 obs) right son=29 (561 obs)
##   Primary splits:
##       signup_date.d        < 12.5 to the left,  improve=34.417690, (0 missing)
##       signup_os            splits as  LLLRRR,   improve= 8.755807, (0 missing)
##       bgc_date.d           < 15.5 to the left,  improve= 8.443629, (17 missing)
##       vehicle_added_date.d < 26.5 to the right, improve= 6.909582, (4 missing)
##       new                  splits as  LRLRR,    improve= 3.556948, (0 missing)
```

```
##   Surrogate splits:
##       bgc_date.d < 15.5 to the left,  agree=0.804, adj=0.440, (0 split)
##       signup_os  splits as  LRRRRR,   agree=0.656, adj=0.017, (0 split)
##
## Node number 15: 500 observations
##   predicted class=1  expected loss=0.238  P(node) =0.01306302
##     class counts:   119   381
##    probabilities: 0.238 0.762
##
## Node number 28: 302 observations
##   predicted class=0  expected loss=0.3278146  P(node) =0.007890062
##     class counts:   203    99
##    probabilities: 0.672 0.328
##
## Node number 29: 561 observations
##   predicted class=1  expected loss=0.3761141  P(node) =0.0146567
##     class counts:   211   350
##    probabilities: 0.376 0.624
```

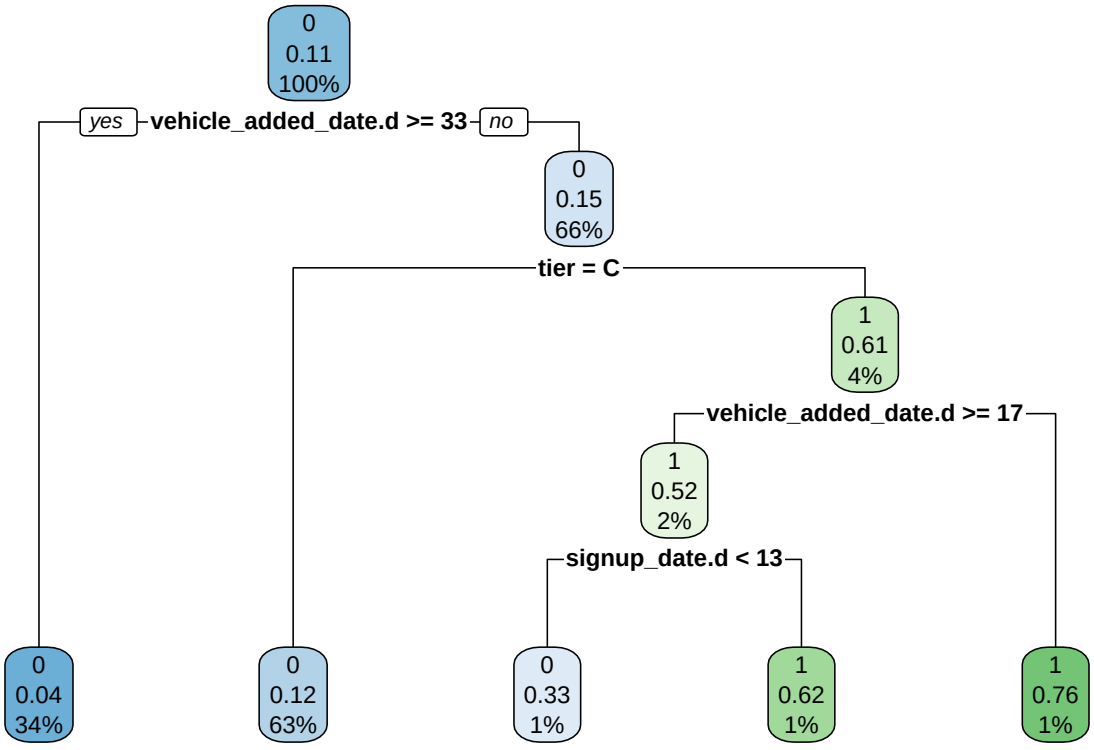


Figure 20: Decision Tree for First Trip Completion

[Interpretation]

Figure 20 shows the plot of the decision tree. 34% drivers added vehicles after Feburary 3, but only have the possibility of 4% to finish the first trip. Moreover, 63% drivers added their vehicle and having a vehicle that belongs Tier C, they share a completion chance of 12%. The highest possibility comes from drivers who added the vehicle before Jan 17, 2016, with a vehicle that were not Tier C. But only 1% of the drivers belong to this category.

The summary of the decision tree shows that the most important predictors are: vehicle added date, car brand tier, background check date and signup date. And the nodes are split according to these predictors.

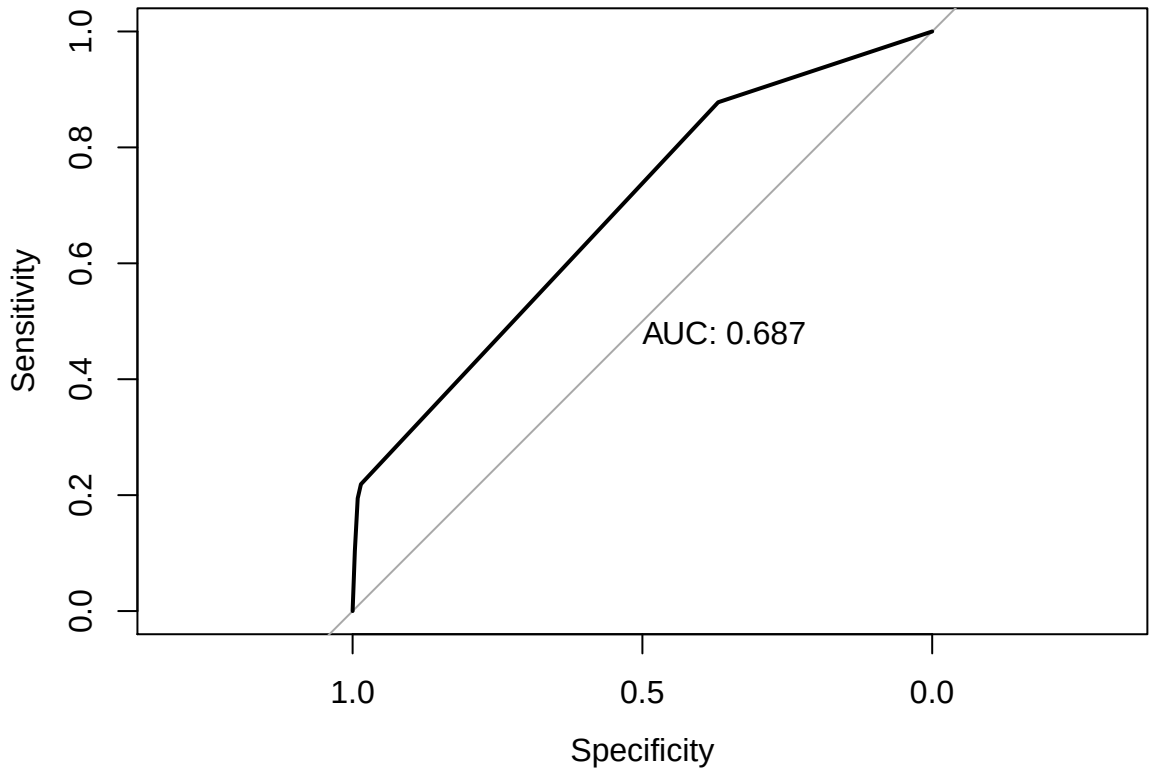


Figure 21: ROC of Decision Tree for First Trip Completion

Table 3: Rules of Decision Tree (With Dates)

	complete														
2	0.04	when	vehicle_added_date.d	>=			33								
6	0.12	when	vehicle_added_date.d	<	33			&	tier	is	C				
28	0.33	when	vehicle_added_date.d	is	17	to	33	&	tier	is	A or B or S	&	signup_date.d	<	13
29	0.62	when	vehicle_added_date.d	is	17	to	33	&	tier	is	A or B or S	&	signup_date.d	>=	13
15	0.76	when	vehicle_added_date.d	<	17			&	tier	is	A or B or S				

Figure 21 shows the ROC curve. The AUC is 0.687, which is not very good. In fact, only 3 predictors were used in this decision tree.

Table 3 shows the rule of this model.

[**Implication**]

Instead of signup date, signup source and other preditors that were considered important in the pre-analysis, this model chose 3 predictors out of 8. This could be the reason why the performance is not good.

Decision Tree (With Dates, After Feature Selection)

```
varImp(dt.uber)
```

```
##                      Overall
## bgc_date.d        354.943270
## new                 3.556948
## signup_channel    559.009738
## signup_date.d     169.370935
```

Table 4: Rules of Decision Tree (With Dates, After Feature Selection)

	complete																
2	0.05	when	vehicle_added_date.d	>=			33										
50	0.09	when	vehicle_added_date.d	<	21			&	tier	is	C	&	signup_date.d	<	16		
24	0.09	when	vehicle_added_date.d	is	21	to	33	&	tier	is	C	&	signup_date.d	<	22		
28	0.33	when	vehicle_added_date.d	is	17	to	33	&	tier	is	A or B or S	&	signup_date.d	<	13		
29	0.62	when	vehicle_added_date.d	is	17	to	33	&	tier	is	A or B or S	&	signup_date.d	>=			13
13	0.73	when	vehicle_added_date.d	<	33			&	tier	is	C	&	signup_date.d	>=			22
15	0.76	when	vehicle_added_date.d	<	17			&	tier	is	A or B or S						
51	0.81	when	vehicle_added_date.d	<	21			&	tier	is	C	&	signup_date.d	is	16	to	22

```
## signup_os              233.983819
## tier                  1045.622718
## vehicle_added_date.d 1288.400960
## city_name               0.000000
```

Hence, `new` and `city_name` will be dropped.

```
dt.uber.2 <- rpart(complete ~ signup_os + signup_channel +
                     vehicle_added_date.d + signup_date.d + tier,
                   data = uber.train, method = "class",
                   control = rpart.control(cp = 0.005))
dt.uber.sum.2 <- summary(dt.uber.2)
```

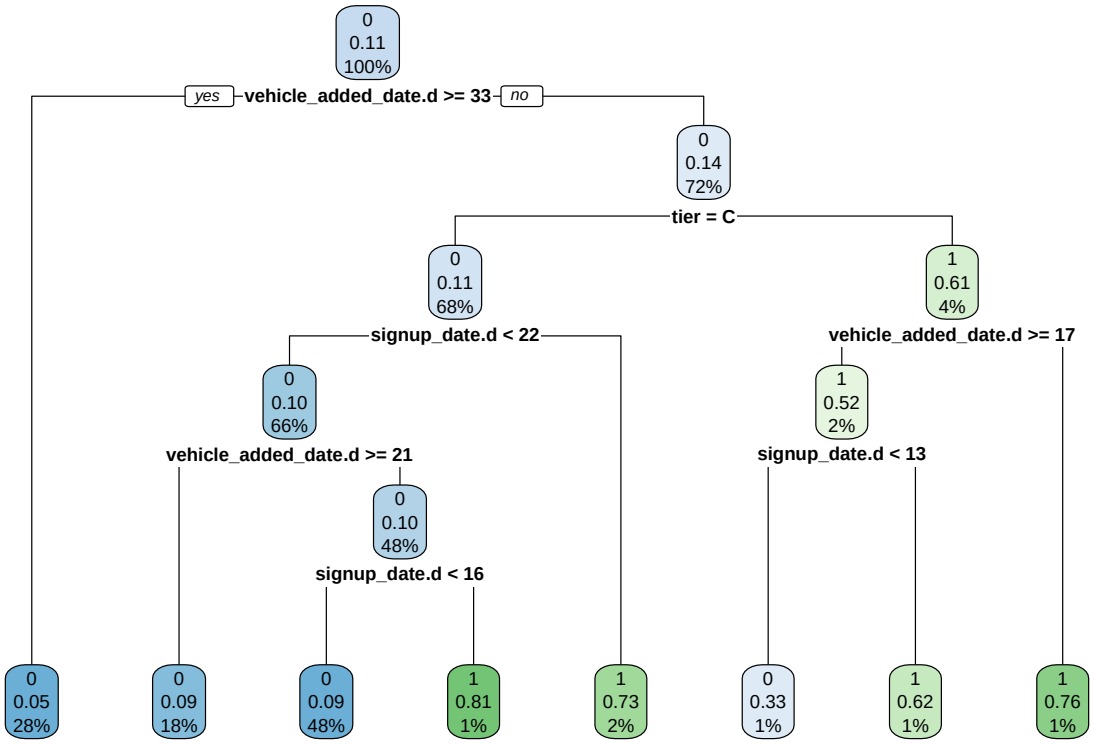


Figure 22: Decision Tree (With Dates, After Feature Selection) for First Trip Completion

[Interpretation]

After removing the two least important variables - city and vehicle conditions, Figure 23 shows that the AUC score has slightly rised to 0.718. Figure 22 shows that there are 4 predictors included. 3 of them are the

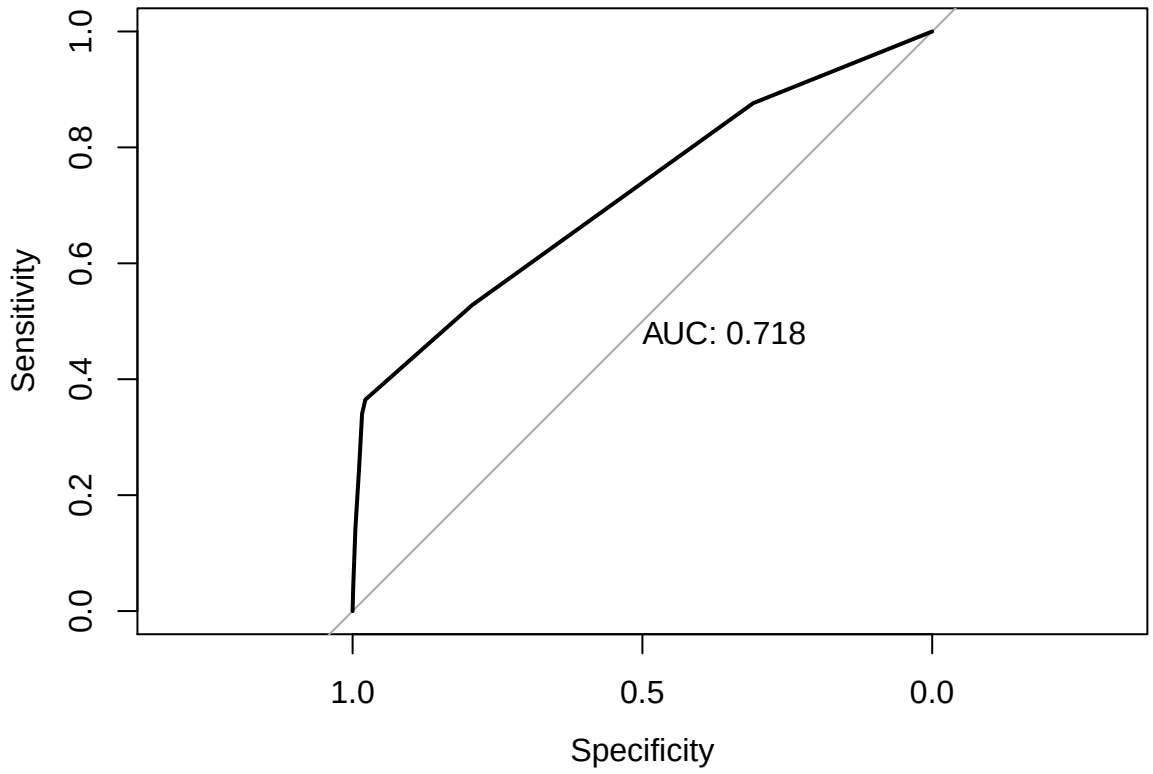


Figure 23: ROC of Decision Tree (With Dates, After Feature Selection) for First Trip Completion

same as the previous tree, and the new one is. 28% of the drivers who added vehicles after February 3 were observed with only 3% of the chance of finishing the first trip. 48% of the drivers who signed up before Jan 17, 2017 and added vehicles before Jan 22, 2016, showed 9% chances to finish the first trip.

Table 4 shows the rule of this model.

[Implication]

The two most important variables are signup date and vehicle registration date in Figure 22, which means that the variables related to time are very important to . However, it is very difficult to explain the correlation between these factors and the possibility of completion. Therefore, in the next decision tree model,

Decision Tree (With Intervals)

```
dt.uber.3 <- rpart(complete ~ signup_os + signup_channel + city_name +
                     signup_bgc + signup_v + tier + v_bgc + new,
                   data = uber.train, method = "class",
                   control = rpart.control(cp = 0.005))
dt.uber.sum.3 <- summary(dt.uber.3)
```

```
varImp(dt.uber.3)
```

```
##                   Overall
## city_name        11.27153
## new              12.41418
## signup_bgc      767.76171
## signup_channel 1161.13142
## signup_os       164.08218
```

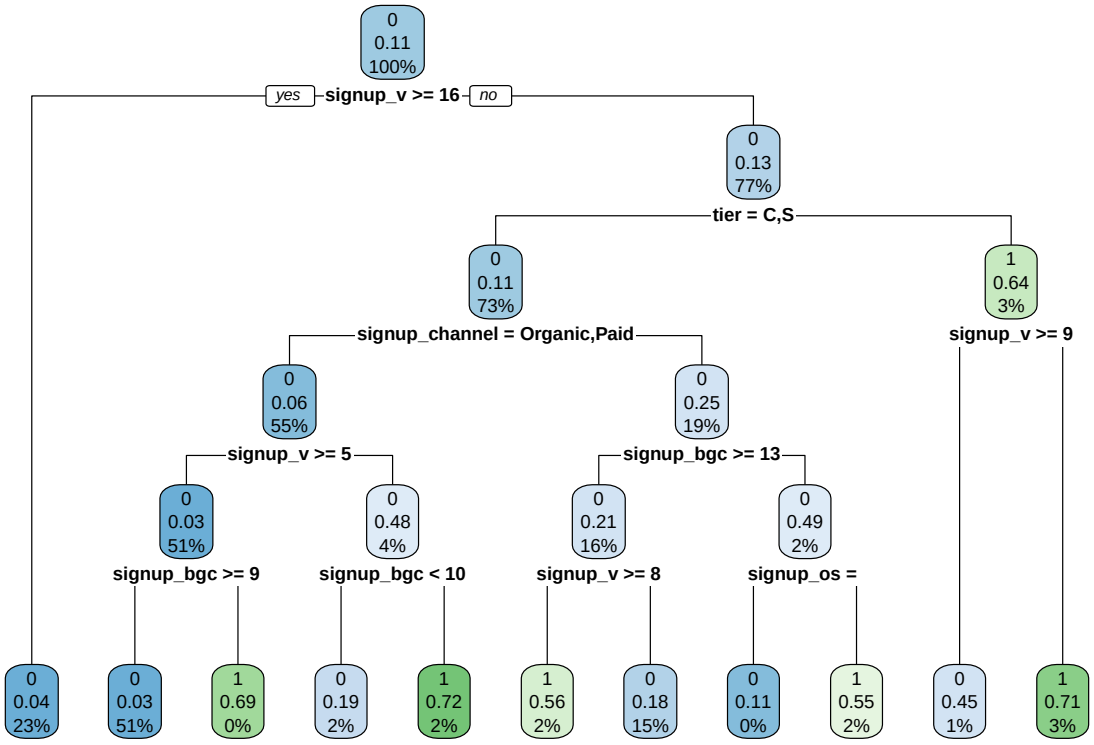


Figure 24: Decision Tree (With Intervals) for First Trip Completion

Table 5: Rules of Decision Tree (With Intervals)

	complete																				
48	0.03	when	signup_v	is	5	to	16	&	tier	is	C or S	&	signup_channel	is	Organic or Paid	&	signup_bgc	>=	9		
2	0.04	when	signup_v	>=			16														
54	0.11	when	signup_v	<	16			&	tier	is	C or S	&	signup_channel	is	Referral	&	signup_bgc	<	13		
53	0.18	when	signup_v	<	8			&	tier	is	C or S	&	signup_channel	is	Referral	&	signup_bgc	>=	13		
50	0.19	when	signup_v	<	5			&	tier	is	C or S	&	signup_channel	is	Organic or Paid	&	signup_bgc	<	10		
14	0.45	when	signup_v	is	9	to	16	&	tier	is	A or B										
55	0.55	when	signup_v	<	16			&	tier	is	C or S	&	signup_channel	is	Referral	&	signup_bgc	<	13	&	signup_os is android web or ios web or mac or other or windows
52	0.56	when	signup_v	is	8	to	16	&	tier	is	C or S	&	signup_channel	is	Referral	&	signup_bgc	>=	13		
49	0.69	when	signup_v	is	5	to	16	&	tier	is	C or S	&	signup_channel	is	Organic or Paid	&	signup_bgc	<	9		
15	0.71	when	signup_v	<	9			&	tier	is	A or B										
51	0.72	when	signup_v	<	5			&	tier	is	C or S	&	signup_channel	is	Organic or Paid	&	signup_bgc	>=	10		

```
## signup_v        2000.63764
## tier            1147.92398
## v_bgc           1023.06751
```

[Interpretation]

Figure 24 shows the decision tree with intervals. The most important variables are signup-vehicle registration intervals, vehicle registration-background check intervals, tiers, signup-background check intervals, sign up channel and signup source. There is only 4% chances for 23% of the drivers who had signup-vehicle registration intervals longer than 16 days. And 3% chances for 51% drivers whose signup-vehicle registration interval is greater than 5 days with a signup channel of either organic and paid, and a signup-background check interval greater than 9 days. The highest possibility, which is 71%, belongs to the drivers whose signup-vehicle registration interval is less than 9 days, and vehicles are not Tier C nor S.

Figure 25 shows that the AUC is 0.834, indicating that the performance is much better than the previous ones.

Table 5 shows the rule of this model.

[Implication]

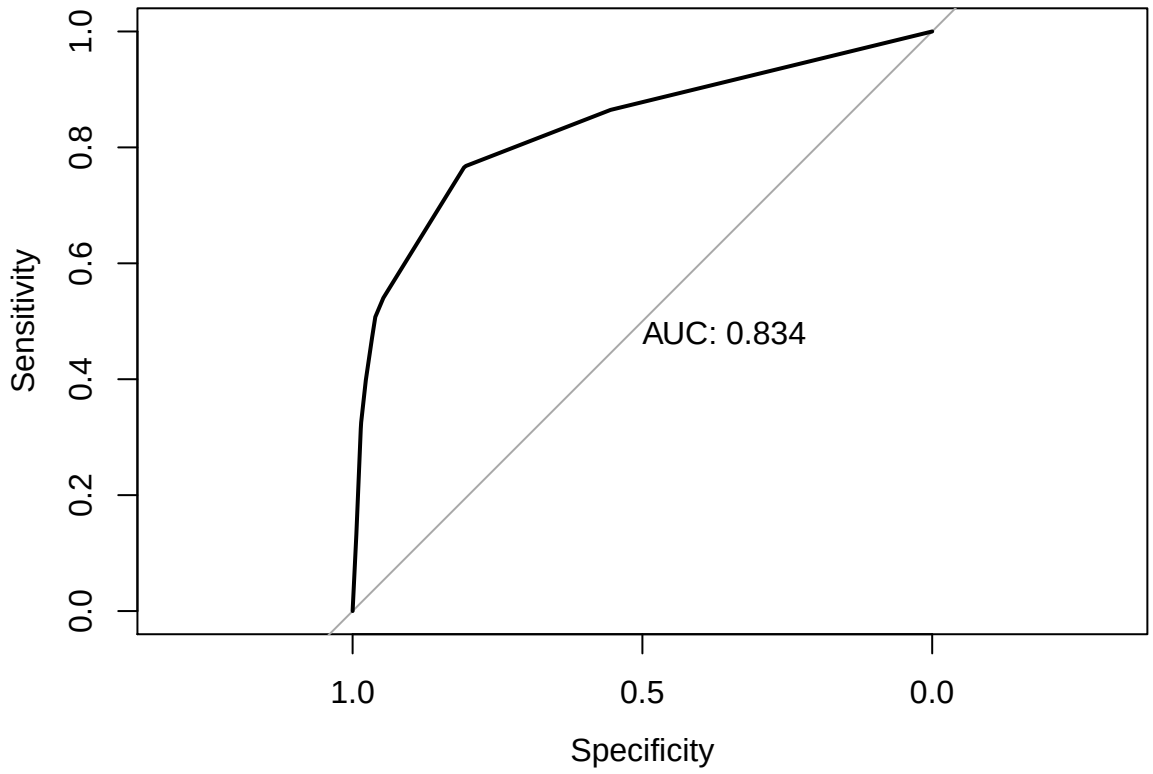


Figure 25: ROC of Decision Tree (With Intervals) for First Trip Completion

Intuitively, the features selected for splitting nodes in this tree make much more sense than the previous trees. The parent node is based on signup-vehicle registration intervals, and Figure 24 shows that generally, the shorter the intervals are, the greater the chances that the drivers would finish the first trip. Another important information is that drivers who signed up thourgh referral channel have a better chance of finishing the first trip than others.

Summary - Decision Tree

According to the bariable importance tables, the most important variables for classification are vehicle added date car brand tier for the desicion tree model with dates, and signup channel, signup-vehicle registration intervals, car brand tier and vehicle registration-background check intervals. One possible reason is that, since the missing values are still included in the model, as long as the vehicle registration and background check dates exist, they may represent that those drivers are more determined to finish the first trip.

Random Forest (With Dates)

```
set.seed(123)
rf.uber.1 <- randomForest(factor(complete) ~ city_name + signup_os + signup_channel +
                    signup_date.d + bgc_date.d + vehicle_added_date.d + tier + new,
                  data = uber.train, na.action = na.omit)
rf.uber.1
```

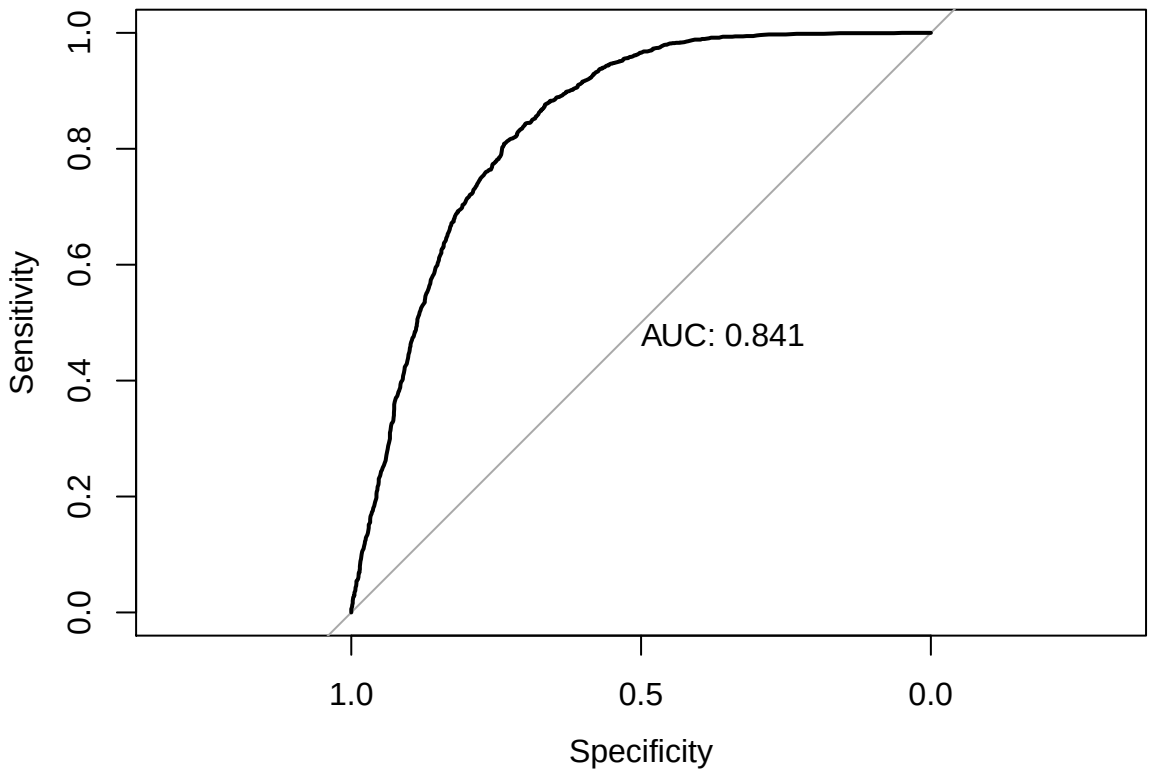
```
##
## Call:
##  randomForest(formula = factor(complete) ~ city_name + signup_os +      signup_channel + signup_date
##                Type of random forest: classification
```

```
##                       Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 22.3%
## Confusion matrix:
##      0    1 class.error
## 0 3770 1211   0.2431239
## 1  806 3257   0.1983756
```

```
importance(rf.uber.1)
```

```
##                       MeanDecreaseGini
## city_name                     134.8312
## signup_os                     280.8275
## signup_channel                145.1504
## signup_date.d                 612.1636
## bgc_date.d                    721.7581
## vehicle_added_date.d         1511.7764
## tier                          119.6299
## new                           238.6723
```

```
rf.test <- predict(rf.uber.1, uber.test, type="prob")
plot.roc(uber.test$complete, rf.test[,2], print.auc = TRUE)
```



[**Interpretation**]

The summary shows that the out-of-bag estimate of error rate is 22.3%. The classification error rate for class 0 (no completion) is 24.31%, and for class 1 (completion) is 19.8%. The most important predictor is vehicle added date, background check date and sigup date.

[**Implication**]

The top 3 important predictors are all related to time.

Random Forest (With Dates, Encoded)

```
set.seed(123)
rf.uber.2 <- randomForest(factor(complete) ~ city_name + signup_os + signup_channel +
                  signup_date.d + bgc_date.d + vehicle_added_date.d + tier + new,
                data = uber.num.train, na.action = na.omit)
rf.uber.2
```

```
##
## Call:
##  randomForest(formula = factor(complete) ~ city_name + signup_os +      signup_channel + signup_date
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 5.97%
## Confusion matrix:
##       0    1 class.error
## 0 32890 1141  0.03352825
## 1  1143 3102  0.26925795
```

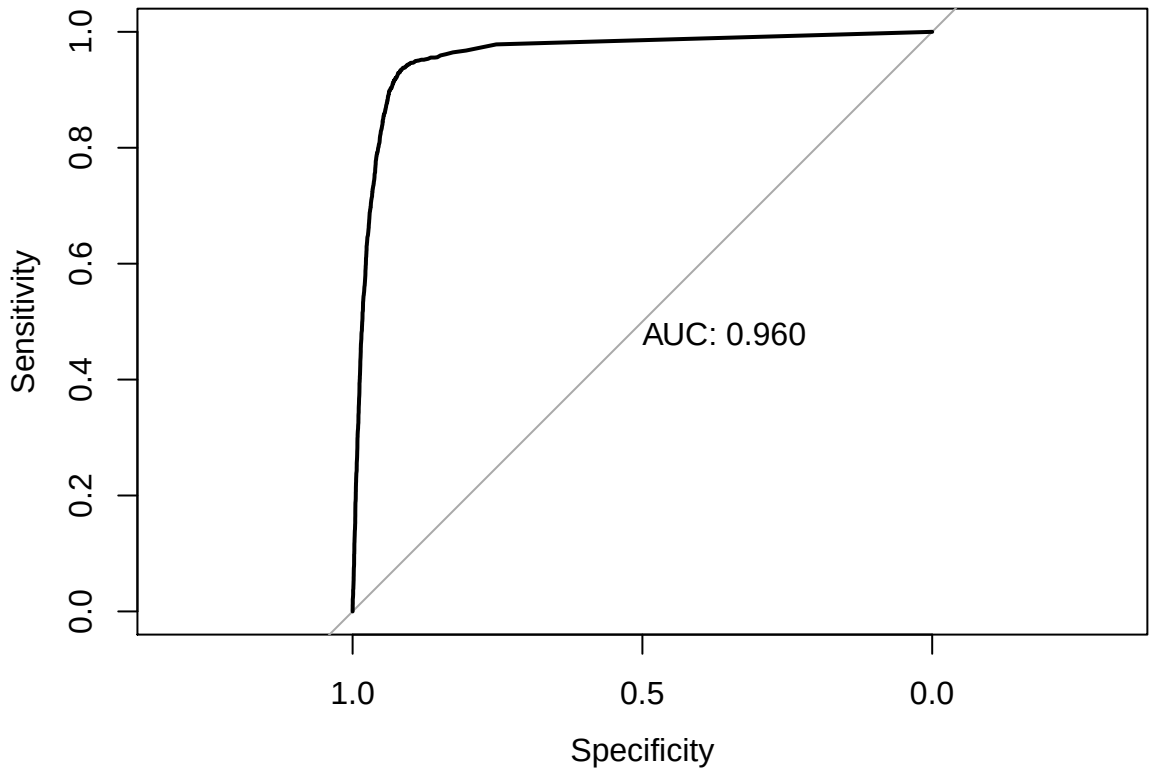


Figure 26: ROC of Random Forest (With Dates, Encoded) for First Trip Completion

```
importance(rf.uber.2)
```

```
##                      MeanDecreaseGini
## city_name                    145.6437
## signup_os                    290.9365
```

```
## signup_channel               204.1451
## signup_date.d                630.0191
## bgc_date.d                   808.5008
## vehicle_added_date.d        3136.7823
## tier                         239.8087
## new                          807.9051
```

[Interpretation]

The random forest for classification generated 500 trees. The out-of-bag error rate is 5.97%, and classification error for class 0 (no completion) is 3.35%, for class 1 (completion) is 26.9%. Figure 26 shows that the AUC is 0.96, which is very satisfying. The top 3 important predictors are vehicle registration date, background check date and vehicle condition.

[Implication]

The encoded variables has two main changes: 1) The missing values are roughly fixed. 2) All the variables are encoded as numbers. Then the random forest model has reached the highest AUC so far. Vehicle condition becomes one of the most important variables instead of background check date.

Random Forest (With Intervals)

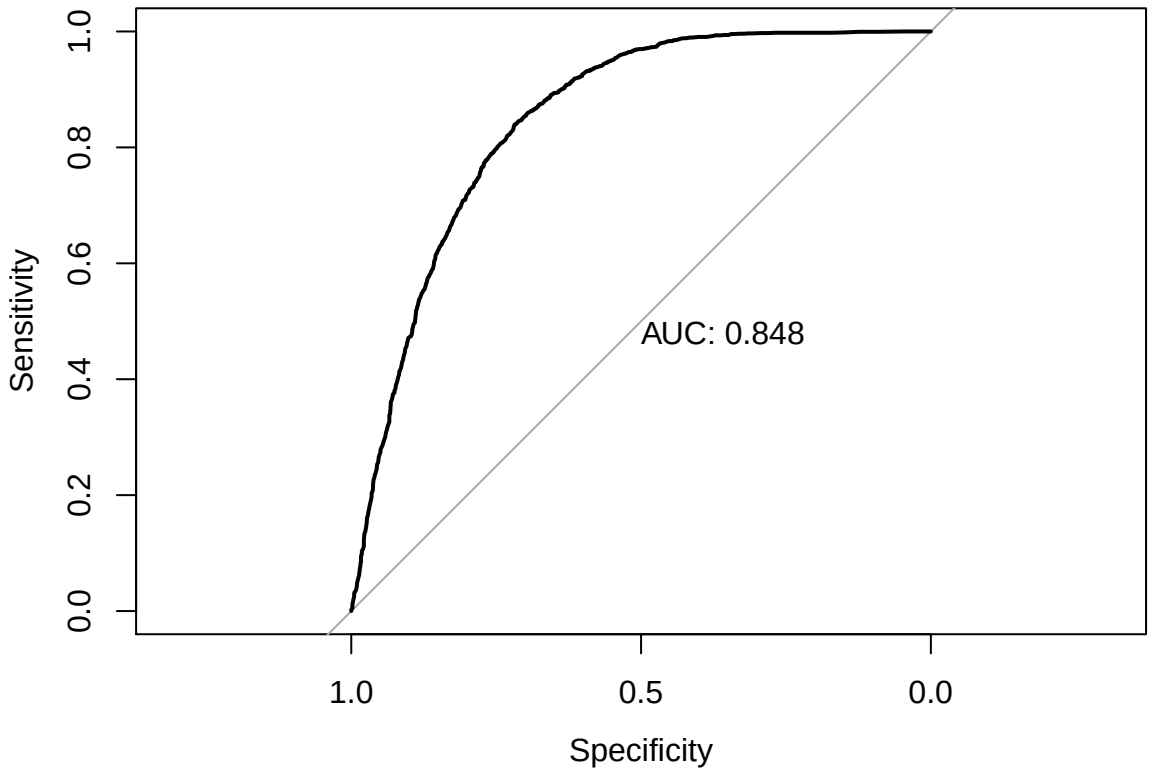
```
set.seed(123)
rf.uber.3 <- randomForest(as.factor(complete) ~ city_name + signup_os +
                            signup_channel + v_bgc + signup_bgc + signup_v + tier + new,
                  data = uber.train, na.action = na.omit)
rf.uber.3
```

```
##
## Call:
##  randomForest(formula = as.factor(complete) ~ city_name + signup_os +      signup_channel + v_bgc + s
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 22.36%
## Confusion matrix:
##      0    1 class.error
## 0 3772 1209   0.2427223
## 1  813 3250   0.2000984
```

```
importance(rf.uber.3)
```

```
##                MeanDecreaseGini
## city_name              130.5835
## signup_os              272.6062
## signup_channel         140.9420
## v_bgc                  749.3539
## signup_bgc             668.0579
## signup_v              1491.6560
## tier                   120.4065
## new                    231.5238
```

```
rf.test.3 <- predict(rf.uber.3, uber.test, type="prob")
plot.roc(uber.test$complete, rf.test.3[,2], print.auc = TRUE)
```



[**Interpretation**]

The random forest for classification generated 500 trees. The out-of-bag error rate is 22.36%, and classification error for class 0 is 24.63%, for class 1 is 20.26%. The top 3 important variables are signup-vehicle registration intervals, vehicle-background check intervals and signup-background check intervals.

[**Implication**]

Time-related factors are still the most important ones.

Random Forest (With Intervals, Encoded)

```
set.seed(123)
rf.uber.4 <- randomForest(as.factor(complete) ~ city_name + signup_os +
                            signup_channel + v_bgc + signup_bgc + signup_v + tier + new,
                  data = uber.num.train, na.action = na.omit)
rf.uber.4
```

```
##
## Call:
##  randomForest(formula = as.factor(complete) ~ city_name + signup_os +      signup_channel + v_bgc + s
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 5.87%
## Confusion matrix:
##       0    1 class.error
## 0 32874 1157  0.03399841
## 1  1088 3157  0.25630153
```

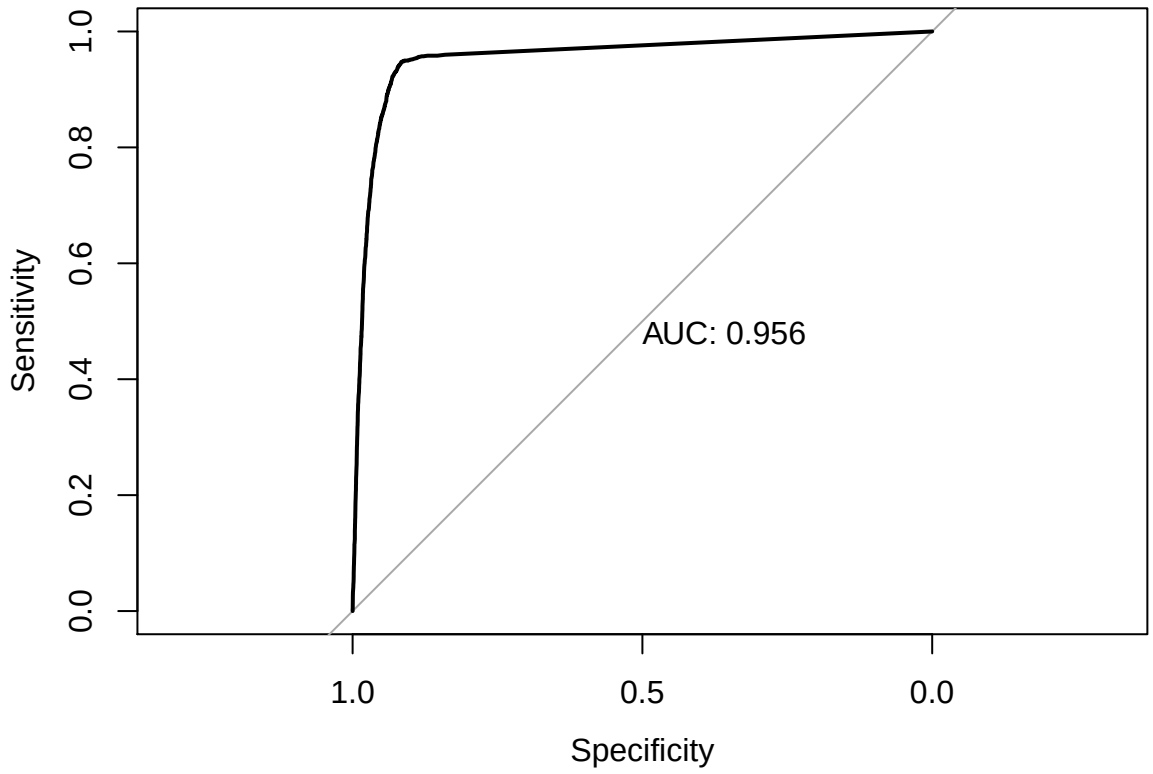


Figure 27: ROC of Random Forest (With Intervals, Encoded) for First Trip Completion

```
importance(rf.uber.4)
```

```
##                MeanDecreaseGini
## city_name              136.0146
## signup_os              261.2129
## signup_channel         170.6134
## v_bgc                 1622.3369
## signup_bgc             748.6127
## signup_v              2725.8167
## tier                   196.2160
## new                    536.9537
```

The top 3 important variables are signup-vehicle registration intervals, vehicle-background check intervals and signup-background check intervals.

[**Interpretation**]

The random forest for classification generated 500 trees. The out-of-bag error rate is 5.87%, and classification error for class 0 is 3.40%, for class 1 is 25.63%. Figure 27 shows that the AUC is 0.956, which is good. The top 3 most important predictors are

[**Implication**]

The reason that this AUC is slightly less that the reandom forest model with dates may be the randomness feature of the model.

Summary - Random Forest

The biggest difference with the random forest models are 1) the missing values are ommited or 2) the categorical variables are encoded, hence the missing values can be roughly fixed with mediums. According to the variable importance tables, in models with dates, the most important variable is the vehicle registration date; in models with intervals, the most important variables are the intervals between signup and vehicle registration, and the intervals between vehicle registration and background check.

Neural Network (With Dates)

```
net.uber <- nnet(complete ~ city_name + signup_os + signup_channel +
                   signup_date.d + vehicle_added_date.d + new + tier,
                 data = uber.train, size = 2, decay = 0.05, maxit = 2000)
```

```
summary(net.uber)
```

```
## a 18-2-1 network with 41 weights
## options were - decay=0.05
##   b->h1  i1->h1  i2->h1  i3->h1  i4->h1  i5->h1  i6->h1  i7->h1  i8->h1
##   -1.89    2.01   -1.26    1.65   -2.02   -1.73   -0.73   -1.09   -1.35
##  i9->h1 i10->h1 i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1
##   -1.84   -3.09    0.57    0.65   -0.75   -0.66   -0.78   -0.07   -2.47
## i18->h1
##    0.00
##   b->h2  i1->h2  i2->h2  i3->h2  i4->h2  i5->h2  i6->h2  i7->h2  i8->h2
##   -0.32    0.07    0.24   -0.65   -0.67   -0.89   -0.96   -0.92   -0.01
##  i9->h2 i10->h2 i11->h2 i12->h2 i13->h2 i14->h2 i15->h2 i16->h2 i17->h2
##   -0.45   -0.08    0.10   -0.36   -0.33   -0.38   -0.48    0.04   -0.04
## i18->h2
##    0.35
##  b->o h1->o h2->o
##  2.83 -0.82 -6.80
```

[Interpretation]

Baseline: The completion rate for drivers from Berton, sign up through unknow source and organic channel, and the vehicle condition is unacceptable.

Formula: `complete` ~ `city_name` (Berton) + `signup_os` (Unknown) + `signup_channel` (Organic) + `new` (Unacceptable) + `signup_date.d` + `vehicle_added_date.d`

Figure 30 the AUC score is 0.860. It is a 15-2-1 network with 35 weights. Figure 29 shows that compared to the baseline model, most of the changes will have negative impact on the possibilities. The strongest negative effect comes from signup date.

[Implication]

The AUC score of Neural Network is close to that of logistic regression even with out feature selection.

Neural Network (With Intervals)

```
net.uber.2 <- nnet(complete ~ city_name + signup_os + signup_channel +
                   signup_v + signup_bgc + v_bgc + new + tier,
                 data = uber.train, size = 2, decay = 0.05, maxit = 2000)
```

```
net.uber.2
```

```
## a 19-2-1 network with 43 weights
## inputs: city_nameStrark city_nameWrouver signup_osandroid web signup_osios web signup_osmac signup_os
## output(s): complete
```

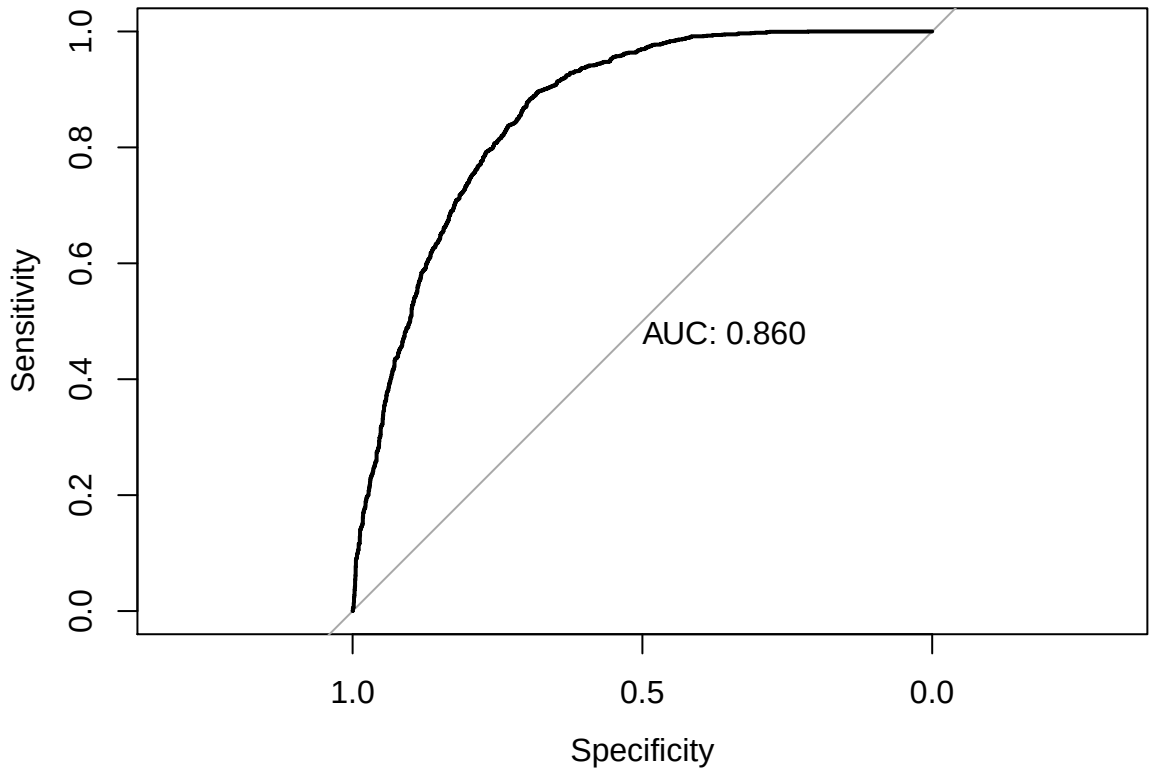


Figure 28: ROC of Neural Network for First Trip Completion

```
## options were - decay=0.05
```

```
summary(net.uber.2)
```

```
## a 19-2-1 network with 43 weights
## options were - decay=0.05
##   b->h1  i1->h1  i2->h1  i3->h1  i4->h1  i5->h1  i6->h1  i7->h1  i8->h1
##    1.54   -0.67    0.39    0.30    1.94    0.76   -0.73    2.00   -1.95
##  i9->h1 i10->h1 i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1
##    1.87   -0.10   -0.05    0.52   -0.31    0.38   -0.53   -0.20   -0.38
## i18->h1 i19->h1
##    1.60    0.44
##   b->h2  i1->h2  i2->h2  i3->h2  i4->h2  i5->h2  i6->h2  i7->h2  i8->h2
##   -0.61    0.05    0.32   -0.72   -0.63   -1.03   -1.15   -0.92   -0.17
##  i9->h2 i10->h2 i11->h2 i12->h2 i13->h2 i14->h2 i15->h2 i16->h2 i17->h2
##   -0.39    0.11    0.00    0.02   -0.45   -0.37   -0.49   -0.62    0.02
## i18->h2 i19->h2
##    0.08    0.26
##  b->o h1->o h2->o
##  1.70  0.93 -6.41
```

[**Interpretaition**]

Baseline: The completion rate for drivers from Berton, sign up through unknow source and organic channel, and the vehicle condition is unacceptable.

Formula: `complete` ~ `city_name` (Berton) + `signup_os` (Unknown) + `signup_channel` (Organic) + `new` (Unacceptable) + `signup_v` + `signup_bgc`

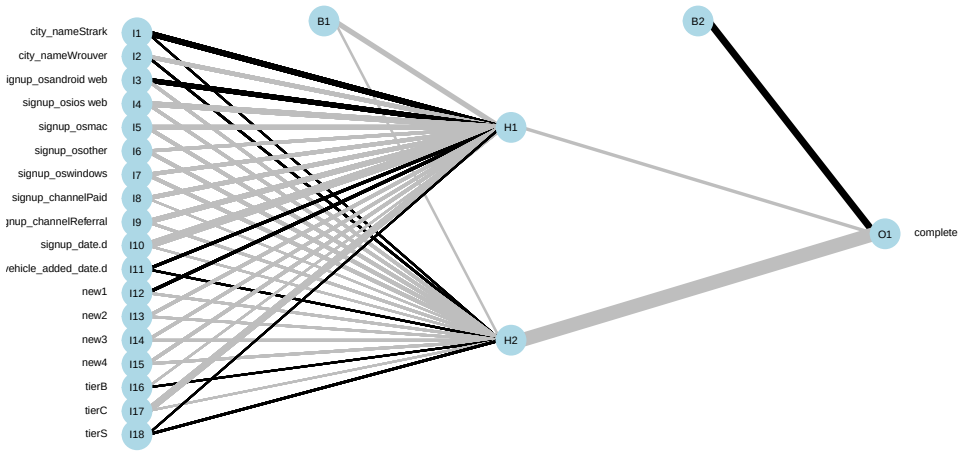


Figure 29: Neural Network (With Dates) for First Trip Completion

Figure 30 shows that the AUC score is 0.856. It is a a 19-2-1 network with 43 weights. Figure 31 shows the plot of this model and nearly half of the changes can cause positive changed to the possibilities. Although influece of the changes are mixed, the strongest comes from signup sources and car brand tiers.

[**Implication**]

Considering the new features, only signup-vehicle registration interval has positive impacts on both hidden layers, which means that the longer signup-vehicle registration interval actually helps to increase the possibility of first trip completion.

Summary - Neural Network

The three most important variables from the outcomes of both models are: car brand tiers, signup sources and vehicle added date.

Summary - All Models

The most important independent variables vary form model to model. However, the best model has reached a ROC of 0.96. Since most of the predictors have missing values, fixing problem can really help improve the performance of the models.

Signup-Vehicle Registration Interval

```
length(uber.train$signup_v[!is.na(uber.test$signup_v)])
```

```
## [1] 9106
```

```
length(uber.test$signup_v[!is.na(uber.test$signup_v)])
```

```
## [1] 3912
```

In this section, only records with vehicle registration dates will be used. Therefore, there are 9106 records in the traning set, and 3912 records in the test set.

Linear Regression

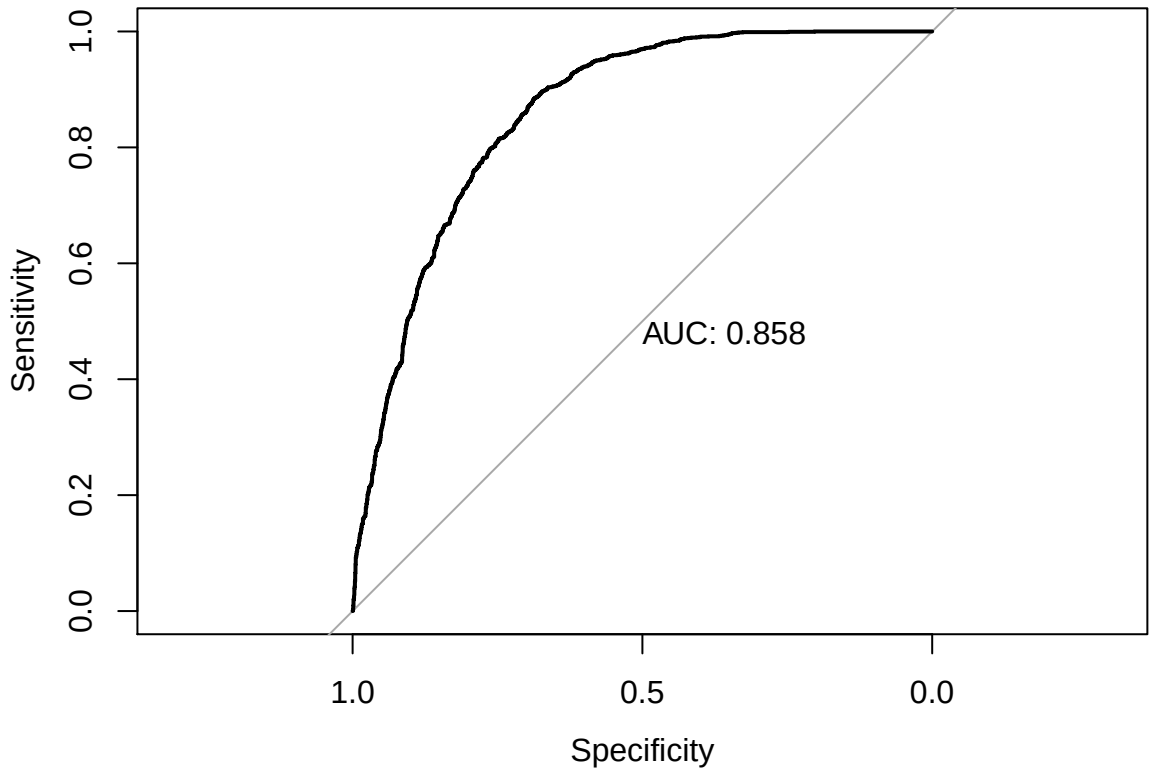


Figure 30: ROC of Neural Network (With Intervals) for First Trip Completion

```
lm.uber <- lm(signup_v ~ city_name + signup_os + signup_channel +
                signup_date.d + new,
              data = uber.train, subset = !is.na(signup_v))
summary(lm.uber)
```

```
##
## Call:
## lm(formula = signup_v ~ city_name + signup_os + signup_channel +
##     signup_date.d + new, data = uber.train, subset = !is.na(signup_v))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -22.786 -11.081  -4.012   9.368  58.540
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              25.84877    0.97495  26.513  < 2e-16 ***
## city_nameStrark          -0.42071    0.30705  -1.370 0.170664
## city_nameWrouver          0.01539    0.56879   0.027 0.978419
## signup_osandroid web     -2.60248    0.77496  -3.358 0.000788 ***
## signup_osios web         -3.46061    0.76016  -4.552 5.37e-06 ***
## signup_osmac             -4.27067    0.81381  -5.248 1.57e-07 ***
## signup_osother           -4.71223    0.89674  -5.255 1.51e-07 ***
## signup_oswindows         -4.69274    0.82181  -5.710 1.16e-08 ***
## signup_channelPaid       -0.21594    0.41502  -0.520 0.602855
## signup_channelReferral -2.70823    0.37488  -7.224 5.44e-13 ***
```

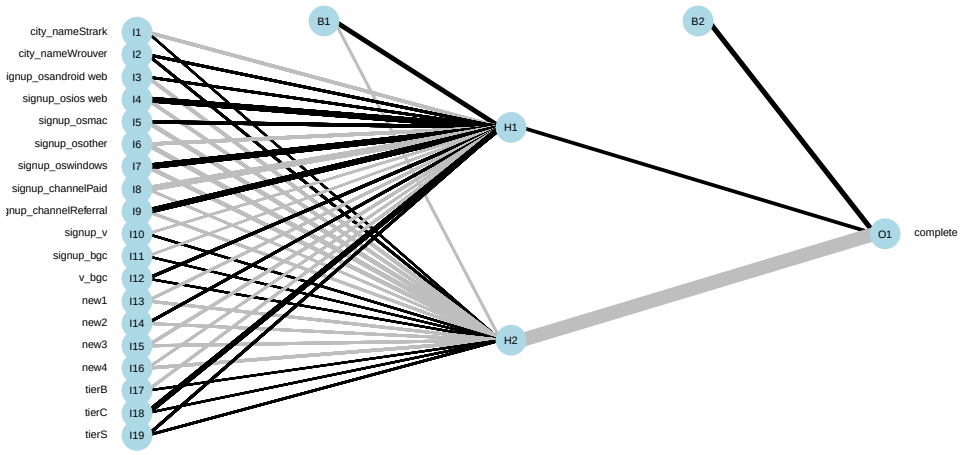


Figure 31: ROC of Neural Network (With Intervals) for First Trip Completion

```
## signup_date.d          -0.28352    0.01688 -16.799  < 2e-16 ***
## new1                   -1.22421    0.67038  -1.826 0.067860 .
## new2                   -1.44204    0.68703  -2.099 0.035849 *
## new3                   -2.01967    0.63553  -3.178 0.001488 **
## new4                   -1.64394    0.61088  -2.691 0.007134 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.78 on 9207 degrees of freedom
## Multiple R-squared:  0.04103,    Adjusted R-squared:  0.03958
## F-statistic: 28.14 on 14 and 9207 DF,  p-value: < 2.2e-16
```

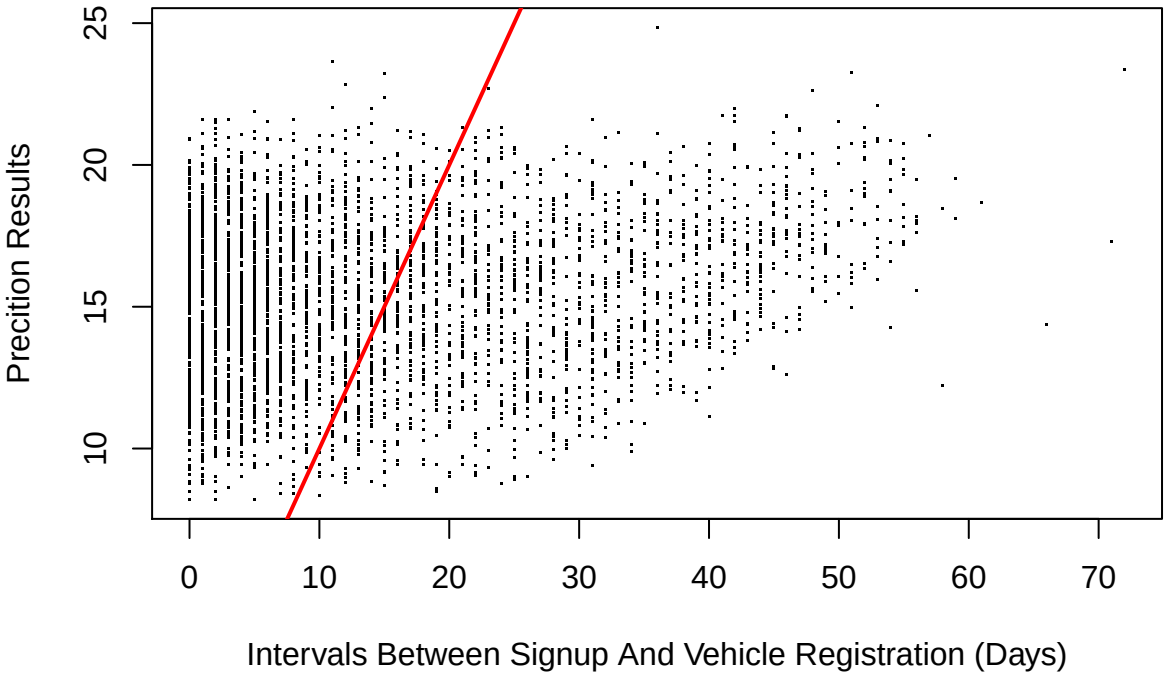


Figure 32: Linear Regression, Predicted vs. Actual

```
RMSE(lm.test, uber.test$signup_v[!is.na(uber.test$signup_v)])
```

```
## [1] 13.83091
```

[Interpretation]

The summary shows that the R-squared is 0.03958, which means that only 4% of the variance can be explained by the model. The RMSE is close to 13.8309.

[Implication]

Since most of the predictors in the model are categorial variables, linear regression may not be the best way for prediction.

Linear Regression (Roughly Fixed, Encoded)

```
lm.uber.2 <- lm(signup_v ~ signup_os + signup_channel +
                  signup_date.d + tier + new + city_name,
                data = uber.num.train, subset = !is.na(signup_v))
summary(lm.uber.2)
```

```
##
## Call:
## lm(formula = signup_v ~ signup_os + signup_channel + signup_date.d +
##     tier + new + city_name, data = uber.num.train, subset = !is.na(signup_v))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.285  -1.458  -0.857  -0.190  59.297
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    19.416370   0.290503  66.837   <2e-16 ***
## signup_os       0.001009   0.024244   0.042   0.9668
## signup_channel -0.092449   0.049302  -1.875   0.0608 .
## signup_date.d  -0.063683   0.004135 -15.400   <2e-16 ***
## tier           -2.200811   0.099679 -22.079   <2e-16 ***
## new            -0.552971   0.056987  -9.703   <2e-16 ***
## city_name      -0.026568   0.058464  -0.454   0.6495
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.054 on 38269 degrees of freedom
## Multiple R-squared:  0.02392,    Adjusted R-squared:  0.02376
## F-statistic: 156.3 on 6 and 38269 DF,  p-value: < 2.2e-16
```

```
RMSE(lm.test.2, uber.num.test$signup_v[!is.na(uber.num.test$signup_v)])
```

```
## [1] 7.03558
```

[Interpretation]

R-squared is 0.02376, meaning that only 2.3% of the variance is explained by this model.

[Implication]

Both linear regression models have failed to fit the dataset and perform prediction.

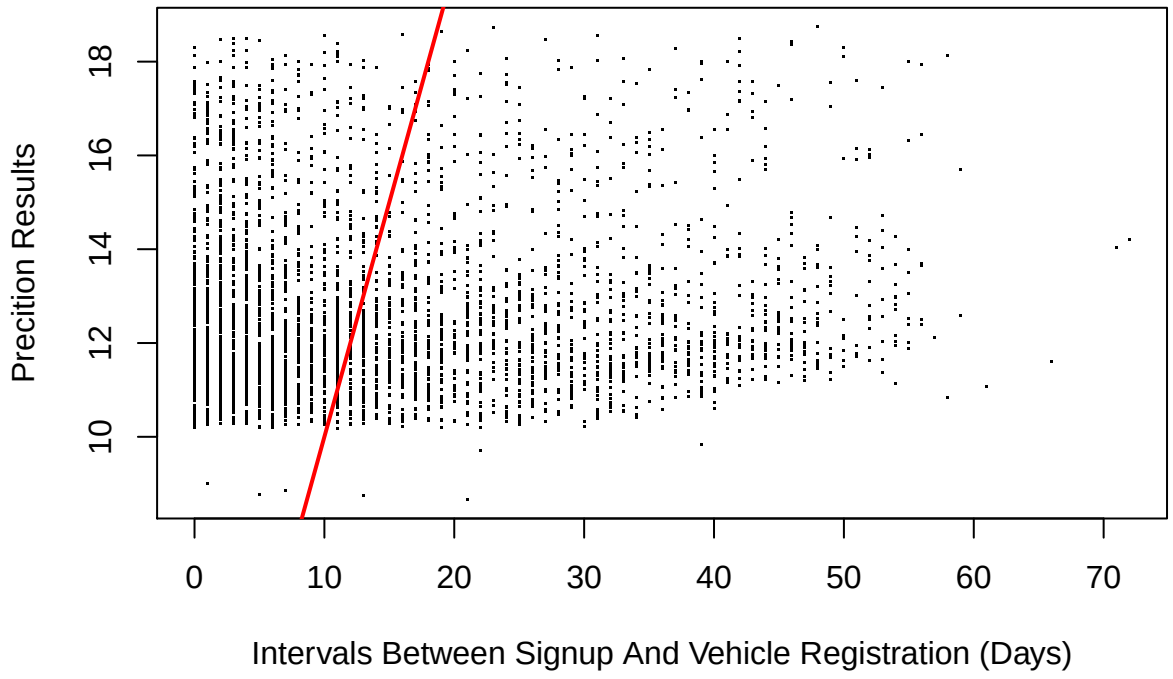


Figure 33: Linear Regression (Roughly Fixed, Encoded), Predicted vs. Actual

Cross-Validation

Logistic Regression (With Dates)

```
set.seed(123)
form <- "complete ~ city_name + signup_os + signup_channel + signup_date.d + vehicle_added_date.d + new
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- glm(form , train, family = "binomial")
 tmp.predict <- predict(tmp.model, newdata = test, type = "response")
 conf.mat <- table(test$complete, tmp.predict)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 99.924 percent"
```

Logistic Regression (With Intervals)

```
set.seed(123)
form <- "complete ~ city_name + signup_os + signup_channel + signup_v + signup_bgc + new"
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))
```

```
for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- glm(form , train, family = "binomial")
 tmp.predict <- predict(tmp.model, newdata = test, type = "response")
 conf.mat <- table(test$complete, tmp.predict)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 99.922 percent"
```

Decision Tree (With Dates)

```
set.seed(123)
form <- "complete ~ city_name + signup_os + signup_channel + new + signup_date.d+ bgc_date.d + vehicle_
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- rpart(form , train, method = "class")
 tmp.predict <- predict(tmp.model, newdata = test, type = "class")
 conf.mat <- table(test$complete, tmp.predict)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 10.146 percent"
```

Decision Tree (With Dates, After Feature Selection)

```
set.seed(123)
form <- "complete ~ signup_os + signup_channel + vehicle_added_date.d + signup_date.d + tier"
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- rpart(form , train, method = "class")
 tmp.predict <- predict(tmp.model, newdata = test, type = "class")
 conf.mat <- table(test$complete, tmp.predict)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 9.217 percent"
```

Decision Tree (With Intervals)

```
set.seed(123)
form <- "complete ~ signup_os + signup_channel + city_name + signup_bgc + signup_v + tier + v_bgc + new
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- rpart(form , train, method = "class")
 tmp.predict <- predict(tmp.model, newdata = test, type = "class")
 conf.mat <- table(test$complete, tmp.predict)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 9.371 percent"
```

Random Forest (With Dates)

```
set.seed(123)
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- randomForest(factor(complete) ~ city_name + signup_os +
                             signup_channel + signup_date.d + bgc_date.d +
                             vehicle_added_date.d + tier + new,
                           train, na.action = na.omit)
 tmp.predict <- predict(tmp.model, newdata = test, type="prob")
 tmp.predict.class <- rep(0, length(tmp.predict[,2]))
 tmp.predict.class[tmp.predict[,2] > 0.5] <- 1
 conf.mat <- table(test$complete, tmp.predict.class)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 5.839 percent"
```

Random Forest (With Dates, Encoded)

```
set.seed(123)
folds <- split(uber, cut(sample(1:nrow(uber.num)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
```

```
 tmp.model <- randomForest(factor(complete) ~ city_name + signup_os +
                             signup_channel + signup_date.d + bgc_date.d +
                             vehicle_added_date.d + tier + new,
                           train, na.action = na.omit)
 tmp.predict <- predict(tmp.model, newdata = test, type="prob")
 tmp.predict.class <- rep(0, length(tmp.predict[,2]))
 tmp.predict.class[tmp.predict[,2] > 0.5] <- 1
 conf.mat <- table(test$complete, tmp.predict.class)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 5.839 percent"
```

Random Forest (With Intervals)

```
set.seed(123)
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- randomForest(as.factor(complete) ~ city_name + signup_os +
                             signup_channel + v_bgc + signup_bgc +
                              signup_v + tier + new,
                           train, na.action = na.omit)
 tmp.predict <- predict(tmp.model, newdata = test, type="prob")
 tmp.predict.class <- rep(0, length(tmp.predict[,2]))
 tmp.predict.class[tmp.predict[,2] > 0.5] <- 1
 conf.mat <- table(test$complete, tmp.predict.class)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 5.709 percent"
```

Random Forest (With Intervals, Encoded)

```
set.seed(123)
folds <- split(uber, cut(sample(1:nrow(uber.num)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- randomForest(as.factor(complete) ~ city_name + signup_os +
                             signup_channel + v_bgc + signup_bgc +
                              signup_v + tier + new,
                           train, na.action = na.omit)
 tmp.predict <- predict(tmp.model, newdata = test, type="prob")
```

```
 tmp.predict.class <- rep(0, length(tmp.predict[,2]))
 tmp.predict.class[tmp.predict[,2] > 0.5] <- 1
 conf.mat <- table(test$complete, tmp.predict.class)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
```

Neural Network (With Dates)

```
set.seed(123)
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- nnet(complete ~ city_name + signup_os + signup_channel +
                   signup_date.d + vehicle_added_date.d + new + tier,
                 data = train, size = 2, decay = 0.05, maxit = 2000)
 tmp.predict <- predict(tmp.model, newdata = test)
 conf.mat <- table(test$complete, tmp.predict)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 99.924 percent"
```

Neural Network (With Intervals)

```
set.seed(123)
folds <- split(uber, cut(sample(1:nrow(uber)),10))
errs <- rep(NA, length(folds))

for (i in 1:length(folds)) {
 test <- ldply(folds[i], data.frame)
 train <- ldply(folds[-i], data.frame)
 tmp.model <- nnet(complete ~ city_name + signup_os + signup_channel +
                   signup_v + signup_bgc + v_bgc + new + tier,
                 data = train, size = 2, decay = 0.05, maxit = 2000)
 tmp.predict <- predict(tmp.model, newdata = test)
 conf.mat <- table(test$complete, tmp.predict)
 errs[i] <- 1-sum(diag(conf.mat))/sum(conf.mat)
}
```

```
print(sprintf("Average error using k-fold cross-validation: %.3f percent", 100*mean(errs)))
```

```
## [1] "Average error using k-fold cross-validation: 99.922 percent"
```

Table 6: Model Comparison For First Trip Completion

Model	ROC AUC	Average 10-Fold Cross Validation Error
Logistic Regression (With Dates)	0.861	99.924%
Logistic Regression (With Intervals)	0.860	99.922%
Decision Tree (With Dates)	0.687	10.146%
Decision Tree (With Dates, After Feature Selection)	0.718	9.217%
Decision Tree (With Intervals)	0.834	9.371%
Random Forest (With Dates)	0.841	5.839%
Random Forest (With Dates, Encoded)	0.960	5.839%
Random Forest (With Intervals)	0.848	5.709%
Random Forest (With Intervals, Encoded)	0.956	5.709%
Neural Network (With Dates)	0.860	99.924%
Neural Network (With Intervals)	0.858	99.922%

Summary

Table 6 shows that there are 4 models with very high AUC's and low average cross-validation errors. The possible reason is that the result of predictors with any missing values is likely to be NA, and wil be considered as 0 since the posibility is considered under the threshold in ROC curve. In the meantime, the `completion` column of the rows with missing valuesare all 0, so that the AUC score is high. However, in cross validation, only prediction results with a probability above 0.5 are considered as 1 (finished the first trip), hence the average cross validation errors are much higher.

According to the pre-analysis and prediction models above, the most important variables for first trip completion prediction are signup source car condition and vehicle added date (or signup-vehicle registration interval). Compare to drivers signed up from unknown sources, drivers signup through mobile deviced have 2.5 times the chances of finishing the first trip, and those from computers have 3.7 times the chance of finishing the first trip. Also, compared to unacceptable condition vehicle drivers, the drivers with cars with any other conditions all have 1.5-1.8 times of chanced of finishing the first trip. Even though car brand tier is not an important variable in logistic regression models, it is important in decision tree models. As long as the vehicle are from A, B, S brands, the drivers are more likely to finish the first trip. In random forest models, the signup date or signup-vehicle registration interval play the most important roles in unencoded and encoded models, respectively.

Problems

1. What fraction of the driver signups took a first trip?

```
sum(uber$complete == 1)/length(uber$id)
```

```
## [1] 0.1122328
```

Only 11.22% of the drivers who signed up finished their first trips.

2. Build a predictive model to help Uber determin whether or not a driver signup will driving. How valid is the model?

Random forest model with dates and encoded features, with missing values roughly fixed.

Table 7: Data Description

Variable Name	Description
id	The unique ID of the driver.
city_name	The city where the driver is at.
signup_os	Signup source/device.
signup_channel	Signup channel.
signup_date	Signup date.
bgc_date	Background check date.
vehicle_added_date	Vehicle registration date.
vehicle_make	The make of the registered vehicle.
vehicle_model	The model of the registered vehicle.
vehicle_year	In which year the vehicle was made.
first_completed_date	When was the first trip completed.
new (new)	The indicator of the condition of the vehicle. (4 -> 0: Newest to Oldest)
tier (new)	The tier of the car brand. (From luxury to economy: S, A, B, C)
timediff	The interval between signup and first trip.
signup_date.d (new)	Intervals between the signup date and 1/1/2016 in days.
bgc_date_date.d (new)	Intervals between the signup date and 1/1/2016 in days.
vehicle_added_date.d (new)	Intervals between the signup date and 1/1/2016 in days.
signup_v (new)	Intervals between the signup date and vehicle added date.
signup_bgc (new)	Intervals between the signup date and background check date.
v_bgc (new)	Intervals between vehicle added date and background check date.

3. Briefly discuss how Uber might leverage the insights gained from the model to generate more first trips?

1) Uber should come up with more promotions to draw new drivers to signup through computers since the drivers signed up from computer sources are 1.5 times more likely to finish their first trip than drivers who signed up from mobile devices.

2) Uber should prohibit the cars made before 2004 for registration. The owners of these vehicles have a relatively low posibility of finishing the first trip and the vehicles may not be as safe as well.

3) The best channel to attact new drivers is referral. Cut the money for paid promotions, and invest more on referral bonus.

Appendix A: Data Dictionary