NYC Taxi trip

Thiago Feitosa Campos

Introduction

This work is a study of the functions in the **MicrosofML Package**, and some of the codes are from this blog (https://blogs.msdn.microsoft.com/microsoftrservertigerteam/2017/01/17/predicting-nyc-taxi-tips-using-microsoftml/)

Running the preamble and loading the data

```
  rm(list=ls())
  gc()
```

```
##           used (Mb) gc trigger (Mb) max used (Mb)
## Ncells  649715 34.7    1168576 62.5   940480 50.3
## Vcells 1003689  7.7    2060183 15.8  1264642  9.7
```

```
  ################################################
  ############Preparing the packages#############
  ################################################
  # Selectin CRAM Mirror
  local({r <- getOption("repos")
  r["CRAN"] <- "https://vps.fmvz.usp.br/CRAN/"
  options(repos=r)})

  Install_And_Load <- function(Required_Packages)
  {
    Remaining_Packages <- Required_Packages[!(Required_Packages %in% installed.packages()[,"P
ackage"])];

    if(length(Remaining_Packages))
    {
      install.packages(Remaining_Packages, dependencies = TRUE);
    }

    for(package_name in Required_Packages)
    {
      library(package_name,character.only=TRUE,quietly=TRUE);

    }
  }
  Required_Packages =c("RODBC","ggplot2","dplyr","stringr",
                       "AUC","anytime","plotROC","CHAID",
                       "RPostgreSQL","xlsx","MicrosoftML",
                       "ggmap","maps","mapdata",
                       "NbClust","factoextra","data.table");

  Install_And_Load(Required_Packages)
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

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

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

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

```
## AUC 0.3.0
```

```
## Type AUCNews() to see the change log and ?AUC to get an overview.
```

```
## Warning: package 'anytime' was built under R version 3.4.2
```

```
## Warning: package 'partykit' was built under R version 3.4.2
```

```
## Warning: package 'RPostgreSQL' was built under R version 3.4.2
```

```
## Warning: package 'DBI' was built under R version 3.4.2
```

```
## Warning: package 'xlsx' was built under R version 3.4.2
```

```
## Warning: package 'rJava' was built under R version 3.4.2
```

```
## Warning: package 'xlsxjars' was built under R version 3.4.2
```

```
## Google Maps API Terms of Service: http://developers.google.com/maps/terms.
```

```
## Please cite ggmap if you use it: see citation("ggmap") for details.
```

```
## Warning: package 'maps' was built under R version 3.4.2
```

```
## Warning: package 'mapdata' was built under R version 3.4.2
```

```
## Warning: package 'factoextra' was built under R version 3.4.2
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFC
Z
```

```
## Warning: package 'data.table' was built under R version 3.4.2
```

```
##
## Attaching package: 'data.table'
```

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

```
  options(scipen = 9999,digits = 10)

  Base<-fread('file:///D:/Disco C/Documentos/r/Aprendendo MicrosoftML/Amostra.csv',sep=';',de
c='.')
```

```
##
Read 6.9% of 1731585 rows
Read 15.6% of 1731585 rows
Read 25.4% of 1731585 rows
Read 34.1% of 1731585 rows
Read 43.3% of 1731585 rows
Read 53.1% of 1731585 rows
Read 62.4% of 1731585 rows
Read 69.9% of 1731585 rows
Read 78.5% of 1731585 rows
Read 86.0% of 1731585 rows
Read 95.9% of 1731585 rows
Read 1731585 rows and 21 (of 21) columns from 0.307 GB file in 00:00:14
```

```
  glimpse(Base)
```

```
## Observations: 1,731,585
## Variables: 21
## $ medallion          <chr> "1DDB1255470A78637646E29BD7053C5C", "1DDB12...
## $ hack_license       <chr> "A0EF5AB4B697801EC119EFA6B60ECF19", "A0EF5A...
## $ vendor_id          <chr> "VTS", "VTS", "VTS", "VTS", "VTS", "VTS", "...
## $ rate_code          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ store_and_fwd_flag <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
## $ pickup_datetime    <chr> "2013-05-05 16:52:00", "2013-05-07 08:43:00...
## $ dropoff_datetime   <chr> "2013-05-05 16:59:00", "2013-05-07 08:59:00...
## $ passenger_count    <int> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6...
## $ trip_time_in_secs  <int> 420, 960, 780, 180, 600, 2760, 480, 480, 11...
## $ trip_distance      <dbl> 1.22, 1.13, 2.47, 0.43, 2.04, 10.05, 1.27, ...
## $ pickup_longitude   <dbl> -73.9683, -73.9634, -73.9888, -73.9530, -73...
## $ pickup_latitude    <dbl> 40.7626, 40.7656, 40.7725, 40.7831, 40.8022...
## $ dropoff_longitude  <dbl> -73.9805, -73.9762, -73.9775, -73.9560, -73...
## $ dropoff_latitude   <dbl> 40.7708, 40.7574, 40.7535, 40.7777, 40.7777...
## $ payment_type       <chr> "CRD", "CRD", "CRD", "CRD", "CSH", "CRD", "...
## $ fare_amount        <dbl> 7.0, 10.5, 11.5, 4.5, 9.5, 38.5, 7.0, 8.0, ...
## $ surcharge          <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0...
## $ mta_tax            <dbl> 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5...
## $ tip_amount         <dbl> 1.40, 0.50, 3.45, 0.90, 0.00, 7.70, 1.40, 1...
## $ tolls_amount       <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 5.33, 0.00, 0...
## $ total_amount       <dbl> 8.90, 11.50, 15.45, 5.90, 10.00, 52.03, 8.9...
```

```
  Base$pickup_datetime<-as.POSIXct(Base$pickup_datetime,"%Y-%m-%d %H:%M:%S",tz='America/New_Y
ork')
  Base$dropoff_datetime<-as.POSIXct(Base$dropoff_datetime,"%Y-%m-%d %H:%M:%S",tz='America/New
_York')
  head(Base)
```

```
##                          medallion                    hack_license
## 1: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 2: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 3: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 4: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 5: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 6: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
##    vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1:       VTS         1                 NA 2013-05-05 16:52:00
## 2:       VTS         1                 NA 2013-05-07 08:43:00
## 3:       VTS         1                 NA 2013-06-07 06:01:00
## 4:       VTS         1                 NA 2013-06-10 09:23:00
## 5:       VTS         1                 NA 2013-06-11 13:53:00
## 6:       VTS         1                 NA 2013-06-12 14:48:00
##       dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1: 2013-05-05 16:59:00               6               420          1.22
## 2: 2013-05-07 08:59:00               6               960          1.13
## 3: 2013-06-07 06:14:00               6               780          2.47
## 4: 2013-06-10 09:26:00               6               180          0.43
## 5: 2013-06-11 14:03:00               6               600          2.04
## 6: 2013-06-12 15:34:00               6              2760         10.05
##    pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1:         -73.9683         40.7626          -73.9805          40.7708
## 2:         -73.9634         40.7656          -73.9762          40.7574
## 3:         -73.9888         40.7725          -73.9775          40.7535
## 4:         -73.9530         40.7831          -73.9560          40.7777
## 5:         -73.9683         40.8022          -73.9795          40.7777
## 6:         -73.9664         40.7530          -73.8618          40.7684
##    payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1:          CRD         7.0         0     0.5       1.40         0.00
## 2:          CRD        10.5         0     0.5       0.50         0.00
## 3:          CRD        11.5         0     0.5       3.45         0.00
## 4:          CRD         4.5         0     0.5       0.90         0.00
## 5:          CSH         9.5         0     0.5       0.00         0.00
## 6:          CRD        38.5         0     0.5       7.70         5.33
##    total_amount
## 1:         8.90
## 2:        11.50
## 3:        15.45
## 4:         5.90
## 5:        10.00
## 6:        52.03
```

```
  tail(Base)
```

```
##                          medallion                    hack_license
## 1: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 2: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 3: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 4: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 5: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
## 6: 1DDB1255470A78637646E29BD7053C5C A0EF5AB4B697801EC119EFA6B60ECF19
##    vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1:       VTS         1                 NA 2013-04-02 10:28:00
## 2:       VTS         1                 NA 2013-04-04 14:38:00
## 3:       VTS         1                 NA 2013-04-04 15:52:00
## 4:       VTS         1                 NA 2013-04-06 10:50:00
## 5:       VTS         1                 NA 2013-04-10 09:22:00
## 6:       VTS         1                 NA 2013-04-30 11:09:00
##       dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1: 2013-04-02 10:51:00               6              1380          3.11
## 2: 2013-04-04 14:47:00               6               540          1.54
## 3: 2013-04-04 16:10:00               6              1080          2.16
## 4: 2013-04-06 11:00:00               6               600          2.43
## 5: 2013-04-10 09:38:00               6               960          1.71
## 6: 2013-04-30 11:24:00               6               900          2.00
##    pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1:         -74.0019         40.7244          -73.9708          40.7558
## 2:         -73.9782         40.7631          -73.9885          40.7791
## 3:         -73.9961         40.7261          -73.9778          40.7506
## 4:         -73.9986         40.7279          -73.9719          40.7460
## 5:         -73.9779         40.7524          -73.9961          40.7466
## 6:         -73.9618         40.7675          -73.9813          40.7471
##    payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1:          CSH        16.5         0     0.5          0            0
## 2:          CSH         8.0         0     0.5          0            0
## 3:          CSH        13.0         0     0.5          0            0
## 4:          CSH        10.0         0     0.5          0            0
## 5:          CSH        11.5         0     0.5          0            0
## 6:          CRD        11.0         0     0.5          1            0
##    total_amount
## 1:         17.0
## 2:          8.5
## 3:         13.5
## 4:         10.5
## 5:         12.0
## 6:         12.5
```

```
glimpse(Base)
```

```
## Observations: 1,731,585
## Variables: 21
## $ medallion          <chr> "1DDB1255470A78637646E29BD7053C5C", "1DDB12...
## $ hack_license       <chr> "A0EF5AB4B697801EC119EFA6B60ECF19", "A0EF5A...
## $ vendor_id          <chr> "VTS", "VTS", "VTS", "VTS", "VTS", "VTS", "...
## $ rate_code          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ store_and_fwd_flag <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
## $ pickup_datetime    <dttm> 2013-05-05 16:52:00, 2013-05-07 08:43:00, ...
## $ dropoff_datetime   <dttm> 2013-05-05 16:59:00, 2013-05-07 08:59:00, ...
## $ passenger_count    <int> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6...
## $ trip_time_in_secs  <int> 420, 960, 780, 180, 600, 2760, 480, 480, 11...
## $ trip_distance      <dbl> 1.22, 1.13, 2.47, 0.43, 2.04, 10.05, 1.27, ...
## $ pickup_longitude   <dbl> -73.9683, -73.9634, -73.9888, -73.9530, -73...
## $ pickup_latitude    <dbl> 40.7626, 40.7656, 40.7725, 40.7831, 40.8022...
## $ dropoff_longitude  <dbl> -73.9805, -73.9762, -73.9775, -73.9560, -73...
## $ dropoff_latitude   <dbl> 40.7708, 40.7574, 40.7535, 40.7777, 40.7777...
## $ payment_type       <chr> "CRD", "CRD", "CRD", "CRD", "CSH", "CRD", "...
## $ fare_amount        <dbl> 7.0, 10.5, 11.5, 4.5, 9.5, 38.5, 7.0, 8.0, ...
## $ surcharge          <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0...
## $ mta_tax            <dbl> 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5...
## $ tip_amount         <dbl> 1.40, 0.50, 3.45, 0.90, 0.00, 7.70, 1.40, 1...
## $ tolls_amount       <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 5.33, 0.00, 0...
## $ total_amount       <dbl> 8.90, 11.50, 15.45, 5.90, 10.00, 52.03, 8.9...
```

summarizing the data

```
summary(Base)
```

```
##   medallion         hack_license        vendor_id
##  Length:1731585     Length:1731585     Length:1731585
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    rate_code          store_and_fwd_flag pickup_datetime
##  Min.   :  0.000000   Length:1731585     Min.   :2013-01-01 00:11:00
##  1st Qu.:  1.000000   Class :character   1st Qu.:2013-03-29 22:43:00
##  Median :  1.000000   Mode  :character   Median :2013-06-24 15:32:00
##  Mean   :  1.033811                      Mean   :2013-06-29 13:56:00
##  3rd Qu.:  1.000000                      3rd Qu.:2013-09-30 13:27:48
##  Max.   :210.000000                      Max.   :2013-12-31 23:58:25
##
##  dropoff_datetime              passenger_count    trip_time_in_secs
##  Min.   :2013-01-01 00:11:00   Min.   :0.000000   Min.   :    -10.000
##  1st Qu.:2013-03-29 22:57:00   1st Qu.:1.000000   1st Qu.:    360.000
##  Median :2013-06-24 15:44:00   Median :1.000000   Median :    600.000
##  Mean   :2013-06-29 14:08:34   Mean   :1.708797   Mean   :    800.021
##  3rd Qu.:2013-09-30 13:41:00   3rd Qu.:2.000000   3rd Qu.:    960.000
##  Max.   :2014-01-01 00:34:00   Max.   :9.000000   Max.   :4294905.000
##
##  trip_distance         pickup_longitude      pickup_latitude
##  Min.   :      0.000   Min.   :-1517.10000   Min.   :-180.00000
##  1st Qu.:      1.040   1st Qu.:  -73.99220   1st Qu.:  40.73450
##  Median :      1.800   Median :  -73.98190   Median :  40.75230
##  Mean   :      6.018   Mean   :  -72.44473   Mean   :  39.74892
##  3rd Qu.:      3.200   3rd Qu.:  -73.96680   3rd Qu.:  40.76710
##  Max.   :5331800.000   Max.   :   40.85680   Max.   :3210.36000
##
##  dropoff_longitude    dropoff_latitude      payment_type
##  Min.   :-740.17300   Min.   :-2497.70000   Length:1731585
##  1st Qu.: -73.99150   1st Qu.:   40.73350   Class :character
##  Median : -73.98020   Median :   40.75270   Mode  :character
##  Mean   : -72.40124   Mean   :   39.72285
##  3rd Qu.: -73.96360   3rd Qu.:   40.76780
##  Max.   :  40.90470   Max.   : 2302.58000
##  NA's   :20           NA's   :20
##   fare_amount          surcharge             mta_tax
##  Min.   :-79.00000   Min.   :-1.0000000   Min.   :-0.5000000
##  1st Qu.:  6.50000   1st Qu.: 0.0000000   1st Qu.: 0.5000000
##  Median :  9.50000   Median : 0.0000000   Median : 0.5000000
##  Mean   : 12.35104   Mean   : 0.3202457   Mean   : 0.4981956
##  3rd Qu.: 14.00000   3rd Qu.: 0.5000000   3rd Qu.: 0.5000000
##  Max.   :620.01000   Max.   :28.0000000   Max.   : 2.5000000
##
##    tip_amount          tolls_amount         total_amount
##  Min.   :  0.000000   Min.   : 0.0000000   Min.   :-79.00000
##  1st Qu.:  0.000000   1st Qu.: 0.0000000   1st Qu.:  8.00000
##  Median :  1.000000   Median : 0.0000000   Median : 11.00000
##  Mean   :  1.365235   Mean   : 0.2504534   Mean   : 14.78529
##  3rd Qu.:  2.000000   3rd Qu.: 0.0000000   3rd Qu.: 16.50000
##  Max.   :182.450000   Max.   :45.0000000   Max.   :620.01000
##
```

```
  Base%>%with(length(unique(medallion)))
```

```
## [1] 13524
```

```
  Base%>%with(length(unique(hack_license)))
```

```
## [1] 39051
```

```
  Base%>%group_by(store_and_fwd_flag)%>%summarise(Qnt=n())%>%mutate(`%`=Qnt/sum(Qnt))
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.2
```

```
## # A tibble: 3 x 3
##   store_and_fwd_flag    Qnt           `%`
##                <chr>  <int>         <dbl>
## 1                  N 849077 0.49034670548
## 2                  Y  20899 0.01206928912
## 3               <NA> 861609 0.49758400541
```

```
  Base%>%group_by(payment_type)%>%summarise(Qnt=n())%>%mutate(`%`=Qnt/sum(Qnt))
```

```
## # A tibble: 5 x 3
##   payment_type    Qnt             `%`
##          <chr>  <int>           <dbl>
## 1          CRD 931290 0.5378251717357
## 2          CSH 792780 0.4578348738295
## 3          DIS   1302 0.0007519122654
## 4          NOC   4169 0.0024076207636
## 5          UNK   2044 0.0011804214058
```

```
  Base%>%group_by(Year_Month=format(pickup_datetime,'%Y-%m'))%>%summarise(Mean_trip_Time=mean
(trip_time_in_secs),Qnt=n())%>%mutate(`%`=Qnt/sum(Qnt))
```

```
## # A tibble: 12 x 4
##    Year_Month Mean_trip_Time    Qnt           `%`
##         <chr>          <dbl>  <int>         <dbl>
##  1    2013-01    681.9155664 147311 0.08507292452
##  2    2013-02    701.4072688 141592 0.08177017010
##  3    2013-03    718.7444048 154150 0.08902248518
##  4    2013-04    744.9185972 150732 0.08704857111
##  5    2013-05    777.9325432 156174 0.09019135647
##  6    2013-06    778.4977133 144093 0.08321451156
##  7    2013-07    750.7912189 137249 0.07926206337
##  8    2013-08   1405.1602936 125189 0.07229734607
##  9    2013-09    784.3848638 144277 0.08332077259
## 10    2013-10    780.3568885 150257 0.08677425596
## 11    2013-11    775.5212709 143553 0.08290265855
## 12    2013-12    788.8799048 137008 0.07912288452
```

- studying extremes values

```
Base%>%filter(pickup_longitude==min(pickup_longitude))
```

```
##                          medallion                       hack_license
## 1 0C4726D4E2AF94BF8FE2D23EFDA20917 42CC3A49C66FA772567BBAB7F85F8835
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       VTS         1               <NA> 2013-02-05 10:17:00
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-02-05 10:59:00               1              2520         11.59
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1          -1517.1          40.769          -73.9827          40.7623
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          UNK        38.5         0     0.5       8.76          4.8
##   total_amount
## 1        52.56
```

```
Base%>%filter(pickup_longitude==max(pickup_longitude))
```

```
##                          medallion                       hack_license
## 1 5DE2E0B9BB72C1407330C14A0886EB85 D4DC28CC815084D8F55B56C4FA379C22
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       VTS         1               <NA> 2013-05-26 00:18:00
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-05-26 00:41:00               5              1380          5.43
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1          40.8568        -73.9337           40.8002         -73.9708
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CRD          21       0.5     0.5          0            0
##   total_amount
## 1           22
```

```
Base%>%filter(pickup_latitude==min(pickup_latitude))
```

```
##                          medallion                       hack_license
## 1 A485E6CAA482169A2A3837DEC32AEBAB 7B19DE6D4D54999531BEB27F758F71F6
## 2 A485E6CAA482169A2A3837DEC32AEBAB 7B19DE6D4D54999531BEB27F758F71F6
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       VTS         5               <NA> 2013-11-27 12:31:00
## 2       VTS         5               <NA> 2013-11-27 13:01:00
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-11-27 12:31:00               1                0             0
## 2 2013-11-27 13:01:00               1                0             0
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1             -180            -180              -180             -180
## 2             -180            -180              -180             -180
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CSH       99.99         0       0          0            0
## 2          CSH      333.38         0       0          0            0
##   total_amount
## 1        99.99
## 2       333.38
```

```
Base%>%filter(pickup_latitude==max(pickup_latitude))
```

```
##                          medallion                    hack_license
## 1 B602FCD21BD17941F53B6B61500A5DA2 5276DE98232EBA65D215BEBA6C0EBB31
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       VTS         1               <NA> 2013-03-18 06:51:00
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-03-18 06:52:00               4                60          0.43
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1                0         3210.36             -0.29          -2497.7
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CRD         3.5         0     0.5          2            0
##   total_amount
## 1            6
```

```
Base%>%filter(dropoff_latitude==min(dropoff_latitude))
```

```
##  [1] medallion          hack_license       vendor_id
##  [4] rate_code          store_and_fwd_flag pickup_datetime
##  [7] dropoff_datetime   passenger_count    trip_time_in_secs
## [10] trip_distance      pickup_longitude   pickup_latitude
## [13] dropoff_longitude  dropoff_latitude   payment_type
## [16] fare_amount        surcharge          mta_tax
## [19] tip_amount         tolls_amount       total_amount
## <0 rows> (or 0-length row.names)
```

```
Base%>%filter(dropoff_longitude==max(dropoff_longitude))
```

```
##  [1] medallion          hack_license       vendor_id
##  [4] rate_code          store_and_fwd_flag pickup_datetime
##  [7] dropoff_datetime   passenger_count    trip_time_in_secs
## [10] trip_distance      pickup_longitude   pickup_latitude
## [13] dropoff_longitude  dropoff_latitude   payment_type
## [16] fare_amount        surcharge          mta_tax
## [19] tip_amount         tolls_amount       total_amount
## <0 rows> (or 0-length row.names)
```

```
Base%>%filter(dropoff_latitude==min(dropoff_latitude))
```

```
##  [1] medallion          hack_license       vendor_id
##  [4] rate_code          store_and_fwd_flag pickup_datetime
##  [7] dropoff_datetime   passenger_count    trip_time_in_secs
## [10] trip_distance      pickup_longitude   pickup_latitude
## [13] dropoff_longitude  dropoff_latitude   payment_type
## [16] fare_amount        surcharge          mta_tax
## [19] tip_amount         tolls_amount       total_amount
## <0 rows> (or 0-length row.names)
```

```
Base%>%filter(dropoff_latitude==max(dropoff_latitude))
```

```
##  [1] medallion          hack_license       vendor_id
##  [4] rate_code          store_and_fwd_flag pickup_datetime
##  [7] dropoff_datetime   passenger_count    trip_time_in_secs
## [10] trip_distance      pickup_longitude   pickup_latitude
## [13] dropoff_longitude  dropoff_latitude   payment_type
## [16] fare_amount        surcharge          mta_tax
## [19] tip_amount         tolls_amount       total_amount
## <0 rows> (or 0-length row.names)
```

```
Base%>%filter(passenger_count==max(passenger_count))
```

```
##                          medallion                     hack_license
## 1 780189E2E3A40092B007D2D708791D22 D3471987DBEE5575F367BA8B74D84E8C
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       CMT         1                  N 2013-09-23 19:50:40
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-09-23 19:56:51               9               371           1.5
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1         -73.9687         40.7909          -73.9582          40.8105
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CRD           7         1     0.5        1.7            0
##   total_amount
## 1         10.2
```

```
Base%>%filter(trip_distance==max(trip_distance))
```

```
##                          medallion                     hack_license
## 1 3DCFC9097488D3F93982ED4A899C6E24 0B21B07994F53C937E2687A81488A3C5
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       CMT         1                  Y 2013-08-07 15:09:17
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-08-07 15:17:07               1               470       5331800
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1         -73.9805         40.7509          -73.9797          40.7444
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CRD           7         0     0.5        1.5            0
##   total_amount
## 1            9
```

```
Base%>%filter(trip_time_in_secs==max(trip_time_in_secs))
```

```
##                          medallion                    hack_license
## 1 99115D1EA0AE33939899D652DCC34089 B508465FAC4F54A40CFDBB2B69707F5A
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       CMT         1                  N 2013-08-04 03:27:21
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-08-04 09:48:39               1           4294905            16
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1         -74.0087         40.7115          -74.0106          40.7125
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CSH          44         0     0.5          0            0
##   total_amount
## 1         44.5
```

```
Base%>%filter(fare_amount==max(fare_amount))
```

```
##                          medallion                    hack_license
## 1 AC7A85219867AB060609BA214C124969 8FB6EB354A3D5986F098EEC2642D1340
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       CMT         5                  N 2013-08-21 00:11:07
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-08-21 00:13:04               1               117           0.4
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1         -74.0428         40.7201          -74.0358          40.7174
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          DIS      620.01         0       0          0            0
##   total_amount
## 1       620.01
```

```
Base%>%filter(surcharge==max(surcharge))
```

```
##                          medallion                    hack_license
## 1 039A16D739D799891C9211A55F731263 4626C9FE17E1AAD39BC797B3F6417C78
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       CMT         1                  N 2013-08-25 17:18:14
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-08-25 17:19:17               1                63           0.1
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1         -73.9933         40.7539           -73.988          40.7517
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          NOC           3        28     0.5          0            0
##   total_amount
## 1         31.5
```

```
Base%>%filter(tip_amount==max(tip_amount))
```

```
##                          medallion                      hack_license
## 1 848AC8C699E0BE2A7A04605006116745 C6C54D46C2272CD5C69BD2E6502B5D1C
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       VTS         1               <NA> 2013-01-23 21:39:00
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-01-23 21:46:00               1               420          1.43
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1          -73.995           40.74          -73.9895          40.7556
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CRD           7       0.5     0.5     182.45            0
##   total_amount
## 1       190.45
```

```
Base%>%filter(tolls_amount==max(tolls_amount))
```

```
##                          medallion                      hack_license
## 1 77D0801A31D1326C3D71CB8E5611D30C AE103F4D67534DCF2A19B6EC0E8CAEBD
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       VTS         1               <NA> 2013-08-26 14:07:00
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-08-26 14:20:00               3               780          1.13
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1         -73.8874          40.7511          -73.8944          40.7578
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          CRD         9.5         0     0.5          0           45
##   total_amount
## 1           55
```

```
Base%>%filter(total_amount==max(total_amount))
```

```
##                          medallion                      hack_license
## 1 AC7A85219867AB060609BA214C124969 8FB6EB354A3D5986F098EEC2642D1340
##   vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 1       CMT         5                  N 2013-08-21 00:11:07
##      dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 1 2013-08-21 00:13:04               1               117           0.4
##   pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 1         -74.0428          40.7201          -74.0358          40.7174
##   payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 1          DIS      620.01         0       0          0            0
##   total_amount
## 1       620.01
```

- Excluding taxi trips that had began or had ended outside NYC

```
#########################
#NYC extreme coordinates#
#North:40.917577########
#South:40.477399########
#East: -73.700272#######
#West:-74.259090########
#########################
Base<-Base%>%filter(!((pickup_latitude<0|dropoff_latitude<0)|(pickup_longitude>0|dropoff_long
itude>0)))
Base<-Base%>%filter((pickup_longitude>-74.259090 & pickup_longitude< -73.700272) &
                      (dropoff_longitude>-74.259090 & dropoff_longitude< -73.700272) &
                      (dropoff_latitude>40.477399 & dropoff_latitude< 40.917577) &
                      (pickup_latitude>40.477399 & pickup_latitude< 40.917577))
```

- Plotting the pickpu and dropoff coordinates of the taxi trips

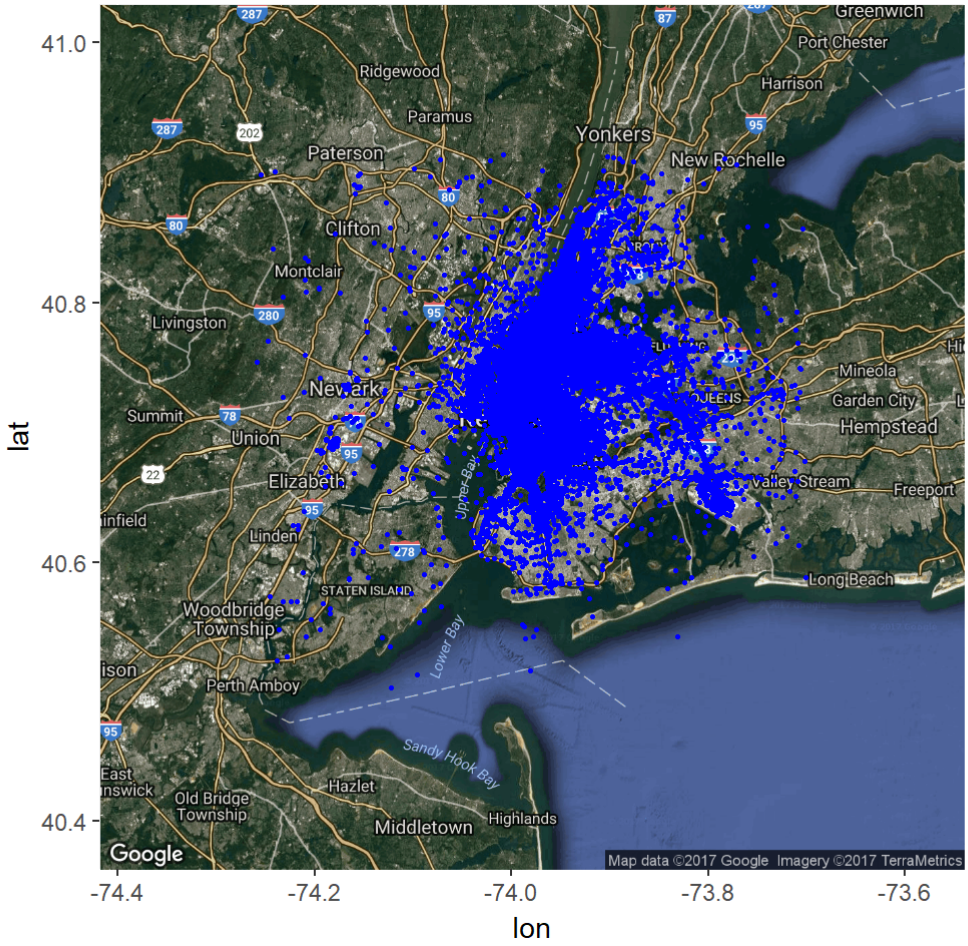
```
NYC_Map<-get_map(c(-74.259090,40.477399,-73.700272,40.917577),zoom = 10,maptype = 'hybrid')
```

```
## Warning: bounding box given to google - spatial extent only approximate.
```

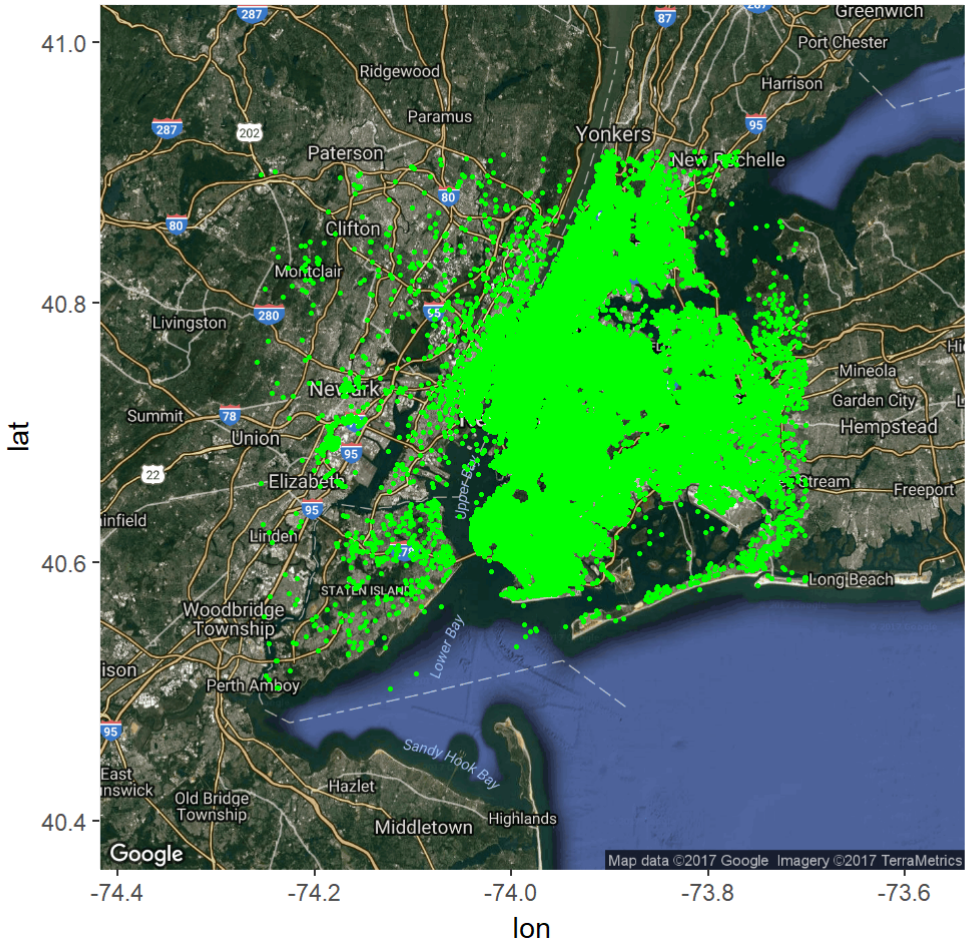
```
## converting bounding box to center/zoom specification. (experimental)
```

```
## Source : https://maps.googleapis.com/maps/api/staticmap?center=40.697488,-73.979681&zoom=1
0&size=640x640&scale=2&maptype=hybrid&language=en-EN
```

```
#############
###Pickup###
#############
NYC_Map%>%
    ggmap()+
    geom_point(data=Base,aes(x=pickup_longitude,y=pickup_latitude),color='blue',size=.5)
```




```
##############
###Dropoff###
##############
NYC_Map%>%
    ggmap()+
    geom_point(data=Base,aes(x=dropoff_longitude,y=dropoff_latitude),color='green',size=.5)
```



I identified more than 90 differents cluster, by the pickup and dropoff coordinates, but it didn't bring any practical improvment in the analysis, so i'm going to supress this part.

Studying the realations between the tipped, or not, trips and the others variable

```
Base<-Base%>%filter(trip_time_in_secs>0)
Base<-Base%>%mutate(Flag_Tip=0)
Base$Flag_Tip[Base$tip_amount>0]<-1
Base$Flag_Tip_Label<-'Not Tipped'
Base$Flag_Tip_Label[Base$Flag_Tip==1]<-'Tipped'

Base$store_and_fwd_flag_Label<-'Maybe'
Base$store_and_fwd_flag_Label[Base$store_and_fwd_flag=='Y']<-'Yes'
Base$store_and_fwd_flag_Label[Base$store_and_fwd_flag=='N']<-'No'
Base$store_and_fwd_flag_Label<-factor(Base$store_and_fwd_flag_Label,levels = c('Yes','Maybe',
'No'))

Base<-Base%>%mutate(trip_distance_01=(trip_distance-min(trip_distance))/(max(trip_distance)-m
in(trip_distance)),
                    trip_time_in_secs_01=(trip_time_in_secs-min(trip_time_in_secs))/(max(tr
ip_time_in_secs)-min(trip_time_in_secs)))

Base%>%select(Flag_Tip_Label,store_and_fwd_flag_Label)%>%
   with(table(Flag_Tip_Label,store_and_fwd_flag_Label))%>%
   prop.table(1)%>%addmargins(2)
```

```
##               store_and_fwd_flag_Label
## Flag_Tip_Label           Yes         Maybe            No           Sum
##     Not Tipped 0.01238479798 0.49399643141 0.49361877061 1.00000000000
##     Tipped     0.01178600433 0.49868029237 0.48953370330 1.00000000000
```

```
Base%>%select(Flag_Tip_Label,passenger_count)%>%
  with(table(Flag_Tip_Label,passenger_count))%>%
  prop.table(1)%>%addmargins(2)
```

```
##               passenger_count
## Flag_Tip_Label                 0                 1                 2
##     Not Tipped 0.000000000000000 0.686359510799242 0.144655232830983
##     Tipped     0.000004511820971 0.719942248691572 0.128773010286952
##               passenger_count
## Flag_Tip_Label                 3                 4                 5
##     Not Tipped 0.047738802357099 0.025423382528297 0.056953726026278
##     Tipped     0.037356749684173 0.016233531853456 0.058106614329543
##               passenger_count
## Flag_Tip_Label                 6                 8                 9
##     Not Tipped 0.038869345458103 0.000000000000000 0.000000000000000
##     Tipped     0.039579949467605 0.000002255910485 0.000001127955243
##               passenger_count
## Flag_Tip_Label               Sum
##     Not Tipped 1.000000000000000
##     Tipped     1.000000000000000
```

```
Base%>%select(Flag_Tip_Label,mta_tax)%>%
  with(table(Flag_Tip_Label,mta_tax))%>%
  prop.table(1)%>%addmargins(2)
```

```
##               mta_tax
## Flag_Tip_Label            -0.5                 0             0.5
##     Not Tipped 0.00004457635744 0.00232416174779 0.99763126189477
##     Tipped     0.00000000000000 0.00332634001083 0.99667365998917
##               mta_tax
## Flag_Tip_Label             Sum
##     Not Tipped 1.00000000000000
##     Tipped     1.00000000000000
```

```
Base%>%group_by(store_and_fwd_flag_Label)%>%summarise(Qnt=n(),Tip_Ratio=mean(Flag_Tip))%>%mut
ate(Freq=Qnt/sum(Qnt))
```

```
## # A tibble: 3 x 4
##   store_and_fwd_flag_Label    Qnt    Tip_Ratio          Freq
##                     <fctr>  <int>        <dbl>         <dbl>
## 1                      Yes  20451 0.5109285610 0.01207144767
## 2                    Maybe 841063 0.5256562231 0.49644750830
## 3                       No 832649 0.5212292334 0.49148104403
```

```
Base%>%group_by(payment_type)%>%summarise(Qnt=n(),Tip_Ratio=mean(Flag_Tip))%>%mutate(Freq=Qnt
/sum(Qnt))
```

```
## # A tibble: 5 x 4
##   payment_type    Qnt        Tip_Ratio            Freq
##          <chr>  <int>            <dbl>           <dbl>
## 1          CRD 911995 0.96992308071864 0.5383159707773
## 2          CSH 774891 0.00009033528587 0.4573886928235
## 3          DIS   1279 0.00781860828772 0.0007549450673
## 4          NOC   4057 0.00295785062854 0.0023946928365
## 5          UNK   1941 0.98042246264812 0.0011456984954
```

```
Base%>%group_by(Flag_Tip_Label)%>%summarise(Passenger_Ratio=mean(passenger_count))
```

```
## # A tibble: 2 x 2
##   Flag_Tip_Label Passenger_Ratio
##            <chr>           <dbl>
## 1     Not Tipped     1.738564617
## 2         Tipped     1.682533613
```

```
Base%>%group_by(Flag_Tip_Label)%>%
  summarise(MinDistance=min(trip_distance),
            Q1distance=quantile(trip_distance,.25),
            Q2distance=quantile(trip_distance,.5),
            meanDistance=mean(trip_distance),
            Q3distance=quantile(trip_distance,.75),
            Maxdistance=max(trip_distance),
            VarDistance=var(trip_distance),
            DPDistance=sd(trip_distance),
            CVDistance=sd(trip_distance)/mean(trip_distance))
```

```
## # A tibble: 2 x 10
##   Flag_Tip_Label MinDistance Q1distance Q2distance meanDistance Q3distance
##            <chr>       <dbl>      <dbl>      <dbl>        <dbl>      <dbl>
## 1     Not Tipped           0       0.95       1.60  2.677764025        2.9
## 2         Tipped           0       1.19       1.99  9.190742578        3.5
## # ... with 4 more variables: Maxdistance <dbl>, VarDistance <dbl>,
## #   DPDistance <dbl>, CVDistance <dbl>
```

```
Base%>%group_by(Flag_Tip_Label)%>%
  summarise(MinTime=min(trip_time_in_secs),
            Q1Time=quantile(trip_time_in_secs,.25),
            Q2Time=quantile(trip_time_in_secs,.5),
            meanTime=mean(trip_time_in_secs),
            Q3Time=quantile(trip_time_in_secs,.75),
            MaxTime=max(trip_time_in_secs),
            VarTime=var(trip_time_in_secs),
            DPTime=sd(trip_time_in_secs),
            CVTime=sd(trip_time_in_secs)/mean(trip_time_in_secs))
```

```
## # A tibble: 2 x 10
##   Flag_Tip_Label MinTime Q1Time Q2Time    meanTime Q3Time MaxTime
##            <chr>   <dbl>  <dbl>  <dbl>       <dbl>  <dbl>   <dbl>
## 1     Not Tipped       1    347    559 778.8271416    900 4294905
## 2         Tipped       1    420    660 819.1549529   1020 4293588
## # ... with 3 more variables: VarTime <dbl>, DPTime <dbl>, CVTime <dbl>
```

```
Base%>%group_by(Flag_Tip_Label)%>%
  summarise(MinFare=min(fare_amount),
            Q1Fare=quantile(fare_amount,.25),
            Q2Fare=quantile(fare_amount,.5),
            meanFare=mean(fare_amount),
            Q3Fare=quantile(fare_amount,.75),
            MaxFare=max(fare_amount),
            VarFare=var(fare_amount),
            DPFare=sd(fare_amount),
            CVFare=sd(fare_amount)/mean(fare_amount))
```

```
## # A tibble: 2 x 10
##   Flag_Tip_Label MinFare Q1Fare Q2Fare    meanFare Q3Fare MaxFare
##            <chr>   <dbl>  <dbl>  <dbl>       <dbl>  <dbl>   <dbl>
## 1     Not Tipped     -52      6    8.5 11.48785553     13  620.01
## 2         Tipped       0      7   10.0 12.98928961     15  400.00
## # ... with 3 more variables: VarFare <dbl>, DPFare <dbl>, CVFare <dbl>
```

```
Base%>%group_by(Flag_Tip_Label)%>%
  summarise(MinSurcharge=min(surcharge),
            Q1Surcharge=quantile(surcharge,.25),
            Q2Surcharge=quantile(surcharge,.5),
            meanSurcharge=mean(surcharge),
            Q3Surcharge=quantile(surcharge,.75),
            MaxSurcharge=max(surcharge),
            VarSurcharge=var(surcharge),
            DPSurcharge=sd(surcharge),
            CVSurcharge=sd(surcharge)/mean(surcharge))
```

```
## # A tibble: 2 x 10
##   Flag_Tip_Label MinSurcharge Q1Surcharge Q2Surcharge meanSurcharge
##            <chr>        <dbl>       <dbl>       <dbl>         <dbl>
## 1     Not Tipped           -1           0         0.0  0.3132306344
## 2         Tipped            0           0         0.5  0.3274646950
## # ... with 5 more variables: Q3Surcharge <dbl>, MaxSurcharge <dbl>,
## #   VarSurcharge <dbl>, DPSurcharge <dbl>, CVSurcharge <dbl>
```

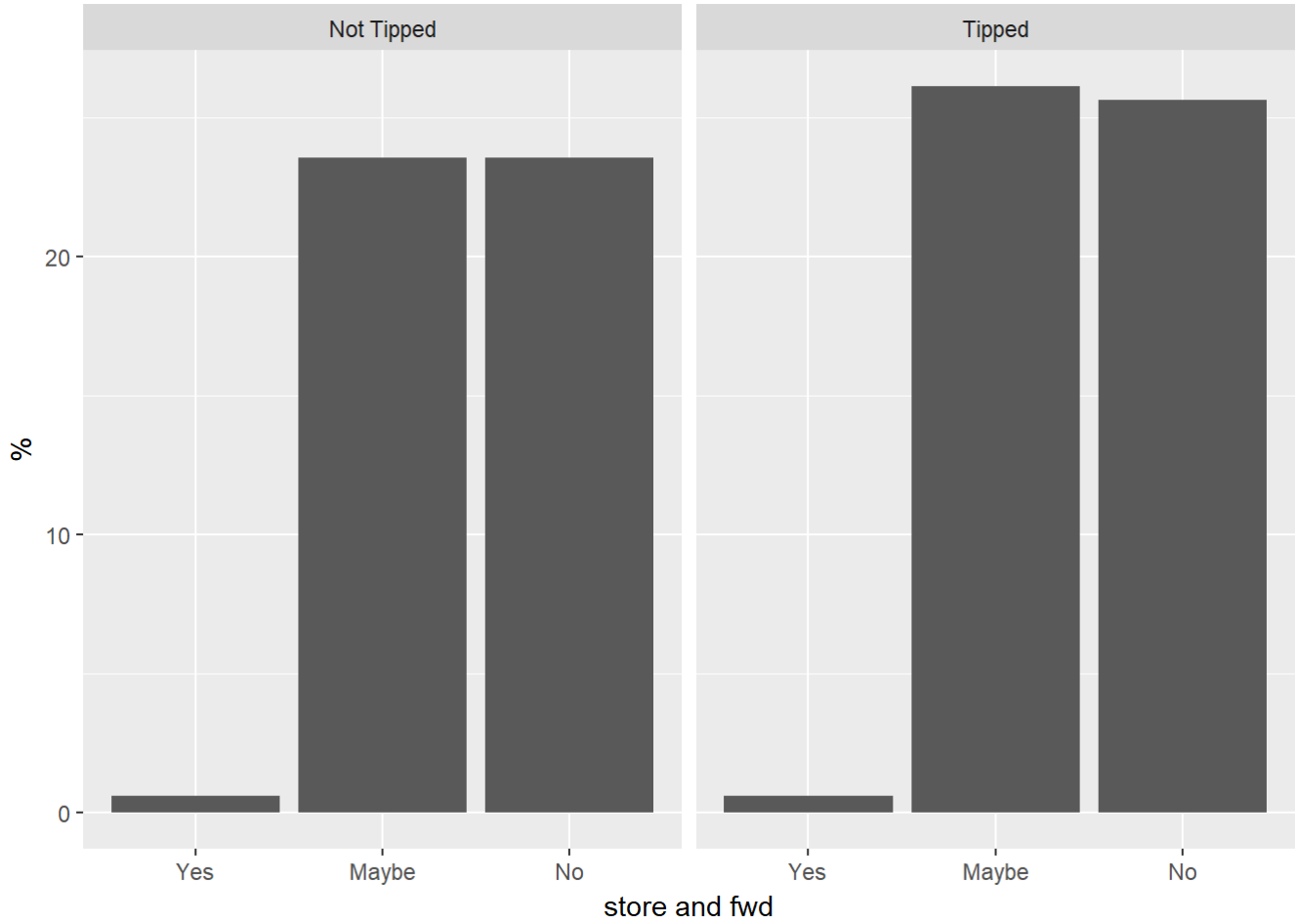
```
Base%>%group_by(Flag_Tip_Label)%>%
  summarise(MinToll=min(tolls_amount),
            Q1Toll=quantile(tolls_amount,.25),
            Q2Toll=quantile(tolls_amount,.5),
            meanToll=mean(tolls_amount),
            Q3Toll=quantile(tolls_amount,.75),
            MaxToll=max(tolls_amount),
            VarToll=var(tolls_amount),
            DPToll=sd(tolls_amount),
            CVToll=sd(tolls_amount)/mean(tolls_amount))
```

```
## # A tibble: 2 x 10
##   Flag_Tip_Label MinToll Q1Toll Q2Toll     meanToll Q3Toll MaxToll
##            <chr>   <dbl>  <dbl>  <dbl>        <dbl>  <dbl>   <dbl>
## 1     Not Tipped       0      0      0 0.1894105148      0   45.00
## 2          Tipped       0      0      0 0.3027054006      0   24.31
## # ... with 3 more variables: VarToll <dbl>, DPToll <dbl>, CVToll <dbl>
```

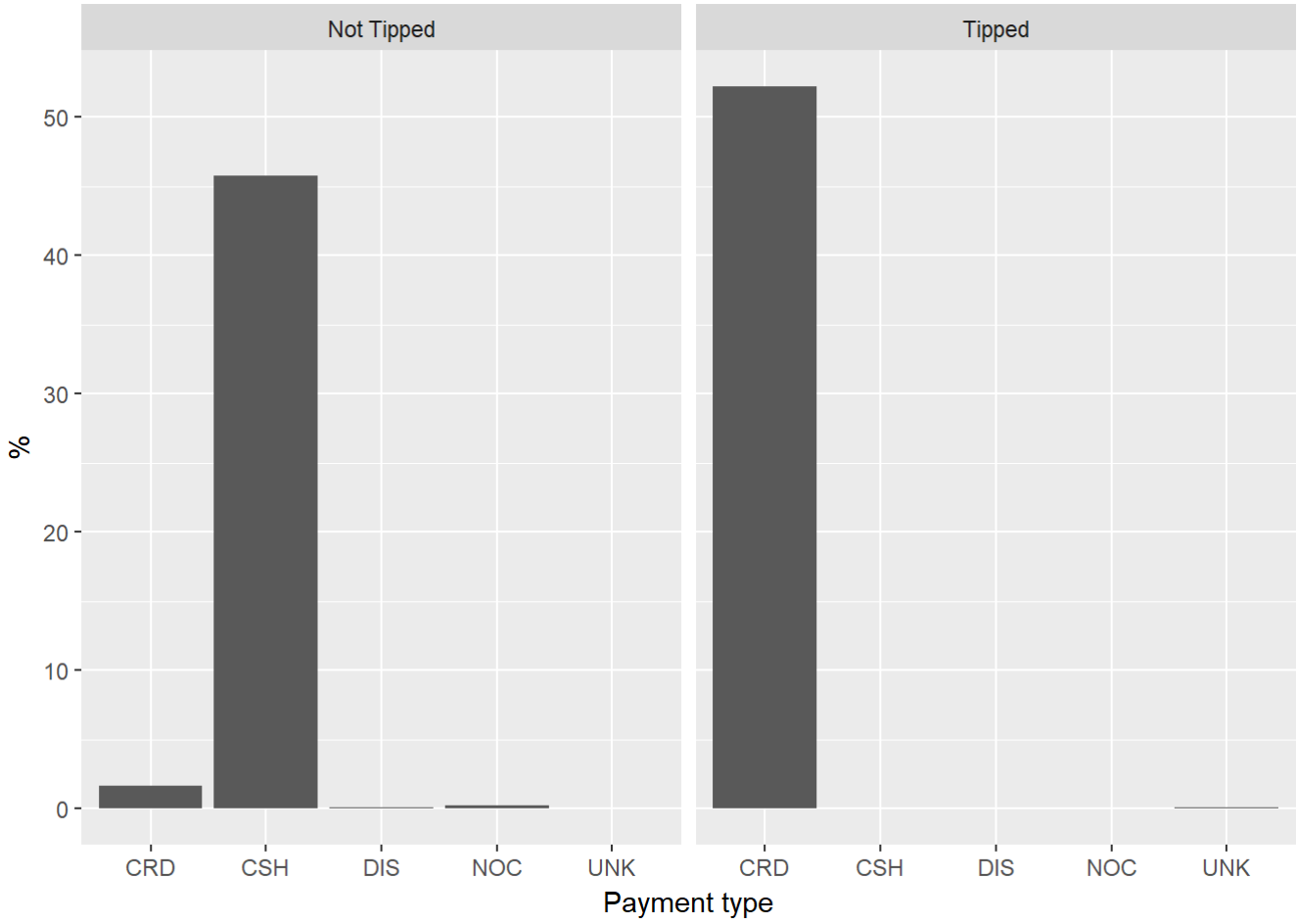
```
Base%>%group_by(Flag_Tip_Label)%>%
  summarise(MinTotal=min(total_amount),
            Q1Total=quantile(total_amount,.25),
            Q2Total=quantile(total_amount,.5),
            meanTotal=mean(total_amount),
            Q3Total=quantile(total_amount,.75),
            MaxTotal=max(total_amount),
            VarTotal=var(total_amount),
            DPTotal=sd(total_amount),
            CVTotal=sd(total_amount)/mean(total_amount))
```

```
## # A tibble: 2 x 10
##   Flag_Tip_Label MinTotal Q1Total Q2Total   meanTotal Q3Total MaxTotal
##            <chr>    <dbl>   <dbl>   <dbl>       <dbl>   <dbl>    <dbl>
## 1     Not Tipped   -52.50     7.0     9.5 12.48940600    14.0   620.01
## 2          Tipped     3.01     9.5    12.6 16.70995896    18.6   480.00
## # ... with 3 more variables: VarTotal <dbl>, DPTotal <dbl>, CVTotal <dbl>
```

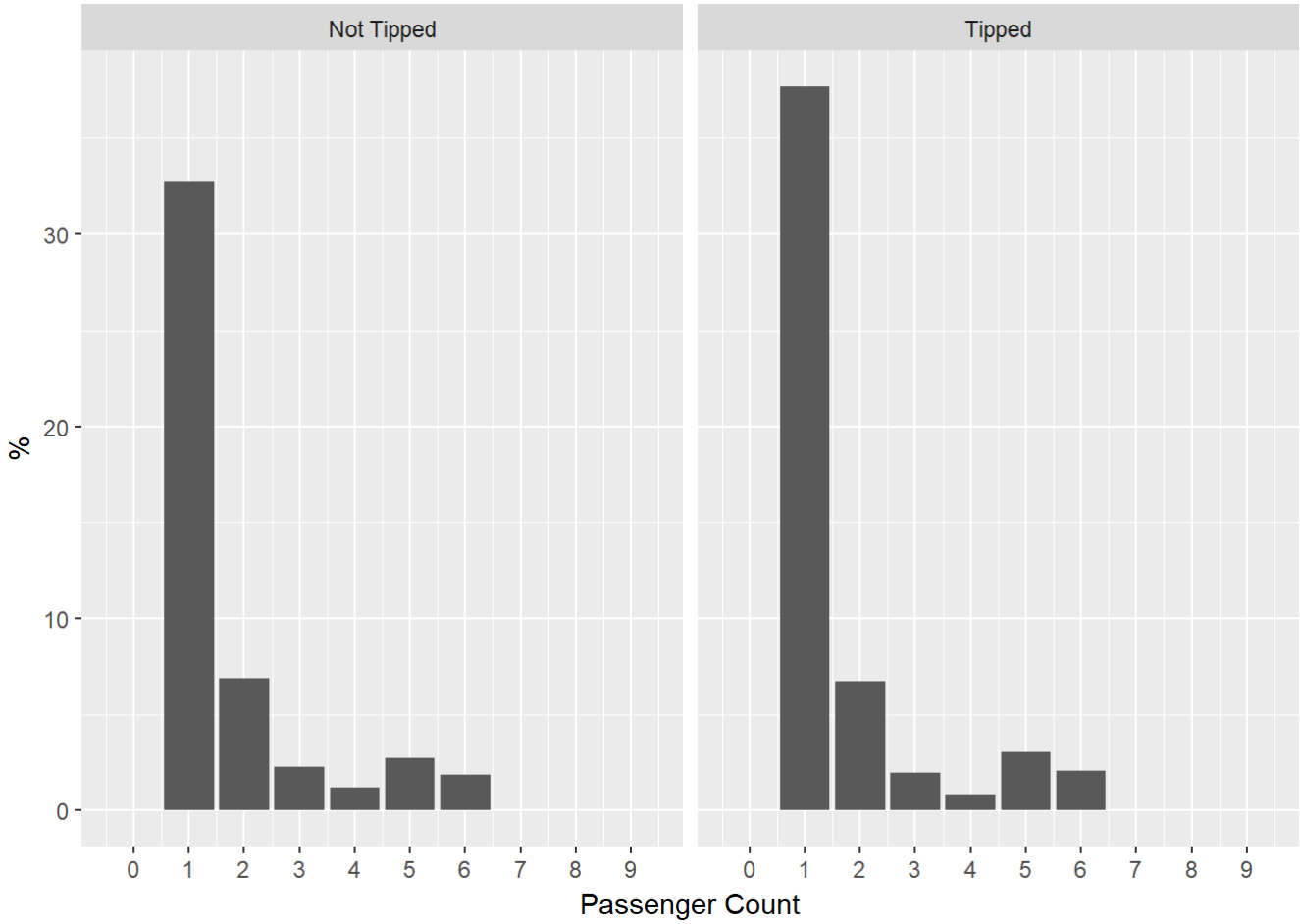
```
Base%>%ggplot(aes(x=store_and_fwd_flag_Label))+
  geom_bar(aes(y=100*(..count../sum(..count..))))+
  facet_grid(.~Flag_Tip_Label)+
  xlab('store and fwd')+
  ylab('%')
```



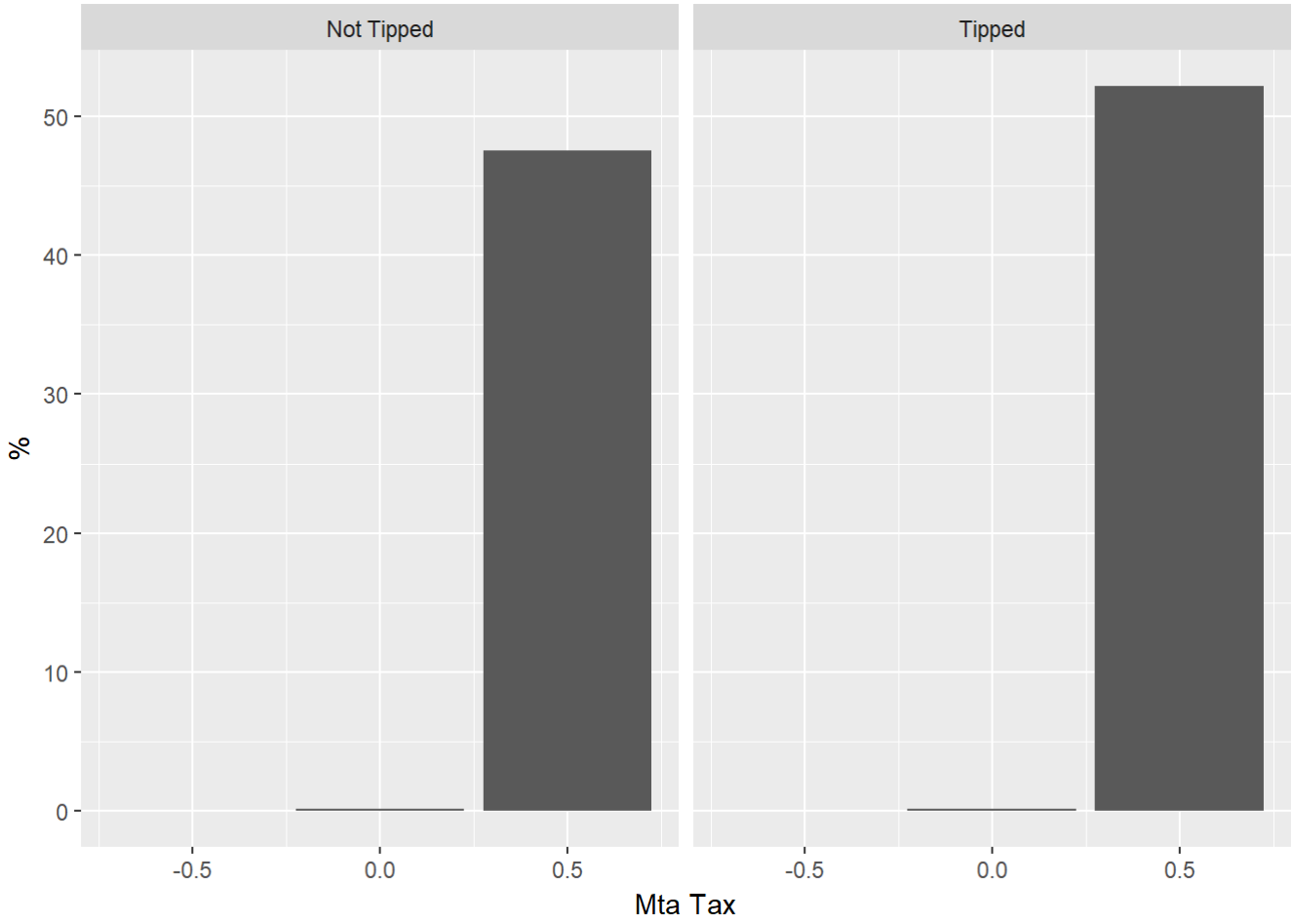

```
Base%>%ggplot(aes(x=payment_type))+
  geom_bar(aes(y=100*(..count../sum(..count..))))+
  facet_grid(.~Flag_Tip_Label)+
  xlab('Payment type')+
  ylab('%')
```




```
Base%>%ggplot(aes(x=passenger_count))+
  geom_bar(aes(y=100*(..count../sum(..count..))))+
  scale_x_continuous(breaks=c(0,1,2,3,4,5,6,7,8,9))+
  facet_grid(.~Flag_Tip_Label)+
  xlab('Passenger Count')+
  ylab('%')
```

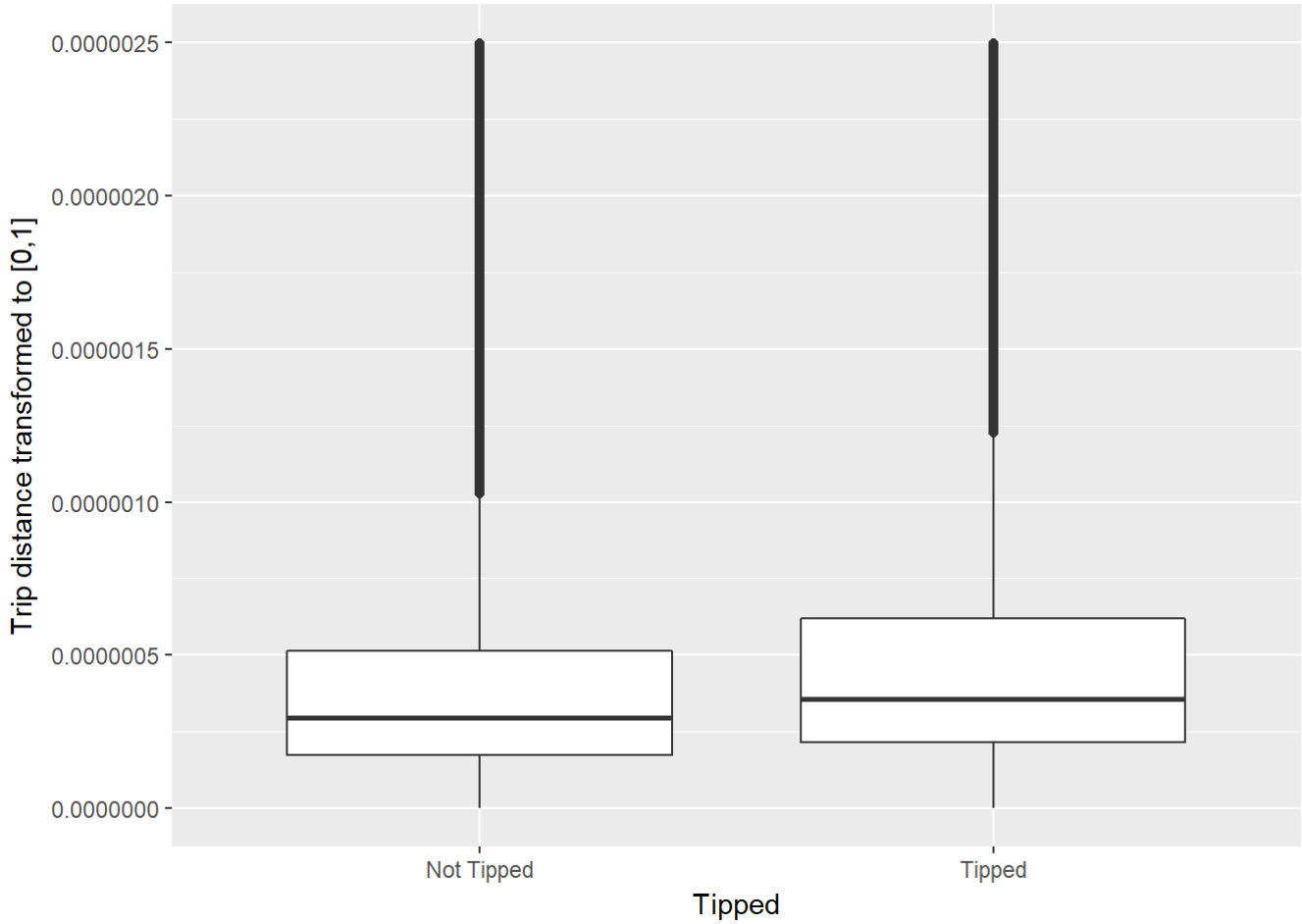



```
Base%>%ggplot(aes(x=mta_tax))+
  geom_bar(aes(y=100*(..count../sum(..count..))))+
  scale_x_continuous(breaks=c(-0.5,0,0.5))+
  facet_grid(.~Flag_Tip_Label)+
  xlab('Mta Tax')+
  ylab('%')
```



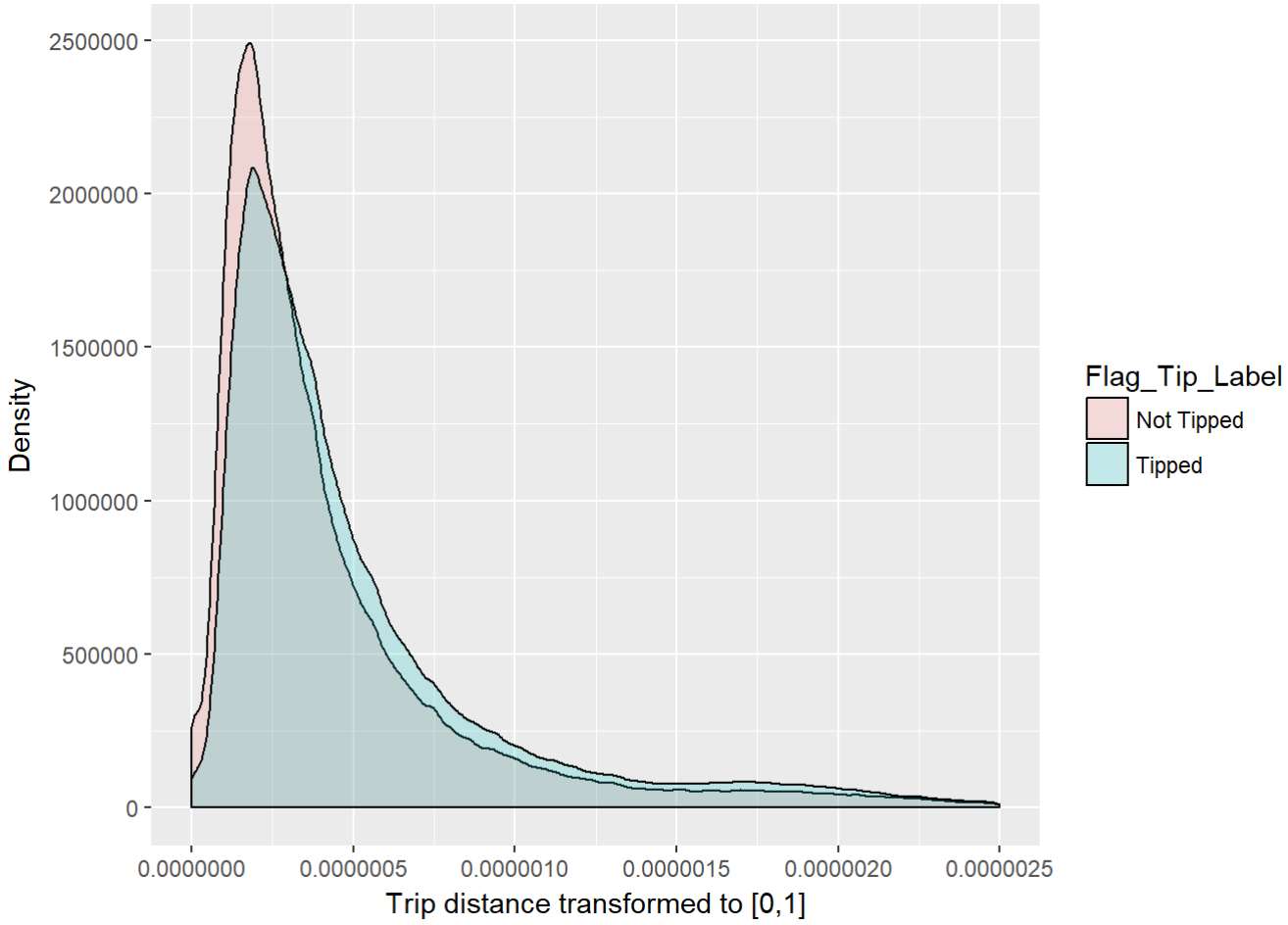

```
Base%>%ggplot(aes(x=Flag_Tip_Label,y=trip_distance_01))+
  geom_boxplot()+
  xlab('Tipped')+
  ylab('Trip distance transformed to [0,1]')+
  scale_y_continuous(limits = c(0,.0000025))
```

```
## Warning: Removed 43452 rows containing non-finite values (stat_boxplot).
```



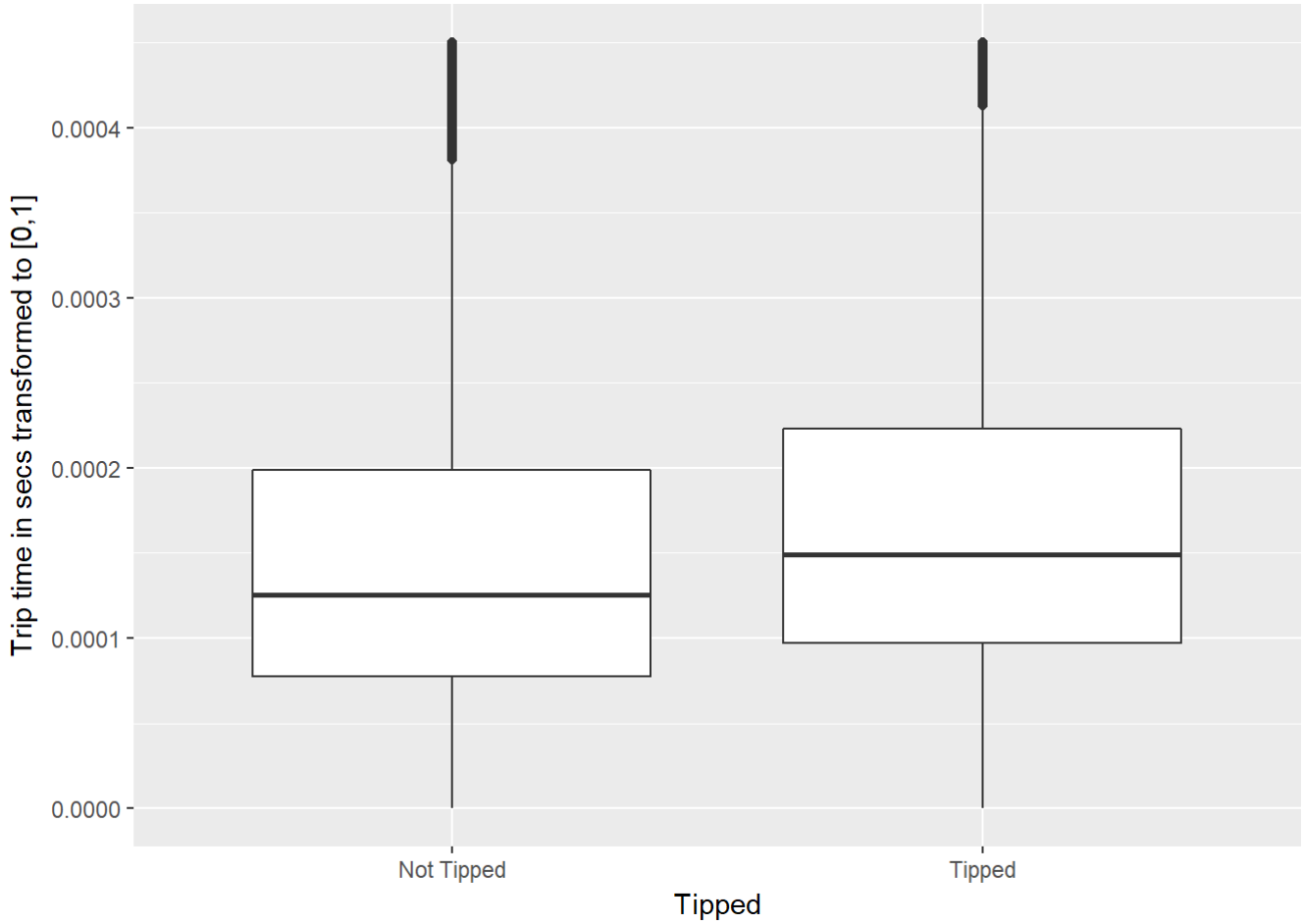

```
Base%>%ggplot(aes(x=trip_distance_01, fill=Flag_Tip_Label))+
  geom_density(alpha=.2)+
  scale_x_continuous(limits = c(0,.0000025))+
  xlab('Trip distance transformed to [0,1]')+
  ylab('Density')
```

```
## Warning: Removed 43452 rows containing non-finite values (stat_density).
```



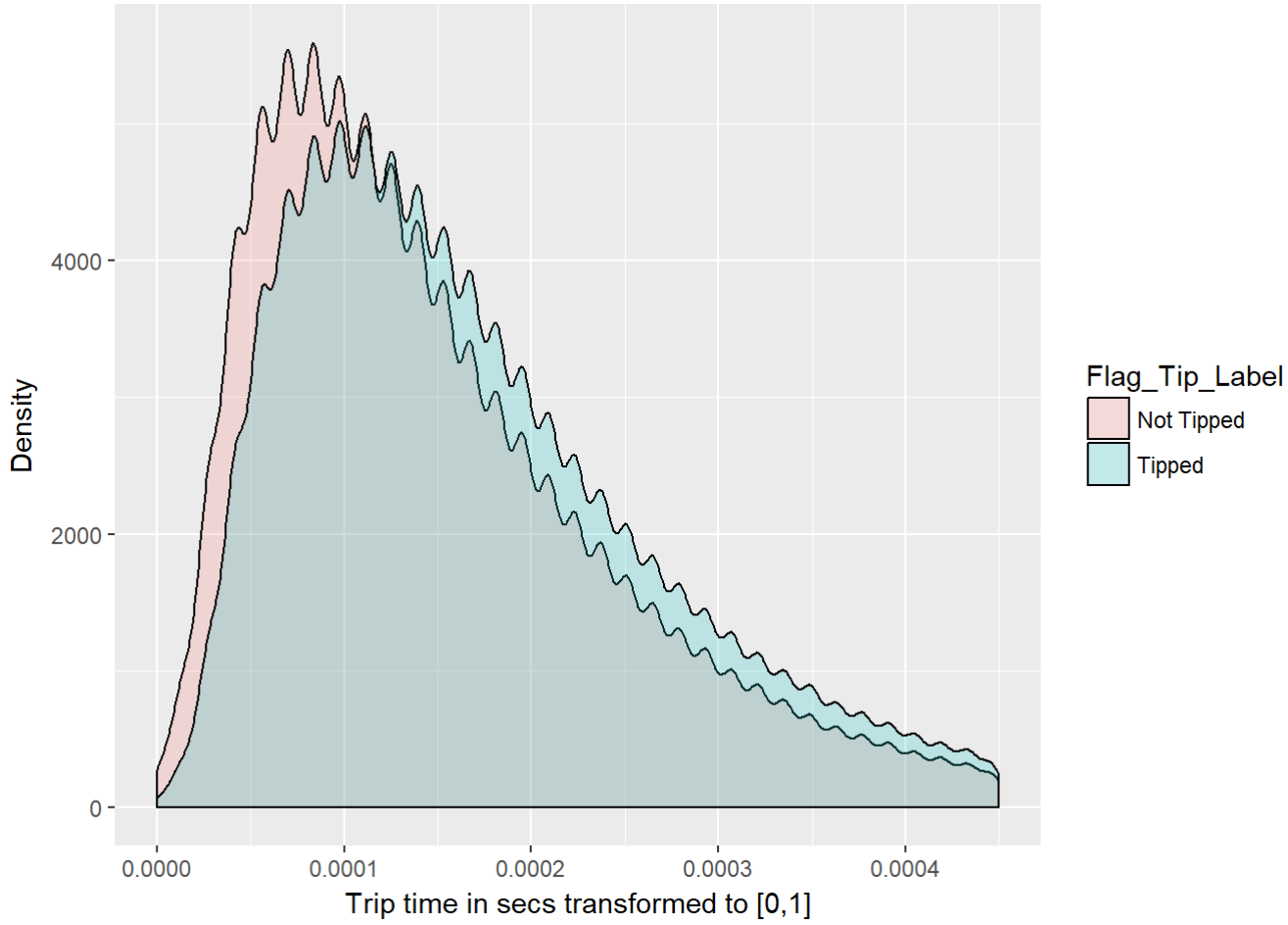

```
Base%>%ggplot(aes(x=Flag_Tip_Label,y=trip_time_in_secs_01))+
  geom_boxplot()+
  xlab('Tipped')+
  ylab('Trip time in secs transformed to [0,1]')+
  scale_y_continuous(limits = c(0,.00045))
```

```
## Warning: Removed 65917 rows containing non-finite values (stat_boxplot).
```



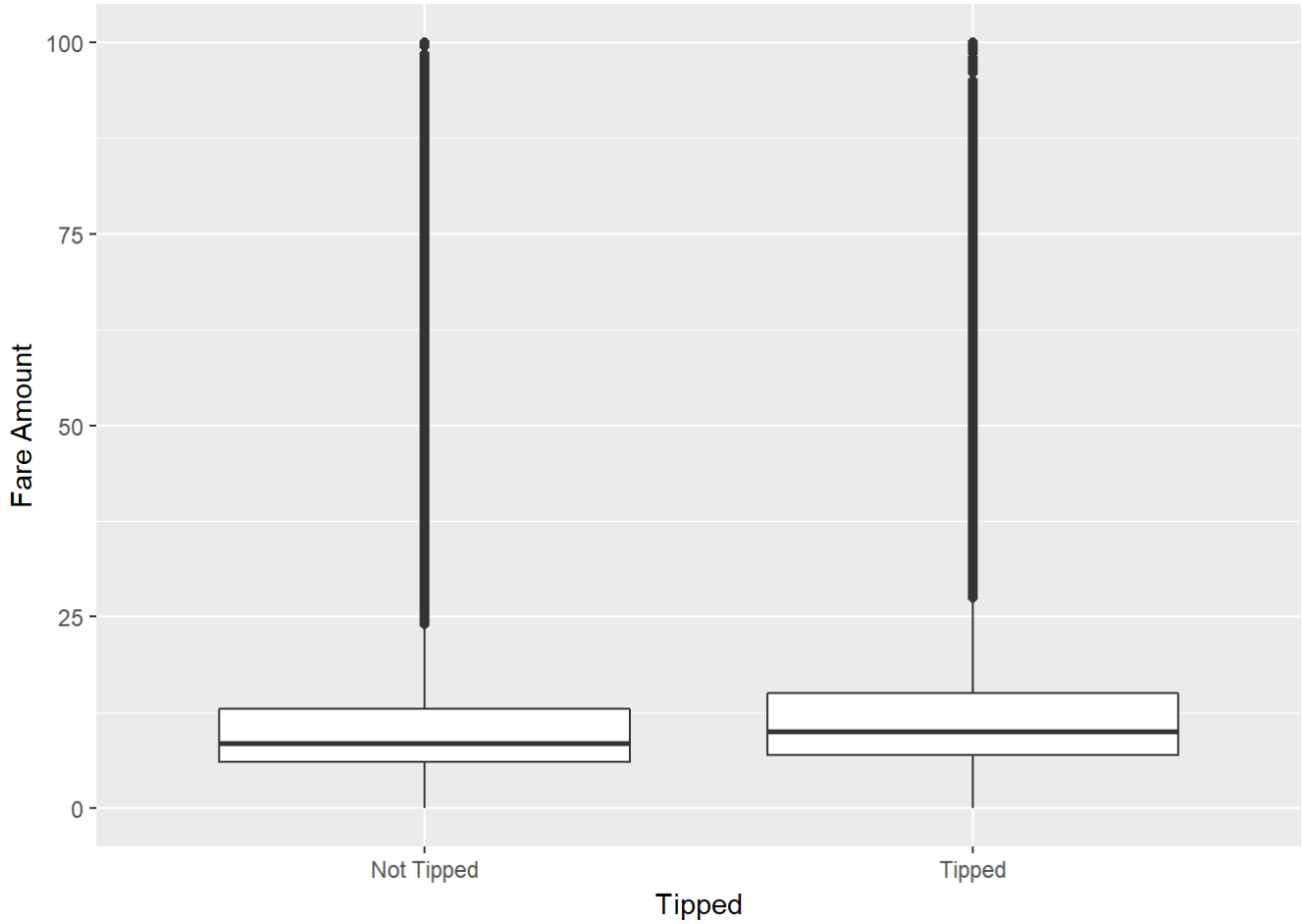

```
Base%>%ggplot(aes(x=trip_time_in_secs_01,fill=Flag_Tip_Label))+
  geom_density(alpha=.2)+
  ylab('Density')+
  xlab('Trip time in secs transformed to [0,1]')+
  scale_x_continuous(limits = c(0,.00045))
```

```
## Warning: Removed 65917 rows containing non-finite values (stat_density).
```



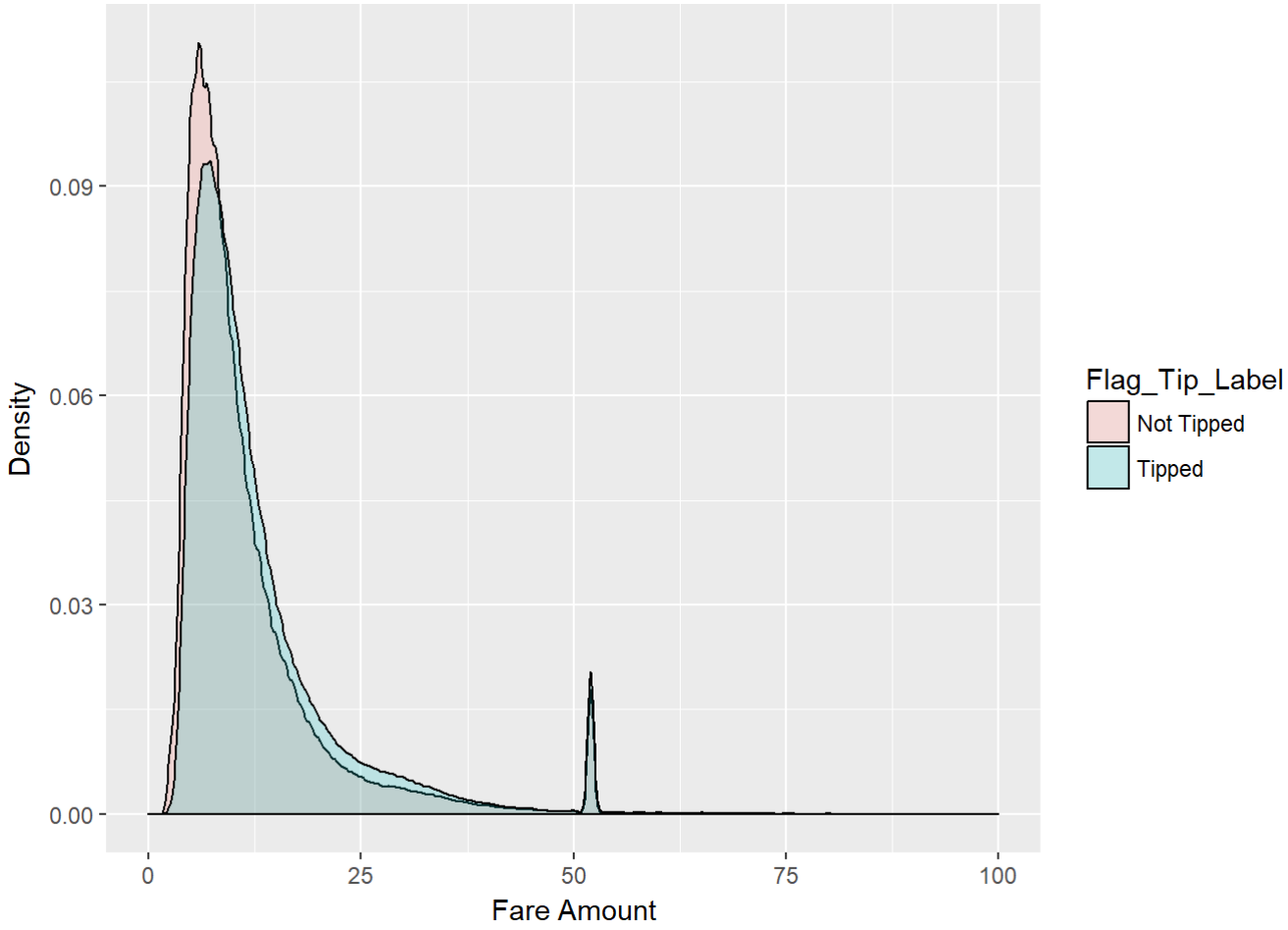

```
Base%>%ggplot(aes(x=Flag_Tip_Label,y=fare_amount))+
  geom_boxplot()+
  xlab('Tipped')+
  ylab('Fare Amount')+
  scale_y_continuous(limits = c(0,100))
```

```
## Warning: Removed 305 rows containing non-finite values (stat_boxplot).
```



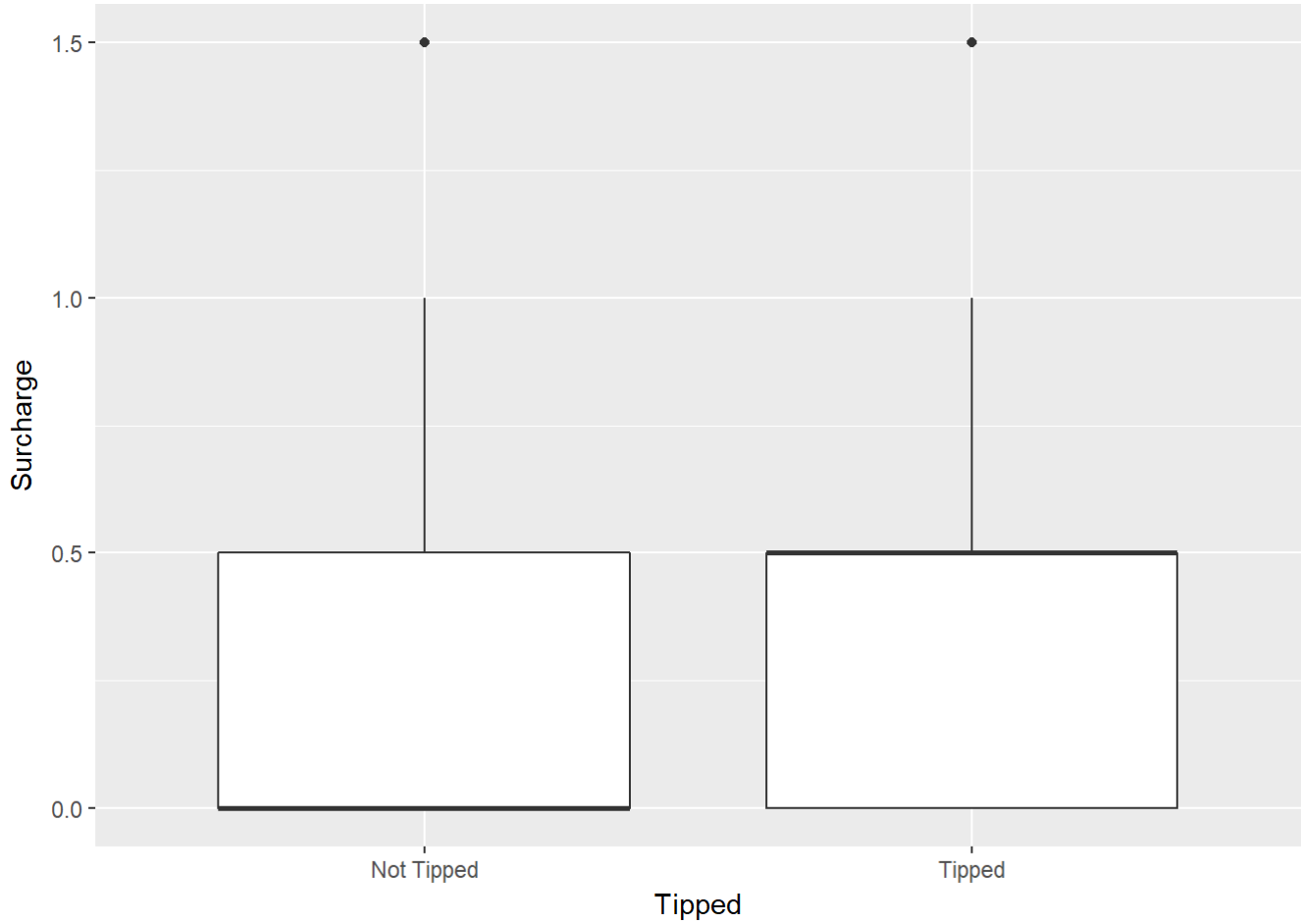

```
Base%>%ggplot(aes(x=fare_amount,fill=Flag_Tip_Label))+
  geom_density(alpha=.2)+
  ylab('Density')+
  xlab('Fare Amount')+
  scale_x_continuous(limits = c(0,100))
```

```
## Warning: Removed 305 rows containing non-finite values (stat_density).
```



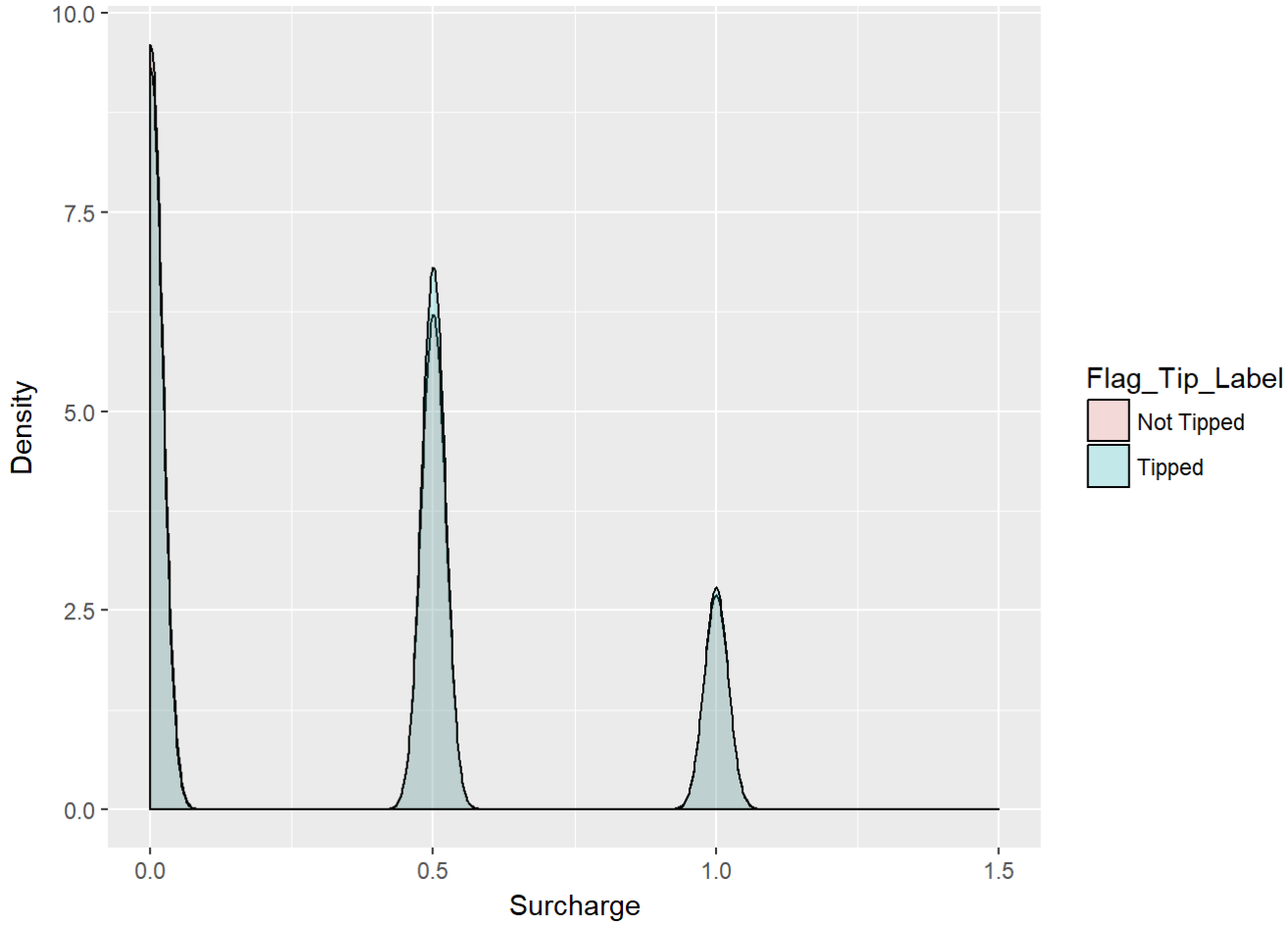

```
Base%>%ggplot(aes(x=Flag_Tip_Label,y=surcharge))+
  geom_boxplot()+
  xlab('Tipped')+
  ylab('Surcharge')+
  scale_y_continuous(limits = c(0,1.5))
```

```
## Warning: Removed 31 rows containing non-finite values (stat_boxplot).
```

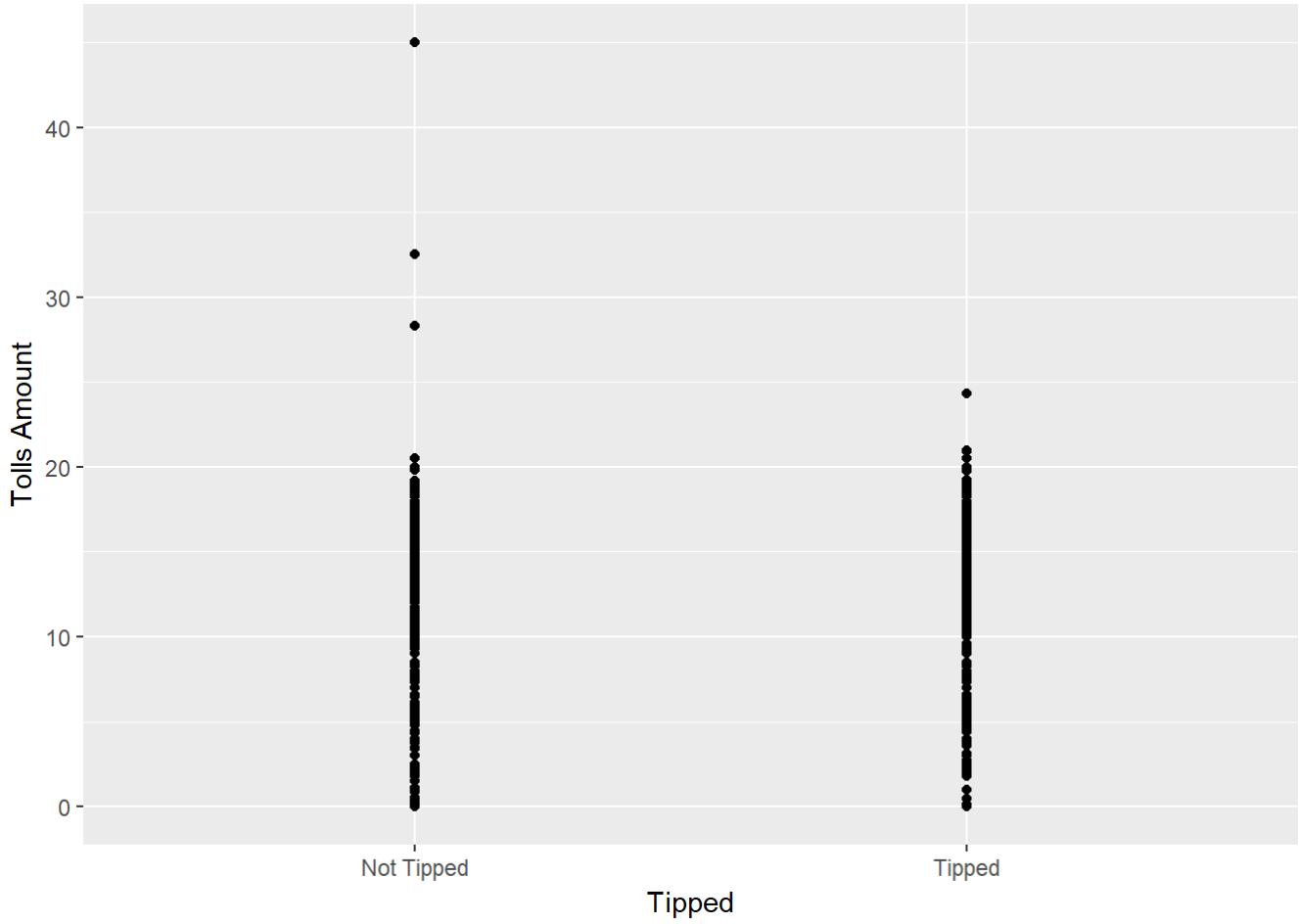



```
Base%>%ggplot(aes(x=surcharge,fill=Flag_Tip_Label))+
  geom_density(alpha=.2)+
  ylab('Density')+
  xlab('Surcharge')+
  scale_x_continuous(limits = c(0,1.5))
```

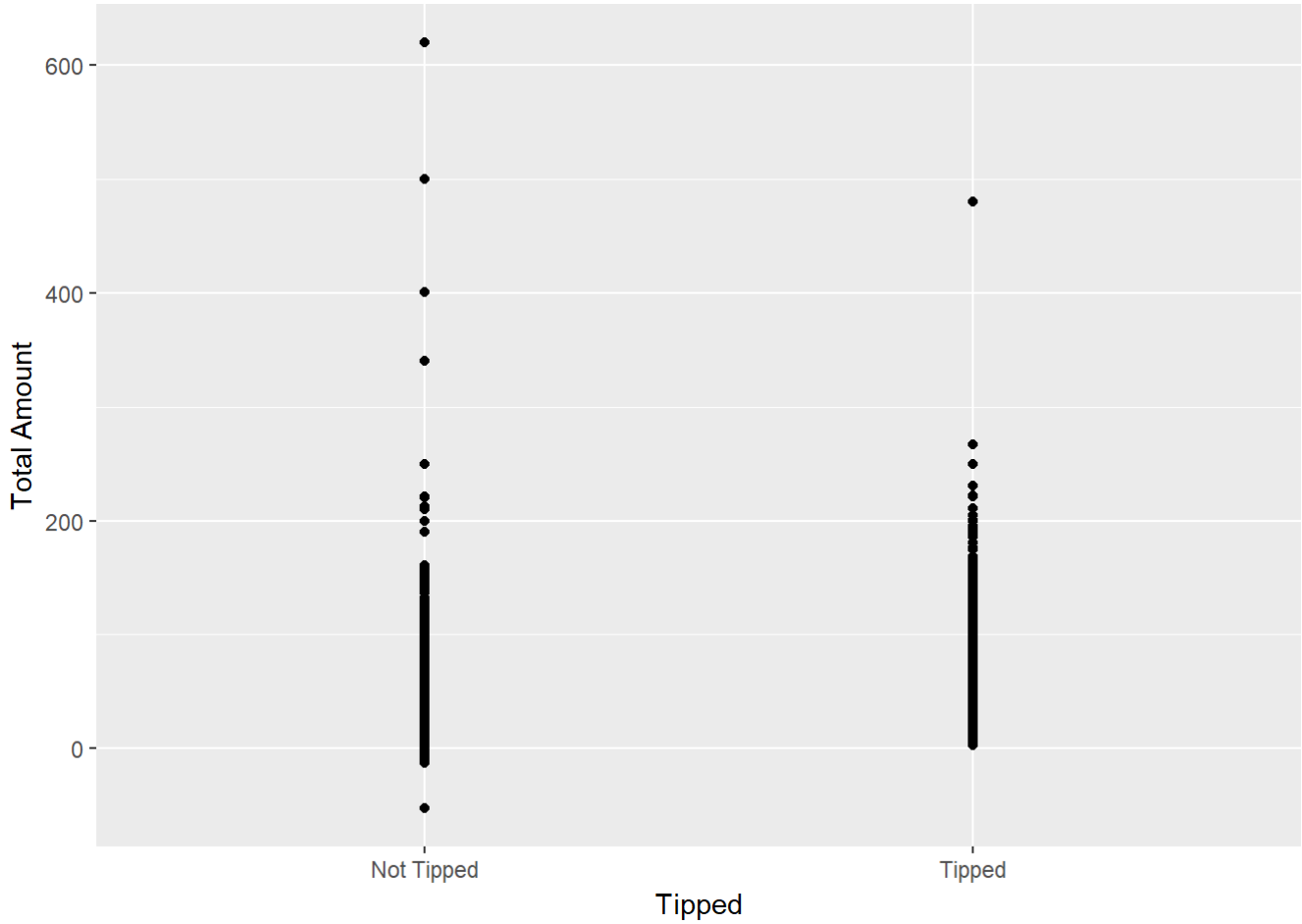
```
## Warning: Removed 31 rows containing non-finite values (stat_density).
```




```
Base%>%ggplot(aes(y=tolls_amount,x=Flag_Tip_Label))+
  geom_point()+
  ylab('Tolls Amount')+
  xlab('Tipped')
```




```
Base%>%ggplot(aes(y=total_amount,x=Flag_Tip_Label))+
  geom_point()+
  ylab('Total Amount')+
  xlab('Tipped')
```



This part of the study ends up to be a kind of variable selection, a.k.a feature selection, not an automated way, but a very interesting way to do it.

It was possible to see how the Tipped variabel, **Flag_Tip_Label** interacts with all the other variabels, except the ids variabels **medallion**, **hack_license** and **vendor_id**.

The original post (https://blogs.msdn.microsoft.com/microsoftrservertigerteam/2017/01/17/predicting-nyc-taxi-tips-using-microsoftml/) states a formula, explaining the Tipped variabel through **passenger_count**, **trip_time_in_secs**, **trip_distance** and **total_amount** variabels.

I thick that the **payment_tipe** variabel also should to be in the model, maybe I wuold compare this new sugestion with the first one.

Fitting models

From here, the code is the same one founded in the original post (https://blogs.msdn.microsoft.com/microsoftrservertigerteam/2017/01/17/predicting-nyc-taxi-tips-using-microsoftml/)

```
set.seed(2345, "L'Ecuyer-CMRG")
  # Randomly split the data 75-25 between train and test sets.
  dataProb <- c(Train = 0.75, Test = 0.25)
  dataSplit <-
    rxSplit(Base,
            splitByFactor = "splitVar",
            transforms = list(splitVar =
                                sample(dataFactor,
                                       size = .rxNumRows,
                                       replace = TRUE,
                                       prob = dataProb)),
            transformObjects =
              list(dataProb = dataProb,
                   dataFactor = factor(names(dataProb),
                                       levels = names(dataProb))),
            outFilesBase = tempfile())
```

```
## Rows Read: 1694163, Total Rows Processed: 1694163, Total Chunk Time: 3.730 seconds
## Rows Read: 1694163, Total Rows Processed: 1694163Rows Read: 1269673, Total Rows Processed:
1269673, Total Chunk Time: 2.653 seconds
## Rows Read: 424490, Total Rows Processed: 424490, Total Chunk Time: 0.933 seconds
## , Total Chunk Time: 15.984 seconds
```

```
  # Name the train and test datasets.
  dataTrain <- dataSplit[[1]]
  dataTest <- dataSplit[[2]]
  rxSummary(~ Flag_Tip, dataTrain)$sDataFrame
```

```
## Rows Read: 1269673, Total Rows Processed: 1269673, Total Chunk Time: 0.063 seconds
## Computation time: 0.353 seconds.
```

```
##       Name         Mean       StdDev Min Max ValidObs MissingObs
## 1 Flag_Tip 0.5235631537 0.4994446659   0   1  1269673          0
```

```
  rxSummary(~ Flag_Tip, dataTest)$sDataFrame
```

```
## Rows Read: 424490, Total Rows Processed: 424490, Total Chunk Time: 0.008 seconds
## Computation time: 0.010 seconds.
```

```
##       Name         Mean       StdDev Min Max ValidObs MissingObs
## 1 Flag_Tip 0.5225234988 0.4994930227   0   1   424490          0
```

```
model <- formula(paste("Flag_Tip ~ passenger_count + trip_time_in_secs + trip_distance + tota
l_amount"))
rxLogisticRegressionFit <- rxLogisticRegression(model, data = dataTrain)
```

```
## Automatically adding a MinMax normalization transform, use 'norm=Warn' or 'norm=No' to tur
n this behavior off.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.105, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Automatically converting column 'Flag_Tip' into a factor.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.117, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.102, Transform Time: 0
## Beginning read for block: 2
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.099, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Warning: The number of threads specified in trainer arguments is larger than the concurren
cy factor setting of the environment. Using 2 training threads instead.
## LBFGS multi-threading will attempt to load dataset into memory. In case of out-of-memory i
ssues, turn off multi-threading by setting trainThreads to 1.
## Beginning optimization
## num vars: 5
## improvement criterion: Mean Improvement
## L1 regularization selected 4 of 5 weights.
## Not training a calibrator because it is not needed.
## Elapsed time: 00:00:12.3652212
## Elapsed time: 00:00:00.1948485
```

```
rxFastLinearFit <- rxFastLinear(model, data = dataTrain)
```

```
## Automatically adding a MinMax normalization transform, use 'norm=Warn' or 'norm=No' to tur
n this behavior off.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.1, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Automatically converting column 'Flag_Tip' into a factor.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.098, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.099, Transform Time: 0
## Beginning read for block: 2
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.098, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Using 2 threads to train.
## Automatically choosing a check frequency of 2.
## Auto-tuning parameters: maxIterations = 2.
## Auto-tuning parameters: L2 = 3,938022E-06.
## Auto-tuning parameters: L1Threshold (L1/L2) = 0.
## Using model from last iteration.
## Not training a calibrator because it is not needed.
## Elapsed time: 00:00:02.1652462
## Elapsed time: 00:00:00.0206831
```

```
rxFastTreesFit <- rxFastTrees(model, data = dataTrain)
```

```
## Warning: The number of threads specified in trainer arguments is larger than the concurren
cy factor setting of the environment. Using 2 training threads instead.
## Not adding a normalizer.
## Automatically converting column 'Flag_Tip' into a factor.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.1, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Making per-feature arrays
## Changing data from row-wise to column-wise
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.099, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Processed 1269673 instances
## Binning and forming Feature objects
## Reserved memory for tree learner: 121680 bytes
## Starting to train ...
## Not training a calibrator because it is not needed.
## Elapsed time: 00:00:11.2225005
```

```
rxFastForestFit <- rxFastForest(model, data = dataTrain)
```

```
## Warning: The number of threads specified in trainer arguments is larger than the concurren
cy factor setting of the environment. Using 2 training threads instead.
## Not adding a normalizer.
## Automatically converting column 'Flag_Tip' into a factor.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.1, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Making per-feature arrays
## Changing data from row-wise to column-wise
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.099, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Processed 1269673 instances
## Binning and forming Feature objects
## Reserved memory for tree learner: 121680 bytes
## Starting to train ...
## Warning: 2 of the boosting iterations failed to grow a tree. This is commonly because the
minimum documents in leaf hyperparameter was set too high for this dataset.
## Training calibrator.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.1, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Elapsed time: 00:00:14.5551461
```

```
rxNeuralNetFit <- rxNeuralNet(model, data = dataTrain)
```

```
## Automatically adding a MinMax normalization transform, use 'norm=Warn' or 'norm=No' to tur
n this behavior off.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.098, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Automatically converting column 'Flag_Tip' into a factor.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.098, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.099, Transform Time: 0
## Beginning read for block: 2
## Beginning read for block: 1
## Rows Read: 1269673, Read Time: 0.099, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Using: SSE Math
##
## ***** Net definition *****
##   input Data [4];
##   hidden H [100] sigmoid { // Depth 1
##     from Data all;
##   }
##   output Result [1] sigmoid { // Depth 0
##     from H all;
##   }
## ***** End net definition *****
## Input count: 4
## Output count: 1
## Output Function: Sigmoid
## Loss Function: LogLoss
## PreTrainer: NoPreTrainer
## ________________________________________________________________________
## Starting training...
## Learning rate: 0,001000
## Momentum: 0,000000
## InitWtsDiameter: 0,100000
## ________________________________________________________________________
## Initializing 1 Hidden Layers, 601 Weights...
## Estimated Pre-training MeanError = 0,696093
## Iter:1/100, MeanErr=0,693611(-0,36%%), 390,81M WeightUpdates/sec
## Iter:2/100, MeanErr=0,693291(-0,05%%), 387,40M WeightUpdates/sec
## Iter:3/100, MeanErr=0,692855(-0,06%%), 375,15M WeightUpdates/sec
## Iter:4/100, MeanErr=0,691979(-0,13%%), 387,75M WeightUpdates/sec
## Iter:5/100, MeanErr=0,689888(-0,30%%), 385,49M WeightUpdates/sec
## Iter:6/100, MeanErr=0,685817(-0,59%%), 383,79M WeightUpdates/sec
## Iter:7/100, MeanErr=0,680775(-0,74%%), 382,70M WeightUpdates/sec
## Iter:8/100, MeanErr=0,677331(-0,51%%), 385,03M WeightUpdates/sec
## Iter:9/100, MeanErr=0,675852(-0,22%%), 382,97M WeightUpdates/sec
## Iter:10/100, MeanErr=0,675440(-0,06%%), 385,69M WeightUpdates/sec
## Iter:11/100, MeanErr=0,675325(-0,02%%), 390,11M WeightUpdates/sec
## Iter:12/100, MeanErr=0,675226(-0,01%%), 385,74M WeightUpdates/sec
## Iter:13/100, MeanErr=0,675272(0,01%%), 386,22M WeightUpdates/sec
## Iter:14/100, MeanErr=0,675198(-0,01%%), 385,46M WeightUpdates/sec
## Iter:15/100, MeanErr=0,675134(-0,01%%), 389,66M WeightUpdates/sec
```

```
## Iter:16/100, MeanErr=0,675098(-0,01%%), 372,16M WeightUpdates/sec
## Iter:17/100, MeanErr=0,675066(0,00%%), 384,64M WeightUpdates/sec
## Iter:18/100, MeanErr=0,675049(0,00%%), 376,91M WeightUpdates/sec
## Iter:19/100, MeanErr=0,675026(0,00%%), 386,71M WeightUpdates/sec
## Iter:20/100, MeanErr=0,674977(-0,01%%), 384,51M WeightUpdates/sec
## Iter:21/100, MeanErr=0,674924(-0,01%%), 382,25M WeightUpdates/sec
## Iter:22/100, MeanErr=0,674912(0,00%%), 385,25M WeightUpdates/sec
## Iter:23/100, MeanErr=0,674844(-0,01%%), 389,40M WeightUpdates/sec
## Iter:24/100, MeanErr=0,674823(0,00%%), 382,62M WeightUpdates/sec
## Iter:25/100, MeanErr=0,674803(0,00%%), 385,59M WeightUpdates/sec
## Iter:26/100, MeanErr=0,674746(-0,01%%), 390,01M WeightUpdates/sec
## Iter:27/100, MeanErr=0,674730(0,00%%), 385,58M WeightUpdates/sec
## Iter:28/100, MeanErr=0,674655(-0,01%%), 383,01M WeightUpdates/sec
## Iter:29/100, MeanErr=0,674642(0,00%%), 385,75M WeightUpdates/sec
## Iter:30/100, MeanErr=0,674607(-0,01%%), 382,99M WeightUpdates/sec
## Iter:31/100, MeanErr=0,674538(-0,01%%), 387,29M WeightUpdates/sec
## Iter:32/100, MeanErr=0,674536(0,00%%), 378,48M WeightUpdates/sec
## Iter:33/100, MeanErr=0,674481(-0,01%%), 376,62M WeightUpdates/sec
## Iter:34/100, MeanErr=0,674403(-0,01%%), 386,01M WeightUpdates/sec
## Iter:35/100, MeanErr=0,674381(0,00%%), 383,69M WeightUpdates/sec
## Iter:36/100, MeanErr=0,674301(-0,01%%), 386,85M WeightUpdates/sec
## Iter:37/100, MeanErr=0,674228(-0,01%%), 382,40M WeightUpdates/sec
## Iter:38/100, MeanErr=0,674160(-0,01%%), 383,37M WeightUpdates/sec
## Iter:39/100, MeanErr=0,674069(-0,01%%), 388,91M WeightUpdates/sec
## Iter:40/100, MeanErr=0,673979(-0,01%%), 383,82M WeightUpdates/sec
## Iter:41/100, MeanErr=0,673848(-0,02%%), 383,83M WeightUpdates/sec
## Iter:42/100, MeanErr=0,673775(-0,01%%), 383,35M WeightUpdates/sec
## Iter:43/100, MeanErr=0,673637(-0,02%%), 390,07M WeightUpdates/sec
## Iter:44/100, MeanErr=0,673551(-0,01%%), 392,98M WeightUpdates/sec
## Iter:45/100, MeanErr=0,673453(-0,01%%), 386,19M WeightUpdates/sec
## Iter:46/100, MeanErr=0,673382(-0,01%%), 388,55M WeightUpdates/sec
## Iter:47/100, MeanErr=0,673295(-0,01%%), 382,80M WeightUpdates/sec
## Iter:48/100, MeanErr=0,673224(-0,01%%), 375,06M WeightUpdates/sec
## Iter:49/100, MeanErr=0,673198(0,00%%), 390,59M WeightUpdates/sec
## Iter:50/100, MeanErr=0,673144(-0,01%%), 391,19M WeightUpdates/sec
## Iter:51/100, MeanErr=0,673115(0,00%%), 386,64M WeightUpdates/sec
## Iter:52/100, MeanErr=0,673088(0,00%%), 384,55M WeightUpdates/sec
## Iter:53/100, MeanErr=0,673085(0,00%%), 392,09M WeightUpdates/sec
## Iter:54/100, MeanErr=0,673028(-0,01%%), 387,47M WeightUpdates/sec
## Iter:55/100, MeanErr=0,673027(0,00%%), 384,23M WeightUpdates/sec
## Iter:56/100, MeanErr=0,672973(-0,01%%), 382,95M WeightUpdates/sec
## Iter:57/100, MeanErr=0,672930(-0,01%%), 389,52M WeightUpdates/sec
## Iter:58/100, MeanErr=0,672963(0,00%%), 383,34M WeightUpdates/sec
## Iter:59/100, MeanErr=0,672922(-0,01%%), 388,93M WeightUpdates/sec
## Iter:60/100, MeanErr=0,672896(0,00%%), 386,66M WeightUpdates/sec
## Iter:61/100, MeanErr=0,672868(0,00%%), 387,44M WeightUpdates/sec
## Iter:62/100, MeanErr=0,672845(0,00%%), 385,33M WeightUpdates/sec
## Iter:63/100, MeanErr=0,672800(-0,01%%), 372,28M WeightUpdates/sec
## Iter:64/100, MeanErr=0,672818(0,00%%), 386,99M WeightUpdates/sec
## Iter:65/100, MeanErr=0,672729(-0,01%%), 388,92M WeightUpdates/sec
## Iter:66/100, MeanErr=0,672742(0,00%%), 389,05M WeightUpdates/sec
## Iter:67/100, MeanErr=0,672702(-0,01%%), 388,50M WeightUpdates/sec
## Iter:68/100, MeanErr=0,672702(0,00%%), 387,75M WeightUpdates/sec
## Iter:69/100, MeanErr=0,672631(-0,01%%), 382,91M WeightUpdates/sec
## Iter:70/100, MeanErr=0,672621(0,00%%), 383,43M WeightUpdates/sec
## Iter:71/100, MeanErr=0,672587(-0,01%%), 386,86M WeightUpdates/sec
## Iter:72/100, MeanErr=0,672569(0,00%%), 384,59M WeightUpdates/sec
## Iter:73/100, MeanErr=0,672560(0,00%%), 384,89M WeightUpdates/sec
```

```
## Iter:74/100, MeanErr=0,672498(-0,01%%), 384,37M WeightUpdates/sec
## Iter:75/100, MeanErr=0,672502(0,00%%), 386,16M WeightUpdates/sec
## Iter:76/100, MeanErr=0,672468(-0,01%%), 387,39M WeightUpdates/sec
## Iter:77/100, MeanErr=0,672435(0,00%%), 383,83M WeightUpdates/sec
## Iter:78/100, MeanErr=0,672392(-0,01%%), 374,49M WeightUpdates/sec
## Iter:79/100, MeanErr=0,672355(-0,01%%), 379,71M WeightUpdates/sec
## Iter:80/100, MeanErr=0,672348(0,00%%), 391,53M WeightUpdates/sec
## Iter:81/100, MeanErr=0,672339(0,00%%), 382,67M WeightUpdates/sec
## Iter:82/100, MeanErr=0,672331(0,00%%), 388,71M WeightUpdates/sec
## Iter:83/100, MeanErr=0,672240(-0,01%%), 391,03M WeightUpdates/sec
## Iter:84/100, MeanErr=0,672257(0,00%%), 387,62M WeightUpdates/sec
## Iter:85/100, MeanErr=0,672213(-0,01%%), 390,87M WeightUpdates/sec
## Iter:86/100, MeanErr=0,672157(-0,01%%), 384,19M WeightUpdates/sec
## Iter:87/100, MeanErr=0,672127(0,00%%), 383,61M WeightUpdates/sec
## Iter:88/100, MeanErr=0,672117(0,00%%), 384,70M WeightUpdates/sec
## Iter:89/100, MeanErr=0,672103(0,00%%), 383,15M WeightUpdates/sec
## Iter:90/100, MeanErr=0,672072(0,00%%), 390,25M WeightUpdates/sec
## Iter:91/100, MeanErr=0,672025(-0,01%%), 392,23M WeightUpdates/sec
## Iter:92/100, MeanErr=0,671979(-0,01%%), 387,49M WeightUpdates/sec
## Iter:93/100, MeanErr=0,671955(0,00%%), 373,03M WeightUpdates/sec
## Iter:94/100, MeanErr=0,671944(0,00%%), 383,79M WeightUpdates/sec
## Iter:95/100, MeanErr=0,671884(-0,01%%), 383,42M WeightUpdates/sec
## Iter:96/100, MeanErr=0,671858(0,00%%), 381,62M WeightUpdates/sec
## Iter:97/100, MeanErr=0,671820(-0,01%%), 388,47M WeightUpdates/sec
## Iter:98/100, MeanErr=0,671772(-0,01%%), 378,51M WeightUpdates/sec
## Iter:99/100, MeanErr=0,671738(-0,01%%), 384,15M WeightUpdates/sec
## Iter:100/100, MeanErr=0,671691(-0,01%%), 392,35M WeightUpdates/sec
## Done!
## Estimated Post-training MeanError = 0,671329
## ___________________________________________________________________________
## Not training a calibrator because it is not needed.
## Elapsed time: 00:03:19.9055998
```

```
fitScores <- rxPredict(rxLogisticRegressionFit, dataTest, suffix = ".rxLogisticRegression",
                       extraVarsToWrite = names(dataTest),
                       outData = tempfile(fileext = ".xdf"))
```

```
## Beginning read for block: 1
## Rows Read: 424490, Read Time: 0.615, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Elapsed time: 00:00:18.5327430
## Finished writing 424490 rows.
## Writing completed.
```

```
fitScores <- rxPredict(rxFastLinearFit, fitScores, suffix = ".rxFastLinear",
                       extraVarsToWrite = names(fitScores),
                       outData = tempfile(fileext = ".xdf"))
```

```
## Beginning read for block: 1
## Rows Read: 424490, Read Time: 0.62, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Elapsed time: 00:00:18.4044550
## Finished writing 424490 rows.
## Writing completed.
```

```
fitScores <- rxPredict(rxFastTreesFit, fitScores, suffix = ".rxFastTrees",
                       extraVarsToWrite = names(fitScores),
                       outData = tempfile(fileext = ".xdf"))
```

```
## Beginning read for block: 1
## Rows Read: 424490, Read Time: 0.625, Transform Time: 0.001
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Elapsed time: 00:00:20.5500508
## Finished writing 424490 rows.
## Writing completed.
```

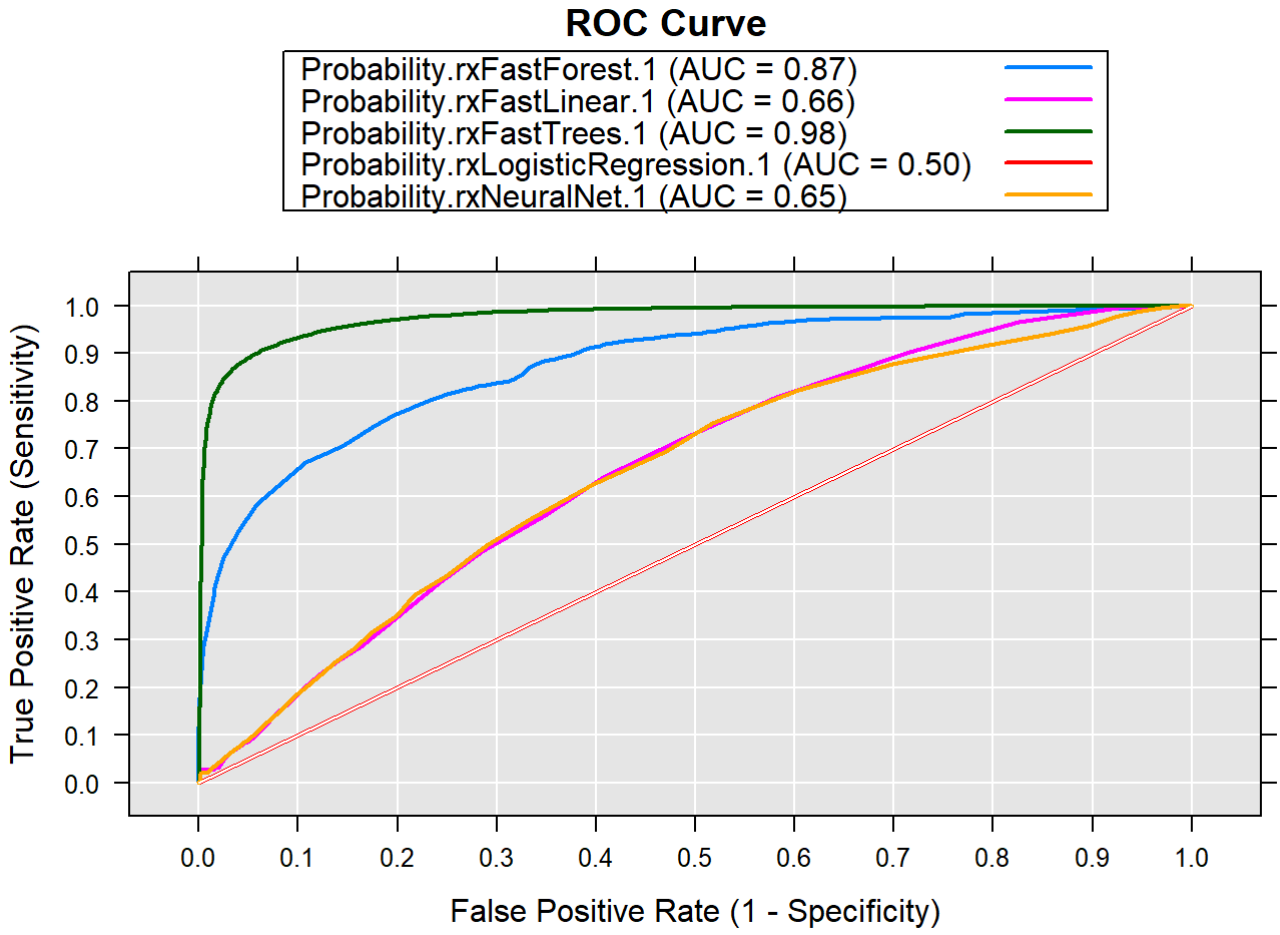
```
fitScores <- rxPredict(rxFastForestFit, fitScores, suffix = ".rxFastForest",
                       extraVarsToWrite = names(fitScores),
                       outData = tempfile(fileext = ".xdf"))
```

```
## Beginning read for block: 1
## Rows Read: 424490, Read Time: 0.633, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Elapsed time: 00:00:21.7202885
## Finished writing 424490 rows.
## Writing completed.
```

```
fitScores <- rxPredict(rxNeuralNetFit, fitScores, suffix = ".rxNeuralNet",
                       extraVarsToWrite = names(fitScores),
                       outData = tempfile(fileext = ".xdf"))
```

```
## Beginning read for block: 1
## Rows Read: 424490, Read Time: 0.637, Transform Time: 0
## Beginning read for block: 2
## No rows remaining. Finished reading data set.
## Elapsed time: 00:00:18.8436828
## Finished writing 424490 rows.
## Writing completed.
```

```
# Compute the fit models's ROC curves.
fitRoc <- rxRoc("Flag_Tip", grep("Probability.", names(fitScores), value = T), fitScores)
# Plot the ROC curves and report their AUCs.
plot(fitRoc)
```




```
# Create a named list of the fit models.
fitList <-
  list(rxLogisticRegression = rxLogisticRegressionFit,
       rxFastLinear = rxFastLinearFit,
       rxFastTrees = rxFastTreesFit,
       rxFastForest = rxFastForestFit,
       rxNeuralNet = rxNeuralNetFit)

# Compute the fit models's AUCs.
fitAuc <- rxAuc(fitRoc)
names(fitAuc) <- substring(names(fitAuc), nchar("Probability.") + 1)

# Find the name of the fit with the largest AUC.
bestFitName <- names(which.max(fitAuc))

# Select the fit model with the largest AUC.
bestFit <- fitList[[bestFitName]]

# Report the fit AUCs.
cat("Fit model AUCs:\n")
```

```
## Fit model AUCs:
```

```
print(fitAuc, digits = 2)
```

```
##         rxFastForest.1         rxFastLinear.1          rxFastTrees.1
##                   0.87                   0.66                   0.98
## rxLogisticRegression.1          rxNeuralNet.1
##                   0.50                   0.65
```

```
# Report the best fit.
cat(paste0("Best fit model with ", bestFitName,
           ", AUC = ", signif(fitAuc[[bestFitName]], digits = 2),
           ".\n"))
```

```
## Best fit model with rxFastTrees.1, AUC = 0.98.
```