Via Taxi Cab Data Challenge

Anthony Lui

Friday, June 10, 2016

Via is considering expanding its service to include rides to and from NYC airports (JFK, LaGuardia). We are trying to decide between these options and how to launch the option we choose, i.e. as part of our core service or as a separate service.

Using the NYC taxi data (described here: http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml (http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml), available either through BiqQuery https://bigquery.cloud.google.com/table/imjasonh-storage:nyctaxi.trip_data (https://bigquery.cloud.google.com/table/imjasonh-storage:nyctaxi.trip_data), or in smaller samples from http://www.andresmh.com/nyctaxitrips/ (http://www.andresmh.com/nyctaxitrips/)), how would you answer the following questions:

```
#Set Working Directory
setwd("C:/Users/Anthony/Documents/Projects/Via")

#Load BigQuery and other libraries
library(bigrquery)
library(ggmap)
library(ggplot2)
library(lubridate)
library(plyr)
library(dplyr)
```

Analysis questions:

1. How would you assess the efficiency of aggregating rides to/from each airport?

The passenger count for cab rides can used to assess the efficiency of aggregating rides to and from each airport. The distribution of passenger count of cab rides can be indicative of the efficacy or potential benefits of a ride share program. Taxi rides with low passenger counts can be combined and result in a more optimized method of transportation.

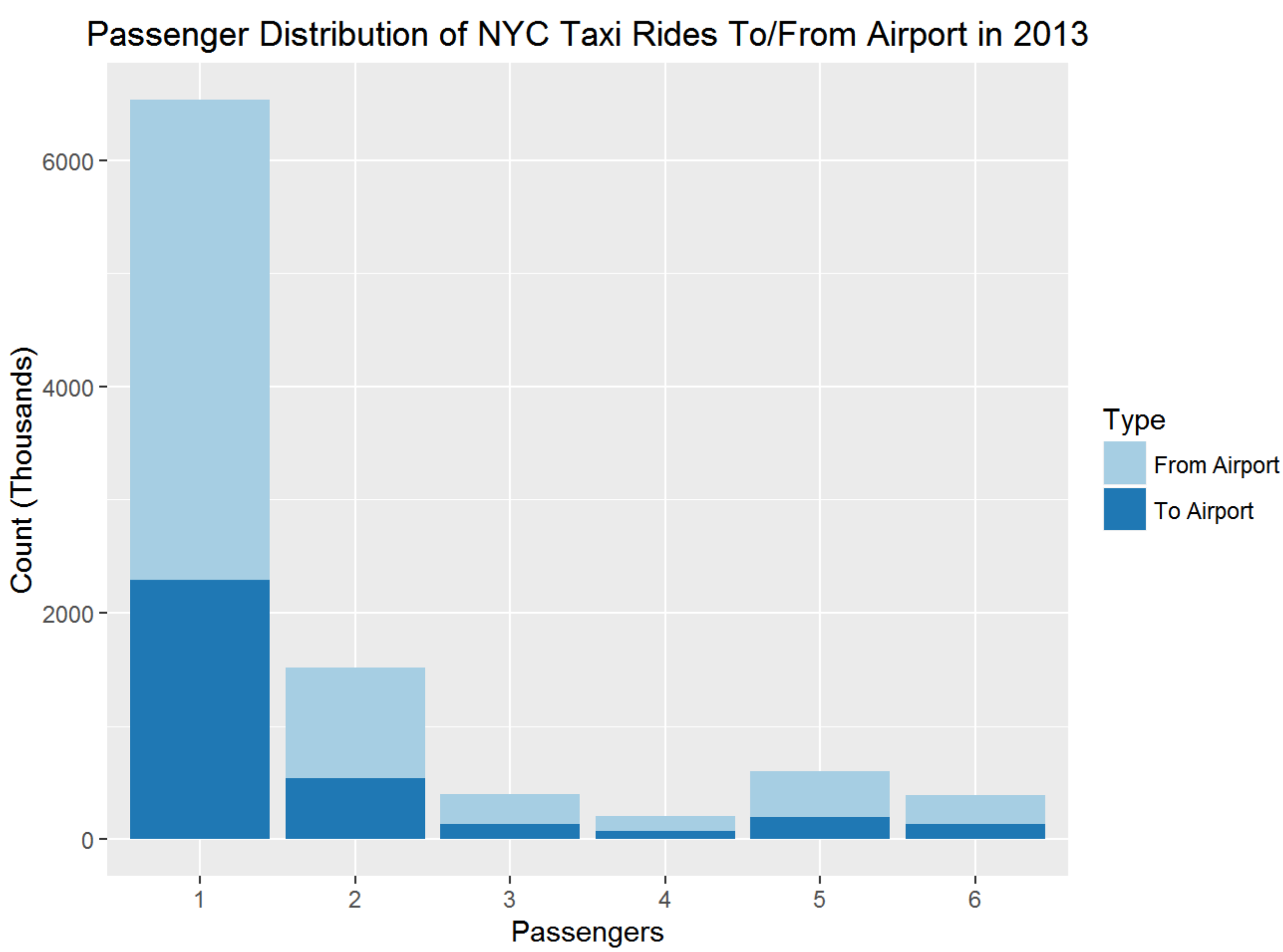
In 2013, there was a total of 9,639,881 taxi trips in NYC headed either to or from an airport (including airport to airport). 83.55% of all taxi rides to the airport contained 1 or 2 passengers indicating that there is a large market for efficiency enhancement through a ride share program.

```
##Airport Dropoff Passenger Counts
project <- "via-challenge"
sql <- "SELECT
                passenger_count,
                count(*) as pass_tot
        FROM [imjasonh-storage:nyctaxi.trip_data]
        WHERE
                (round(cast(dropoff_longitude as float), 4) between -73.7935 and -73.7744 and
                round(cast(dropoff_latitude as float), 4) between 40.6399 and 40.6510) or
                (round(cast(dropoff_latitude as float), 4) between 40.7698 and 40.7736 and
                round(cast(dropoff_longitude as float), 4) between -73.8870 and -73.8844) or
                (round(cast(dropoff_latitude as float), 4) between 40.7705 and  40.7750 and
                round(cast(dropoff_longitude as float), 4) between -73.8795 and -73.8674) or
                (round(cast(dropoff_latitude as float), 4) between 40.7666 and 40.7714 and
                round(cast(dropoff_longitude as float), 4) between -73.8672 and -73.8608)
        GROUP BY passenger_count
        ORDER BY passenger_count
"
air_drop <- query_exec(sql, project=project)
air_drop$pass_dist <- paste(round((air_drop$pass_tot / sum(air_drop$pass_tot))
                                  *100,digits=2),"%",sep="")
air_drop$pass_dist_num <- round((air_drop$pass_tot / sum(air_drop$pass_tot))*100,digits=2)
air_drop$type <- "To Airport"


##Airport Pickup Passenger Counts
project <- "via-challenge"
sql <- "SELECT
                passenger_count,
                count(*) as pass_tot
        FROM [imjasonh-storage:nyctaxi.trip_data]
        WHERE
                (round(cast(pickup_longitude as float), 4) between -73.7935 and -73.7744 and
                round(cast(pickup_latitude as float), 4) between 40.6399 and 40.6510) or
                (round(cast(pickup_latitude as float), 4) between 40.7698 and 40.7736 and
                round(cast(pickup_longitude as float), 4) between -73.8870 and -73.8844) or
                (round(cast(pickup_latitude as float), 4) between 40.7705 and  40.7750 and
                round(cast(pickup_longitude as float), 4) between -73.8795 and -73.8674) or
                (round(cast(pickup_latitude as float), 4) between 40.7666 and 40.7714 and
                round(cast(pickup_longitude as float), 4) between -73.8672 and -73.8608)
        GROUP BY passenger_count
        ORDER BY passenger_count
"
air_pick <- query_exec(sql, project=project)
air_pick$pass_dist <- paste(round((air_pick$pass_tot / sum(air_pick$pass_tot))
                                  *100,digits=2),"%",sep="")
air_pick$pass_dist_num <- round((air_pick$pass_tot / sum(air_pick$pass_tot))*100,digits=2)
air_pick$type <- "From Airport"

air_pass <- subset(rbind(air_drop, air_pick), passenger_count > 0 & passenger_count < 7)
```

```
air_pass_comb <- summarize(group_by(air_pass, passenger_count), total = sum(pass_tot))
air_pass_comb$pass_dist <- paste(round((air_pass_comb$total / sum(air_pass_comb$total))
                                  *100,digits=2),"%",sep="")
air_pass_comb$pass_dist_num <- round((air_pass_comb$total / sum(air_pass_comb$total))*100,digits=2)
air_pass_comb$area <- "Airport"
air_pass_comb
```

```
## Source: local data frame [6 x 5]
##
##   passenger_count   total pass_dist pass_dist_num    area
##             (chr)   (int)     (chr)         (dbl)   (chr)
## 1               1 6538186    67.82%         67.82 Airport
## 2               2 1515622    15.72%         15.72 Airport
## 3               3  394909      4.1%          4.10 Airport
## 4               4  203996     2.12%          2.12 Airport
## 5               5  597986      6.2%          6.20 Airport
## 6               6  389182     4.04%          4.04 Airport
```

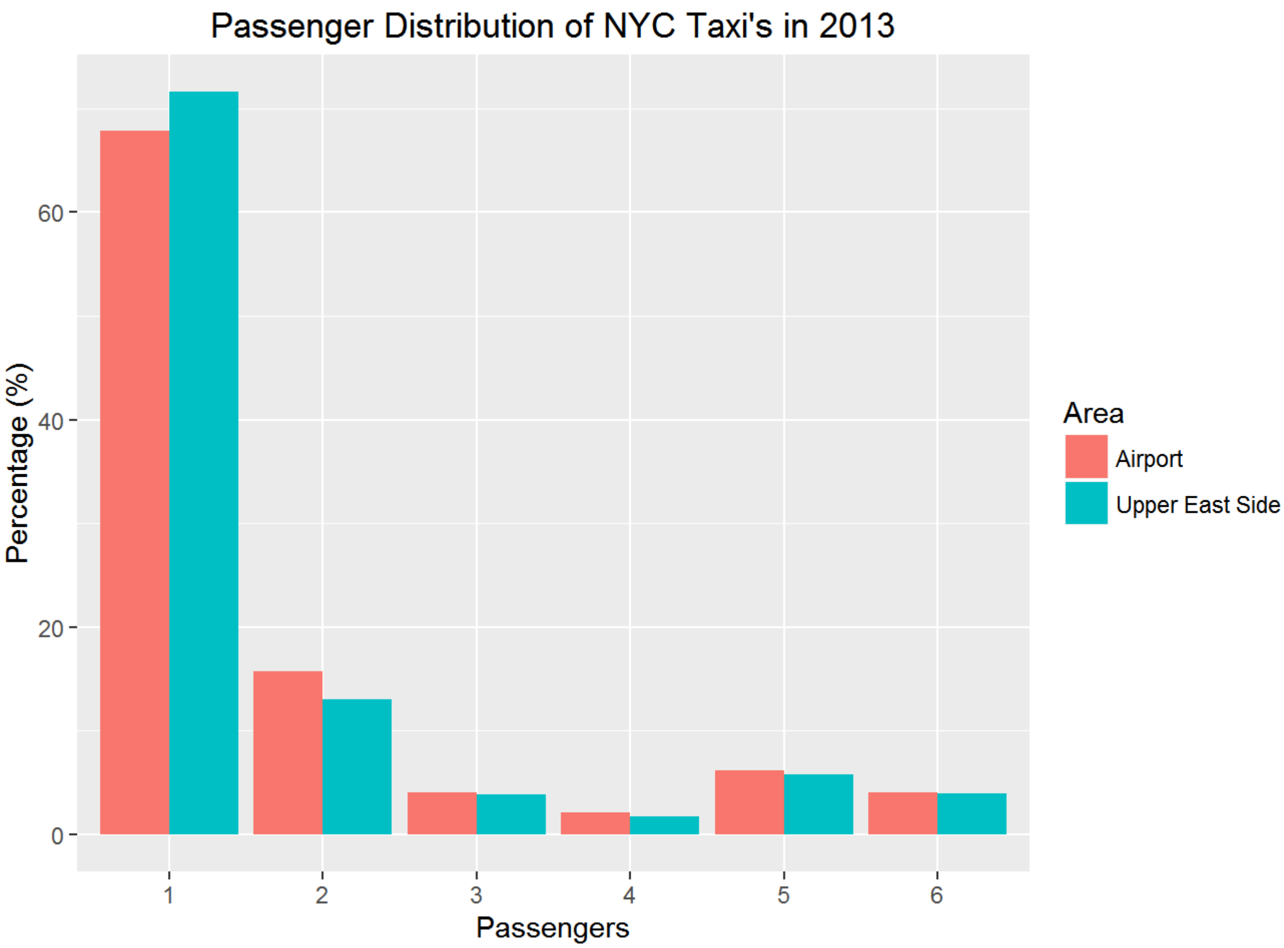
```
ggplot(air_pass, aes(passenger_count, pass_tot/1000, fill=factor(type))) +
        geom_bar(position="stack", stat="identity") +
        labs(x="Passengers", y="Count (Thousands)") +
        ggtitle("Passenger Distribution of NYC Taxi Rides To/From Airport in 2013") +
        scale_fill_brewer(name="Type", palette="Paired")
```



2. How does this compare to our current area of service (e.g. the Upper East Side)?

Analyzing drop offs within a current area of service shows a similar distribution with most of the taxi rides containing 1 or 2 passengers. The success/failure of the current area of service can be indicative of the success/failure of the new airport service given that the two share similar passenger count distributions.

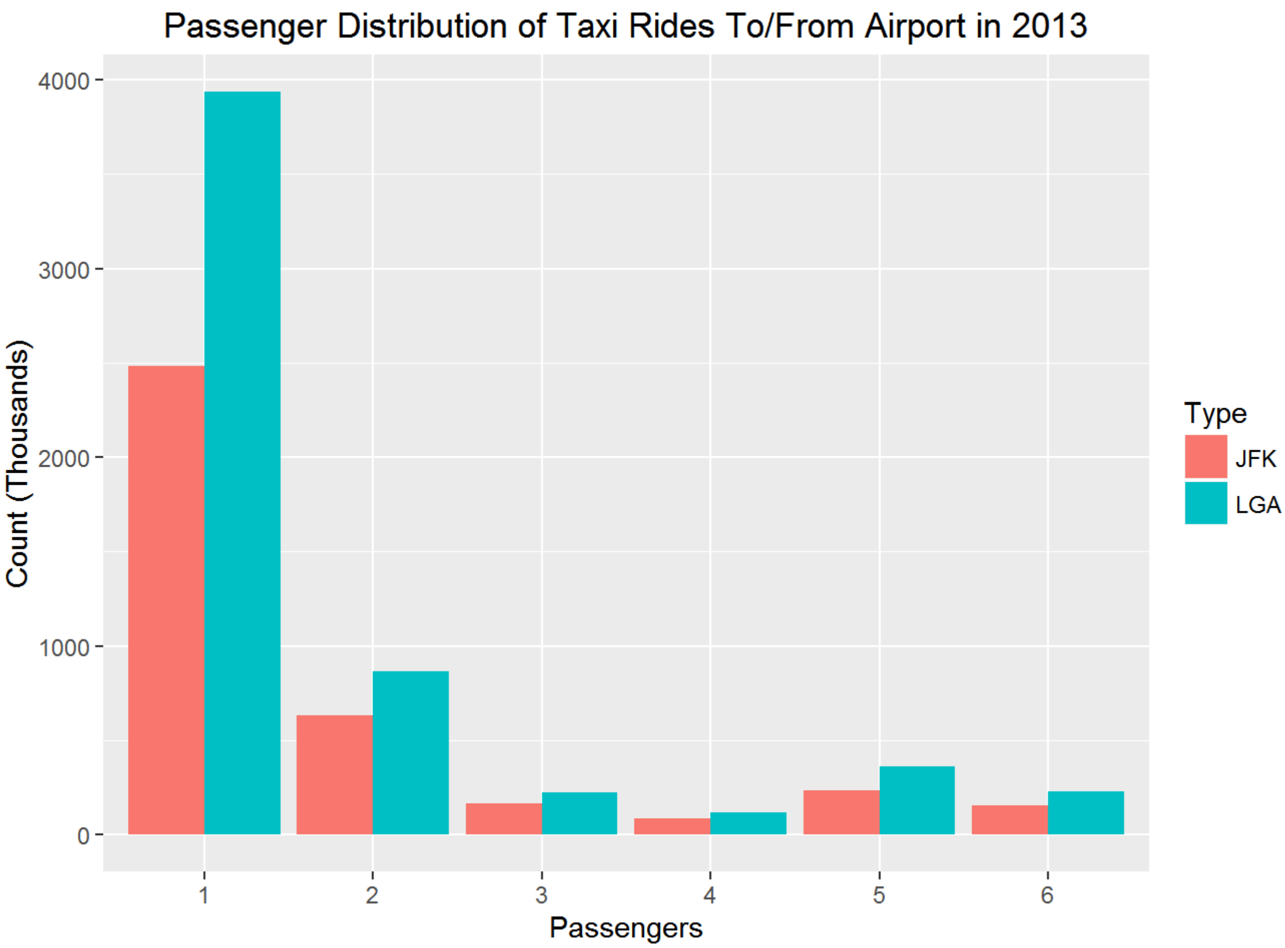
```
project <- "via-challenge"
sql <- "SELECT
                passenger_count,
                count(*) as pass_tot
        FROM [imjasonh-storage:nyctaxi.trip_data]
        WHERE
        (round(cast(dropoff_longitude as float), 4) between -73.9677 and -73.9496 and
        round(cast(dropoff_latitude as float), 4) between 40.7668 and 40.7995) or
        (round(cast(pickup_longitude as float), 4) between -73.9677 and -73.9496 and
        round(cast(pickup_latitude as float), 4) between 40.7668 and 40.7995)
        GROUP BY passenger_count
        ORDER BY passenger_count
"
UES_pass <- query_exec(sql, project=project)
UES_pass$pass_dist <- paste(round((UES_pass$pass_tot / sum(UES_pass$pass_tot))
                                  *100,digits=2),"%",sep="")
UES_pass$pass_dist_num <- round((UES_pass$pass_tot / sum(UES_pass$pass_tot))*100,digits=2)
UES_pass <- subset(UES_pass, passenger_count > 0 & passenger_count < 7)

UES_pass$area <- "Upper East Side"
UES_pass <- rename(UES_pass, total = pass_tot)
comb_pass <- rbind(air_pass_comb, UES_pass)

ggplot(comb_pass, aes(passenger_count, pass_dist_num, fill=factor(area))) +
        geom_bar(position="dodge", stat="identity") +
        labs(x="Passengers", y="Percentage (%)") +
        ggtitle("Passenger Distribution of NYC Taxi's in 2013") +
        scale_fill_discrete(name="Area")
```



3. Which of the airport expansion options is most beneficial and why?

Overall comparison of JFK and LGA indicate that LGA has a higher volume of taxi rides as well as a higher distribution of passenger counts of 1 and 2. Furthermore, LGA is closer to Via's current area of service which might present itself as a lower risk for market entry.

```
project <- "via-challenge"
sql <- "SELECT
                passenger_count,
                count(*) as pass_tot
        FROM [imjasonh-storage:nyctaxi.trip_data]
        WHERE
                (round(cast(pickup_longitude as float), 4) between -73.7935 and -73.7744 and
                round(cast(pickup_latitude as float), 4) between 40.6399 and 40.6510) or
                (round(cast(dropoff_longitude as float), 4) between -73.7935 and -73.7744 and
                round(cast(dropoff_latitude as float), 4) between 40.6399 and 40.6510)
        GROUP BY passenger_count
        ORDER BY passenger_count
"

jfk <- query_exec(sql, project=project)
jfk$airport <- "JFK"

project <- "via-challenge"
sql <- "SELECT
                passenger_count,
                count(*) as pass_tot
        FROM [imjasonh-storage:nyctaxi.trip_data]
        WHERE
                (round(cast(pickup_latitude as float), 4) between 40.7698 and 40.7736 and
                round(cast(pickup_longitude as float), 4) between -73.8870 and -73.8844) or
                (round(cast(pickup_latitude as float), 4) between 40.7705 and  40.7750 and
                round(cast(pickup_longitude as float), 4) between -73.8795 and -73.8674) or
                (round(cast(pickup_latitude as float), 4) between 40.7666 and 40.7714 and
                round(cast(pickup_longitude as float), 4) between -73.8672 and -73.8608) or
                (round(cast(dropoff_latitude as float), 4) between 40.7698 and 40.7736 and
                round(cast(dropoff_longitude as float), 4) between -73.8870 and -73.8844) or
                (round(cast(dropoff_latitude as float), 4) between 40.7705 and  40.7750 and
                round(cast(dropoff_longitude as float), 4) between -73.8795 and -73.8674) or
                (round(cast(dropoff_latitude as float), 4) between 40.7666 and 40.7714 and
                round(cast(dropoff_longitude as float), 4) between -73.8672 and -73.8608)
        GROUP BY passenger_count
        ORDER BY passenger_count
"
lga <- query_exec(sql, project=project)
lga$airport <- "LGA"

air_location <- rbind(jfk,lga)
air_location <- subset(air_location, passenger_count > 0 & passenger_count <7)

ggplot(air_location, aes(passenger_count, pass_tot/1000, fill=factor(airport))) +
        geom_bar(position="dodge", stat="identity") +
        labs(x="Passengers", y="Count (Thousands)") +
        ggtitle("Passenger Distribution of Taxi Rides To/From Airport in 2013") +
        scale_fill_discrete(name="Type")
```



4. Would you launch airports as a separate service or as a new service? Why?

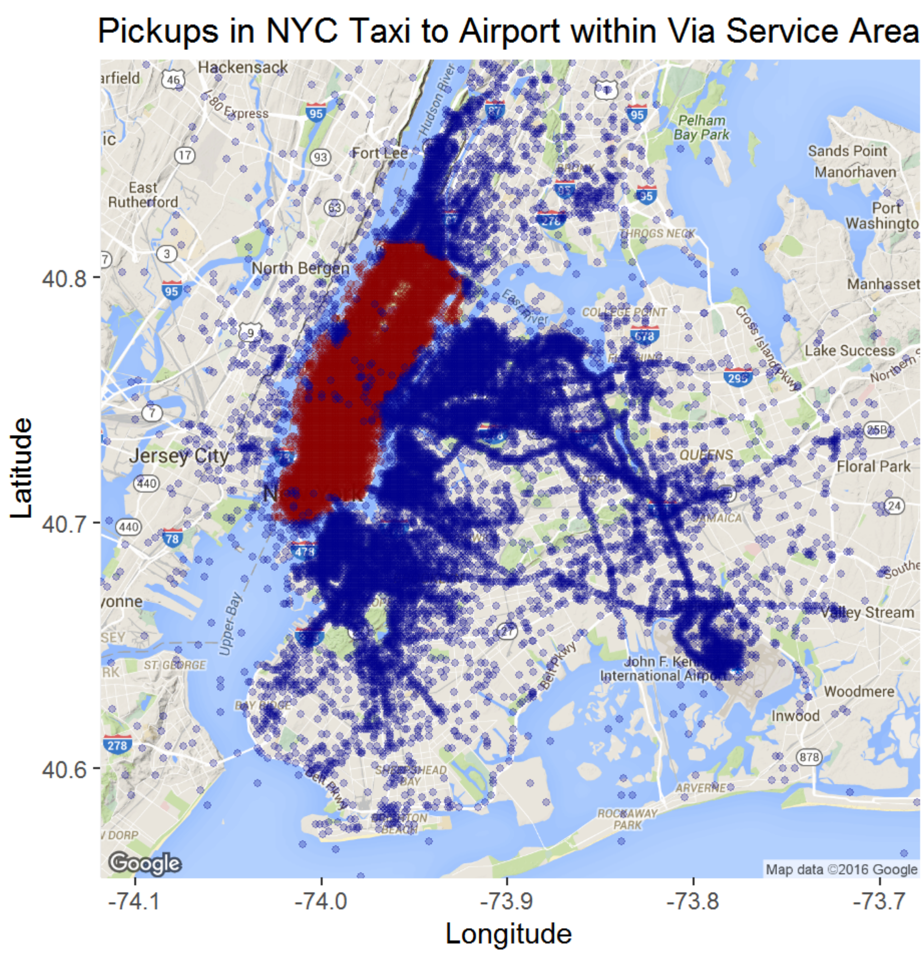
Comparing the current service with airport service can help determine if it makes more sense to launch airports as a separate service or as a new service. 86% of the passengers that are dropped off at airports were picked up within Via's current area of service. With this in mind, it would be feasible to integrate as a new feature in the current service. However, comparing where airport pickups are dropped off may not be as simple to integrate because taxi passengers will be dropped off in locations outside of Via's current service.

```
##Analyze the location where passengers are picked up for LGA and JFK.
##Pull zip codes and compare if its within range of Via's service.



project <- "via-challenge"
sql <- "SELECT
                round(cast(pickup_latitude as float), 3) as lat,
                round(cast(pickup_longitude as float), 3) as long,
                sum(cast(passenger_count as integer)) as pass_tot
        FROM [imjasonh-storage:nyctaxi.trip_data]
        WHERE
                (round(cast(dropoff_longitude as float), 4) between -73.7935 and -73.7744 and
                round(cast(dropoff_latitude as float), 4) between 40.6399 and 40.6510) or
                (round(cast(dropoff_latitude as float), 4) between 40.7698 and 40.7736 and
                round(cast(dropoff_longitude as float), 4) between -73.8870 and -73.8844) or
                (round(cast(dropoff_latitude as float), 4) between 40.7705 and  40.7750 and
                round(cast(dropoff_longitude as float), 4) between -73.8795 and -73.8674) or
                (round(cast(dropoff_latitude as float), 4) between 40.7666 and 40.7714 and
                round(cast(dropoff_longitude as float), 4) between -73.8672 and -73.8608)
        GROUP BY lat, long
"
pass_loc <- query_exec(sql, project=project)
```

```
##
Retrieving data:  2.4s
```

```
#clean up data
pass_loc1 <- subset(pass_loc , lat > 40 & lat < 41 & long >-75 & long < -73)


##Cut down to Via estimate service area (easier for Reverse Geocode to execute)
pass_loc2 <- subset(pass_loc1 , lat > 40.70073 & lat < 40.81487 & long >-74.025805 & long < -73.923798)


##Iterate Reverse Geocode for data frame (need to change IP addresses due to query limits)
#for(i in 1:nrow(pass_loc2)) {
#        pass_loc2[i, 5] <- revgeocode(c(pass_loc2[i,3], pass_loc2[i,2]), override_limit=T)
#       }

##Read in completed file with Reverse Geocode run on different IP addresses (due to query limits)

pass_loc_complete <- read.csv("pass_loc_complete.csv")
pass_loc_complete$V4 <- as.character(pass_loc_complete$V4)
names(pass_loc_complete)[names(pass_loc_complete)=="V4"] <- "addy"

##Via service zip codes
via_zip <-c(
        "10004", "10280", "10006","10005", "10038",
        "10007", "10281", "10282", "10013", "10002",
        "10012", "10014", "10009", "10003", "10011",
        "10278", "10271", "10016", "10010", "10001",
        "10018", "10017", "10036", "10022", "10019",
        "10020", "10065", "10023", "10021", "10075",
        "10024", "10028", "10128", "10025", "10029",
        "10026", "10027", "10035")

##match to via zipcodes
pass_loc_complete$zip <- regmatches(pass_loc_complete$addy, regexec("[0-9]{5}", pass_loc_complete$addy))

pass_loc_complete$via <- pass_loc_complete$zip %in% via_zip

#combine with full data set
pass_loc_complete$key <- paste(pass_loc_complete$long, pass_loc_complete$lat)
pass_loc1$key <- paste(pass_loc1$long, pass_loc1$lat)
pass_loc_not <- subset(pass_loc1, (!(pass_loc1$key %in% pass_loc_complete$key)))
pass_loc_not$via <- FALSE
pass_loc_not$addy <- "N/A"
pass_loc_not$zip <- "N/A"
pass_loc_not$X <- "N/A"

pass_loc_full <- rbind(pass_loc_complete, pass_loc_not)


##Generate visual with full data
myLocation <- c(lon=-73.8993, lat=40.7223)
myMap <- get_map(location=myLocation, source="google", maptype="terrain", crop= FALSE, zoom=11)
```

```
ggmap(myMap, legend="topright") +
        geom_point(aes(x=long, y= lat),
                   subset(pass_loc_full, via==TRUE),
                   alpha=.25, color="darkred", size = 1) +
        geom_point(aes(x=long, y= lat),
                   subset(pass_loc_full, via==FALSE),
                   alpha=.25, color="darkblue", size = 1) +
        labs(x = 'Longitude', y = 'Latitude') + ggtitle('Pickups in NYC Taxi to Airport within Via Service A
rea')
```




```
##Percentage of JFK/LGA passengers from current Via service area
pass_loc_full_via <- ddply(pass_loc_full, .(via), summarise, tot=sum(pass_tot))

pass_loc_full_via$pass_dist <- paste(round((pass_loc_full_via$tot / sum(pass_loc_full_via$tot))
                                           *100,digits=2),"%",sep="")
pass_loc_full_via$pass_dist_num <- round((pass_loc_full_via$tot / sum(pass_loc_full_via$tot))*100,digits=2)

pass_loc_full_via
```

```
##     via     tot pass_dist pass_dist_num
## 1 FALSE  802308    13.82%         13.82
## 2  TRUE 5003938    86.18%         86.18
```

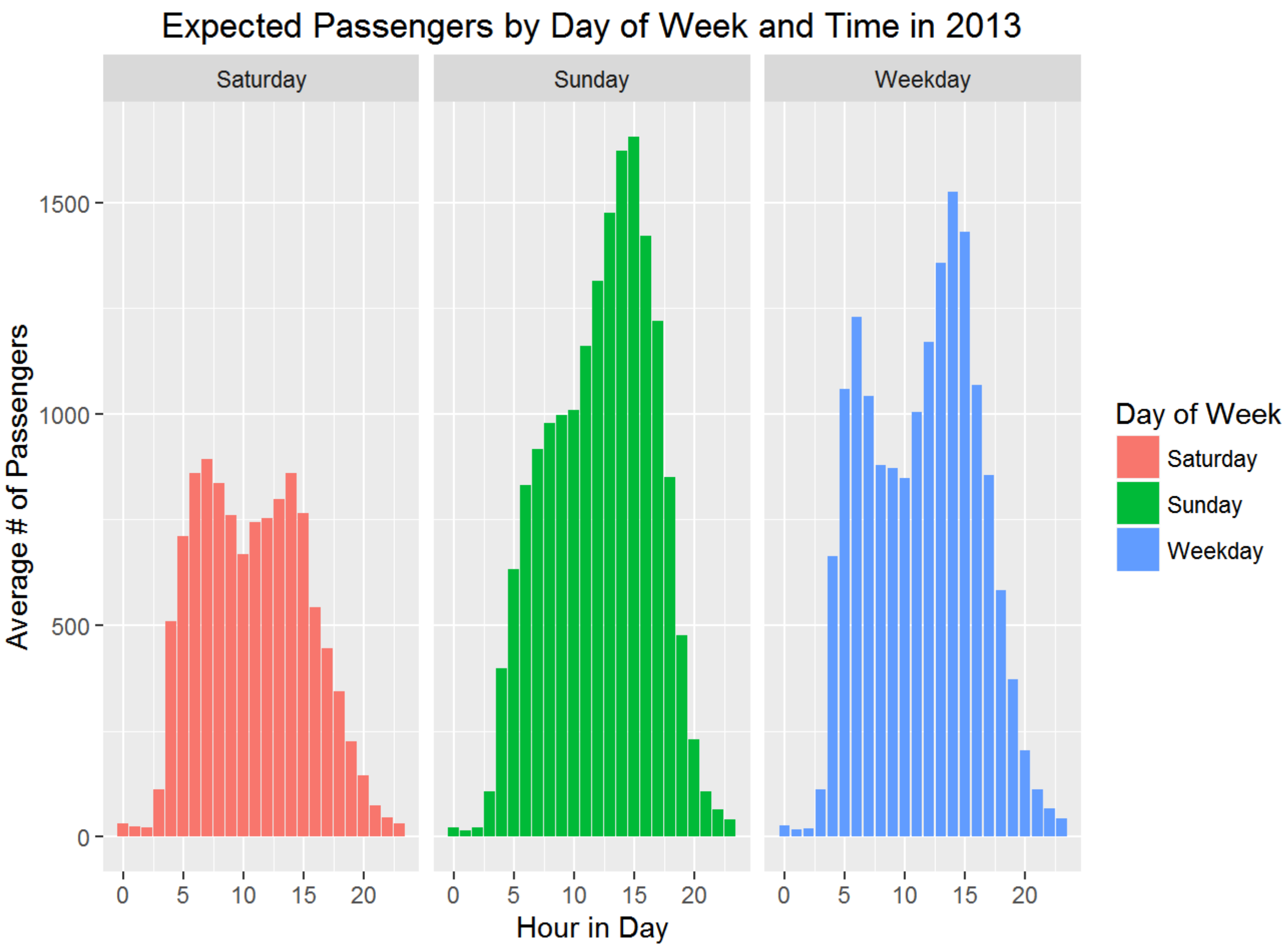
5. Would you launch airport rides during all our hours of service (6am-12am on weekdays and 10am-12am on Saturdays) or only for certain hours? Which hours?

Taxi rides data indicates that pickup times for airports differ between Saturday, Sunday, and Weekdays. On Saturdays, service is comparatively slower with peaks at 7 AM and 2 PM. On Sundays, service gradually builds up peaks at 3 PM. On Weekdays service peaks at 6 AM and 2 PM.

These differ from Via's current hours of service. From a business standpoint, it would make sense to design the product to be available when there are high volumes of demand for taxi rides to the airport. For example, Saturday hours could be lowered while Sunday service is implemented around the peaks where high demand is expected.

```
project <- "via-challenge"
sql <- "SELECT
                date(pickup_datetime) as date,
                hour(pickup_datetime) as time_day,
                sum(cast(passenger_count as integer)) as pass_tot
        FROM [imjasonh-storage:nyctaxi.trip_data]
        WHERE
                (round(cast(dropoff_longitude as float), 4) between -73.7935 and -73.7744 and
                round(cast(dropoff_latitude as float), 4) between 40.6399 and 40.6510) or
                (round(cast(dropoff_latitude as float), 4) between 40.7698 and 40.7736 and
                round(cast(dropoff_longitude as float), 4) between -73.8870 and -73.8844) or
                (round(cast(dropoff_latitude as float), 4) between 40.7705 and  40.7750 and
                round(cast(dropoff_longitude as float), 4) between -73.8795 and -73.8674) or
                (round(cast(dropoff_latitude as float), 4) between 40.7666 and 40.7714 and
                round(cast(dropoff_longitude as float), 4) between -73.8672 and -73.8608)
        GROUP BY date, time_day
        ORDER BY date, time_day
"
pass_time <- query_exec(sql, project=project)
```

```
df<-pass_time
df$date <- as.character(df$date)
df$wday <-wday(df$date)

df$weekend <- ifelse((df$wday==1),"Sunday",
                     ifelse((df$wday==7), "Saturday", "Weekday"))

df_wknd <- ddply(df, .(weekend, time_day), summarize, avg = mean(pass_tot))
df_wday <- ddply(df, .(wday, time_day), summarize, avg = mean(pass_tot))

##weekday, sat, sun time
ggplot(df_wknd, aes(time_day, avg, fill=factor(weekend))) +
        geom_bar(position="dodge", stat="identity") +
        facet_grid(.~weekend) +
        labs(x="Hour in Day", y="Average # of Passengers") +
        ggtitle("Expected Passengers by Day of Week and Time in 2013") +
        scale_fill_discrete(name="Day of Week")
```




```
df_sat <- subset(df, wday==7)
df_sun <- subset(df, wday==1)
df_week <- subset(df, wday>1 & wday<7)
```

6. How would you price airport rides and why (our current model is a $5.00 flat fee weekdays before 9pm and $5.95 weeknights after 9pm and all day Saturdays)?

Based on the results from Question 5, airport service should take advantage of peak times by charging more during periods of high demand and should reduce or cut service during low demand.

A few basic linear regression models were create to estimate expected values for ridership on Saturdays, Sundays, and Weekdays. Combining these estimates of riders with a pricing model was used to forecast revenue.

Using a $5.00 flat fee on any given weekday would generate an expected revenue of $82,792. Implementing $6.00 fee based on time periods of high demand would generate an expected revenue of $92,673, which is a 12% increase.

```
##fitting a few different linear models

df_sat <- subset(df, wday==7)
df_sun <- subset(df, wday==1)
df_week <- subset(df, wday>1 & wday<7)

##fitting a few differe linear models
fit0 <- lm(df$pass_tot ~ factor(df$weekend) * factor(df$time_day))
fit1 <- lm(df$pass_tot ~ factor(df$time_day))
fit2 <- lm(df$pass_tot ~ factor(df$weekend))
fit3 <- lm(df$pass_tot ~ factor(df$weekend) + factor(df$time_day))

fit4 <- lm(df_sat$pass_tot ~ factor(df_sat$time_day))
fit5 <- lm(df_sun$pass_tot ~ factor(df_sun$time_day))
fit6 <- lm(df_week$pass_tot ~ factor(df_week$time_day))

summary(fit4)
```

```
##
## Call:
## lm(formula = df_sat$pass_tot ~ factor(df_sat$time_day))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -842.35  -37.48   -1.31   39.46  944.08
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 31.942     18.054   1.769  0.07710 .
## factor(df_sat$time_day)1    -8.158     25.657  -0.318  0.75057
## factor(df_sat$time_day)2    -9.077     25.532  -0.356  0.72227
## factor(df_sat$time_day)3    80.558     25.532   3.155  0.00164 **
## factor(df_sat$time_day)4   478.750     25.532  18.751  < 2e-16 ***
## factor(df_sat$time_day)5   677.981     25.532  26.554  < 2e-16 ***
## factor(df_sat$time_day)6   827.635     25.532  32.415  < 2e-16 ***
## factor(df_sat$time_day)7   861.404     25.532  33.738  < 2e-16 ***
## factor(df_sat$time_day)8   805.288     25.532  31.540  < 2e-16 ***
## factor(df_sat$time_day)9   727.673     25.532  28.500  < 2e-16 ***
## factor(df_sat$time_day)10  635.500     25.532  24.890  < 2e-16 ***
## factor(df_sat$time_day)11  711.577     25.532  27.870  < 2e-16 ***
## factor(df_sat$time_day)12  721.000     25.532  28.239  < 2e-16 ***
## factor(df_sat$time_day)13  765.673     25.532  29.989  < 2e-16 ***
## factor(df_sat$time_day)14  828.865     25.532  32.464  < 2e-16 ***
## factor(df_sat$time_day)15  733.635     25.532  28.734  < 2e-16 ***
## factor(df_sat$time_day)16  510.288     25.532  19.986  < 2e-16 ***
## factor(df_sat$time_day)17  414.519     25.532  16.235  < 2e-16 ***
## factor(df_sat$time_day)18  311.096     25.532  12.185  < 2e-16 ***
## factor(df_sat$time_day)19  193.558     25.532   7.581 6.75e-14 ***
## factor(df_sat$time_day)20  113.462     25.532   4.444 9.64e-06 ***
## factor(df_sat$time_day)21   41.096     25.532   1.610  0.10775
## factor(df_sat$time_day)22   13.731     25.532   0.538  0.59082
## factor(df_sat$time_day)23   -1.635     25.532  -0.064  0.94896
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 130.2 on 1223 degrees of freedom
## Multiple R-squared:  0.8663, Adjusted R-squared:  0.8638
## F-statistic: 344.5 on 23 and 1223 DF,  p-value: < 2.2e-16
```

```
summary(fit5)
```

```
##
## Call:
## lm(formula = df_sun$pass_tot ~ factor(df_sun$time_day))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1105.92   -48.11     3.13    84.06   617.38
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 22.5000    26.3333   0.854   0.3930
## factor(df_sun$time_day)1    -6.7500    37.2409  -0.181   0.8562
## factor(df_sun$time_day)2    -0.7353    37.4230  -0.020   0.9843
## factor(df_sun$time_day)3    83.8654    37.2409   2.252   0.0245 *
## factor(df_sun$time_day)4   375.7692    37.2409  10.090  < 2e-16 ***
## factor(df_sun$time_day)5   610.6154    37.2409  16.396  < 2e-16 ***
## factor(df_sun$time_day)6   810.1154    37.2409  21.753  < 2e-16 ***
## factor(df_sun$time_day)7   895.3462    37.2409  24.042  < 2e-16 ***
## factor(df_sun$time_day)8   956.0577    37.2409  25.672  < 2e-16 ***
## factor(df_sun$time_day)9   974.4231    37.2409  26.165  < 2e-16 ***
## factor(df_sun$time_day)10  986.3269    37.2409  26.485  < 2e-16 ***
## factor(df_sun$time_day)11 1139.2500    37.2409  30.591  < 2e-16 ***
## factor(df_sun$time_day)12 1291.9423    37.2409  34.691  < 2e-16 ***
## factor(df_sun$time_day)13 1453.8654    37.2409  39.039  < 2e-16 ***
## factor(df_sun$time_day)14 1599.1538    37.2409  42.941  < 2e-16 ***
## factor(df_sun$time_day)15 1633.4231    37.2409  43.861  < 2e-16 ***
## factor(df_sun$time_day)16 1399.5962    37.2409  37.582  < 2e-16 ***
## factor(df_sun$time_day)17 1196.4615    37.2409  32.128  < 2e-16 ***
## factor(df_sun$time_day)18  829.0769    37.2409  22.263  < 2e-16 ***
## factor(df_sun$time_day)19  453.3077    37.2409  12.172  < 2e-16 ***
## factor(df_sun$time_day)20  208.3654    37.2409   5.595 2.72e-08 ***
## factor(df_sun$time_day)21   83.4231    37.2409   2.240   0.0253 *
## factor(df_sun$time_day)22   42.8077    37.2409   1.149   0.2506
## factor(df_sun$time_day)23   19.3462    37.2409   0.519   0.6035
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 189.9 on 1223 degrees of freedom
## Multiple R-squared:  0.8966, Adjusted R-squared:  0.8947
## F-statistic: 461.2 on 23 and 1223 DF,  p-value: < 2.2e-16
```

```
summary(fit6)
```

```
##
## Call:
## lm(formula = df_week$pass_tot ~ factor(df_week$time_day))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1409.75   -61.40    -1.63    64.24   817.81
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   26.341     11.031   2.388  0.01697 *
## factor(df_week$time_day)1     -8.939     15.600  -0.573  0.56666
## factor(df_week$time_day)2     -7.713     15.600  -0.494  0.62103
## factor(df_week$time_day)3     84.801     15.600   5.436 5.65e-08 ***
## factor(df_week$time_day)4    637.261     15.600  40.851  < 2e-16 ***
## factor(df_week$time_day)5   1032.510     15.600  66.188  < 2e-16 ***
## factor(df_week$time_day)6   1201.847     15.600  77.044  < 2e-16 ***
## factor(df_week$time_day)7   1015.598     15.600  65.104  < 2e-16 ***
## factor(df_week$time_day)8    852.966     15.600  54.679  < 2e-16 ***
## factor(df_week$time_day)9    846.590     15.600  54.270  < 2e-16 ***
## factor(df_week$time_day)10   821.326     15.600  52.651  < 2e-16 ***
## factor(df_week$time_day)11   977.019     15.600  62.631  < 2e-16 ***
## factor(df_week$time_day)12  1143.843     15.600  73.325  < 2e-16 ***
## factor(df_week$time_day)13  1331.349     15.600  85.345  < 2e-16 ***
## factor(df_week$time_day)14  1498.406     15.600  96.054  < 2e-16 ***
## factor(df_week$time_day)15  1404.429     15.600  90.030  < 2e-16 ***
## factor(df_week$time_day)16  1041.713     15.600  66.778  < 2e-16 ***
## factor(df_week$time_day)17   829.939     15.600  53.203  < 2e-16 ***
## factor(df_week$time_day)18   556.670     15.600  35.685  < 2e-16 ***
## factor(df_week$time_day)19   345.713     15.600  22.162  < 2e-16 ***
## factor(df_week$time_day)20   178.061     15.600  11.415  < 2e-16 ***
## factor(df_week$time_day)21    84.513     15.600   5.418 6.26e-08 ***
## factor(df_week$time_day)22    41.195     15.600   2.641  0.00829 **
## factor(df_week$time_day)23    17.218     15.600   1.104  0.26973
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 178.2 on 6240 degrees of freedom
## Multiple R-squared:  0.8877, Adjusted R-squared:  0.8873
## F-statistic:  2145 on 23 and 6240 DF,  p-value: < 2.2e-16
```

```
#weekday pricing
price_flat<-data.frame(price=rep(5, 24))
price_custom<-data.frame(price_new=c(5,5,5,5,5,6,6,6,5,5,5,5,6,6,6,6,6,5,5,5,5,5,5,5))

coeff <- data.frame(summary(fit6)$coefficients[,1])
coeff$value <- coeff$summary.fit6..coefficients...1.+26.340996
coeff[1,2] <- 26.340996

coeff <- cbind(coeff, price_flat, price_custom)

coeff$rev_flat <- coeff$value*coeff$price
coeff$rev_custom <- coeff$value*coeff$price_new

sum(coeff$rev_custom-coeff$rev_flat)
```

```
## [1] 9880.421
```

```
sum(coeff$rev_flat)
```

```
## [1] 82792.49
```

```
paste(round(sum(coeff$rev_custom-coeff$rev_flat)/sum(coeff$rev_flat)*100, 2),"%")
```

```
## [1] "11.93 %"
```

Qualitative questions - answer these theoretically, no need to implement:

7. What additional data would you like to see in order to answer questions 1-5 more confidently and how would you incorporate it?

More data can be used to enhance this anlaysis. In particular, more recent data would give a better picture of the current landscape for taxi rides. Data for other years outside of 2013 could allow for better estimates of seasonality. Data regarding competitors such as Uber and Lyft would greatly increase the understanding of the evolving industry. Data on other modes of transportation could also used as it can take away from ridership. Via's proprietary data would also greatly enhance this analysis.

8. How might your answer change over time? What Via data would you monitor to ensure the proposed expansion was a good business decision?

The landscape of the transportation industry will inevitably change over time and it is critical that information be collected and leveraged to guide the decisions of the company.

Via data on passenger volume, pickup times, pickup location, and revenue generated can all help determine whether or not the expansion was a good decision. Sufficient time should be given to monitor the expansion project as it might take time to scale and hit critical mass before becoming profitable.