Analyzing Citi Bike Usage in New York

CS 145 Project (Colab Link: https://colab.research.google.com/drive/1k7xGTh9_fOkNzC0DJX5l4merK6P_FrYm)

Milan Zhou

Goal

The purpose of this project is to answer the following questions about CitiBike.

- What is the optimal time to redistribute bikes?
- Which stations should CitiBike are most strained on their resources?
- What relationships does taxi ride activity have with bike station activity?
- 311 requests and bike station activity?
- Motor-vehicle accidents and bike station activity?
- Trees and bike station activity?

Finally, we will build a linear regression to predict the number of outgoing trips a bike station will have on a given day of the week.

Setting Up BigQuery and Dependencies

The following cells authenticate the BigQuery account and import the needed dependencies for displaying our plots

```
# Run this cell to authenticate yourself to BigQuery
from google.colab import auth
auth.authenticate_user()
project_id = "939858855469"
```

```
# Initialize BigQuery client
from google.cloud import bigquery
client = bigquery.Client(project=project_id)  # pass in your projectid
```

Saved successfully!

```
import matplotlib.pyplot as plt
import numpy as np

%matplotlib inline
plt.style.use('seaborn-dark')
```

Datasets

bigquery-public-data:new_york.citibike_trips

The primary dataset we will be using is New York's Citi Bike database. The database is 4.58 GB large and contains the following useful information about each trip:

- Trip Duration (in seconds)
- Start Time
- Start Station (id, name, location)
- End Station (id, name, location)
- User Type (customer or subscriber)
- Birth Year (can calculate user age)
- Gender

Roughly 10% of the data is "invalid" in some way or another. It turns out we can filter out all "invalid" trips simply by checking just three fields (and thus save money):

- start station id != null
- birth year != null
- start/end station latitude/longitude != 0

bigquery-public-data:new_york.citibike_stations

The citibike_stations dataset supplements our citibike_trips dataset with additional information about a trip. For example, we can get

- the name of the station at which a trip starts or ends
- the number of docks available at a station

We match bike trips with bike stations based on the start or end station id of a ride. The dataset is small. It has just 817 stations and is 100 KB large. We can potentially use other information about stations as features in our ML models later as well.

bigquery-public-data:new_york.tlc_yellow_trips_(2013-2016)

We will incorporate the 2013- 2016 yellow taxi trip database in our analysis. We will largely compare the activity of bike rides at a station to the the activity of taxi rides in that same area, to 3 decimal precision of latitude and longitude, which is accurate to about 100 meters. In addition to taxi activity, we will also analyze how the speed and distance of taxi trips originating from a bike station correlates with that bike station's activity.

bigquery-public-data:new_york.311_service_requests

This 10 GB table has 16,983,314 rows. We will analyze whether there is a relationship between the number of 311 service requests at a bike station and the bike activity at that station. Like before, we restrict our date range to 2016, and we use 3 decimal points of precision.

bigquery-public-data:new_york.nypd_mv_collisions

200 MB table, with 1,000,000 rows, that details all motor-vehicle collisions between 2016 and 2018. We use this in much the same way we use the 311 service requests table.

bigquery-public-data:new_york.tree_census_2015

200 MB table with 700,000 rows that counts all trees in new york. We analyze whether the number of trees near a bike station is correlated with that station's activity.

bigquery-public-data:noaa_gsod.gsod(2013-2016)

We will use the National Oceanic and Atmospheric Administration datasets for weather data in NY. The datasets provide useful features for our ML models, including rain, temperature, and snow.

Getting Started

Let us visualize a few details about the Citi Bike dataset first.

Sample Rows

```
%%bigquery --project $project id
```

Saved successfully!

```
      , start_station_id start_sid
      , end_station_id end_sid
      , start_station_latitude start_lat
      , start_station_longitude start_lon
      , end_station_latitude end_lat
      , end_station_longitude end_lon
      , usertype
      , birth_year
      , gender

FROM `bigquery-public-data.new_york.citibike_trips`
WHERE start_station_id IS NOT NULL
  AND birth_year IS NOT NULL
  AND end_station_longitude != 0
LIMIT 5
```

	tripduration	starttime	start_sid	end_sid	start_lat	start_lon	end_lat	end_lon	usertype	birth
0	722	2016-06-20 06:17:05+00:00	3236	520	40.758985	-73.993800	40.759923	-73.976485	Subscriber	
1	152978	2016-02-22 19:22:07+00:00	250	3019	40.724561	-73.995653	40.716633	-73.981933	Subscriber	
2	100569	2015-12-17 10:24:19+00:00	417	3019	40.712912	-74.010202	40.716633	-73.981933	Subscriber	
3	156	2015-10-07	211	3019	40.717227	-73.983091	40.716633	-73.981933	Subscriber	

Range and Size

```
%%bigquery --project $project_id
SELECT min(starttime) Earliest_Trip, max(stoptime) Latest_Trip, COUNT(*) Total_Trips
FROM `bigquery-public-data.new_york.citibike_trips`
WHERE start_station_id IS NOT NULL
  AND birth_year IS NOT NULL
  AND end_station_longitude != 0
```

	Earliest_Trip	Latest_Trip	Total_Trips
0	2013-07-01 00:01:04+00:00	2016-10-02 00:07:12+00:00	29263661

When comparing the bike dataset to other datasets, we will just use one year's worth of data to avoid having to continually join multiple taxi datasets in later sections. Since 2016 is incomplete, we will use 2015 as the next best range. When not comparing to another dataset, we will use the entire bike dataset.

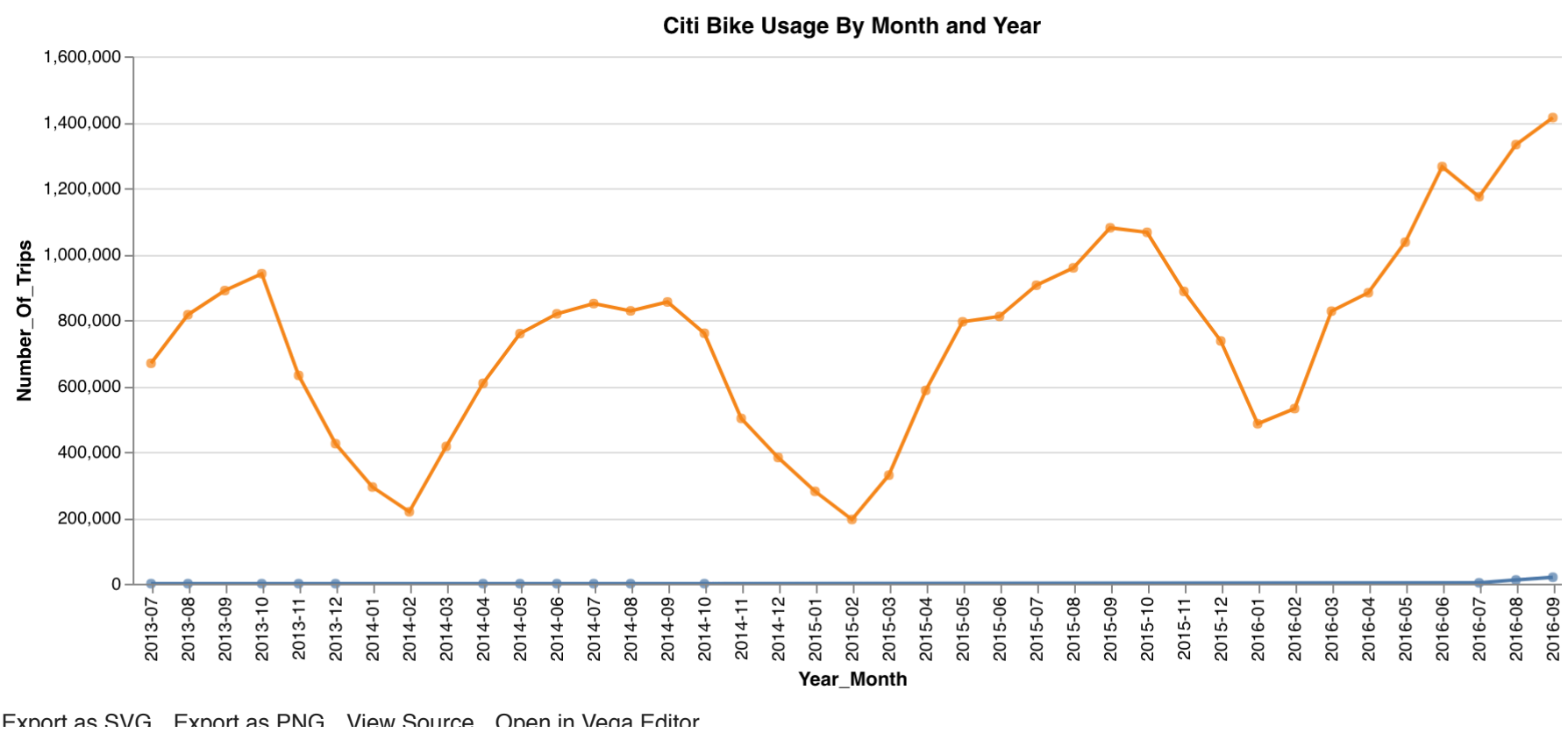
Usage Over Time

```
%%bigquery --project $project_id popularity_time

SELECT COUNT(*) Number_Of_Trips
     , FORMAT_TIMESTAMP("%Y-%m", starttime) as Year_Month
     , usertype
FROM `bigquery-public-data.new_york.citibike_trips`
WHERE start_station_id IS NOT NULL
  AND birth_year IS NOT NULL
  AND end_station_longitude != 0
GROUP BY Year_Month, usertype
```

	21	586173	2015-04	Subscriber
	22	815902	2013-08	Subscriber
	23	531045	**2016**-02	Subscriber
	24	1035946	**2016**-05	Subscriber
	25	759623	2014-10	Subscriber
	26	279924	2015-01	Subscriber
	27	958043	2015-08	Subscriber
	28	810827	2015-06	Subscriber
	29	940266	2013-10	Subscriber
	30	886644	2015-11	Subscriber
	31	889190	2013-09	Subscriber
	32	1331826	**2016**-08	Subscriber
	33	328973	2015-03	Subscriber
	34	293146	2014-01	Subscriber
	35	736248	2015-12	Subscriber
	36	827548	2014-08	Subscriber
	37	[illegible]	[illegible]4-11	Subscriber
	38	[illegible]	[illegible]4-09	Subscriber
	39	758521	2014-05	Subscriber
	40	668501	2013-07	Subscriber
	41	794434	2015-05	Subscriber
	42	607733	2014-04	Subscriber
	43	1265110	**2016**-06	Subscriber
	44	484932	**2016**-01	Subscriber
	45	382573	2014-12	Subscriber
	46	631801	2013-11	Subscriber
	47	218019	2014-02	Subscriber
	48	1173897	**2016**-07	Subscriber
	49	849704	2014-07	Subscriber
	50	416235	2014-03	Subscriber
	51	818640	2014-06	Subscriber
	52	194663	2015-02	Subscriber

Saved successfully!

```
import altair as alt

alt.Chart(popularity_time).mark_line(
).encode(
    x=alt.X('Year_Month:N'),
    y=alt.Y('Number_Of_Trips:Q'),
    color = "usertype",
    tooltip = ["Number_Of_Trips", "usertype", "Year_Month"]
).properties(
    title="Citi Bike Usage By Month and Year"
) + alt.Chart(popularity_time).mark_circle(
).encode(
    x=alt.X('Year_Month:N'),
    y=alt.Y('Number_Of_Trips:Q'),
    color = "usertype",
    tooltip = ["Number_Of_Trips", "usertype", "Year_Month"]
).properties(
    title="Citi Bike Usage By Month and Year"
)
```



Export as SVG Export as PNG View Source Open in Vega Editor

19447 appears in multiple Year_Months for the Customers line. Possibly a logging or duplication error?

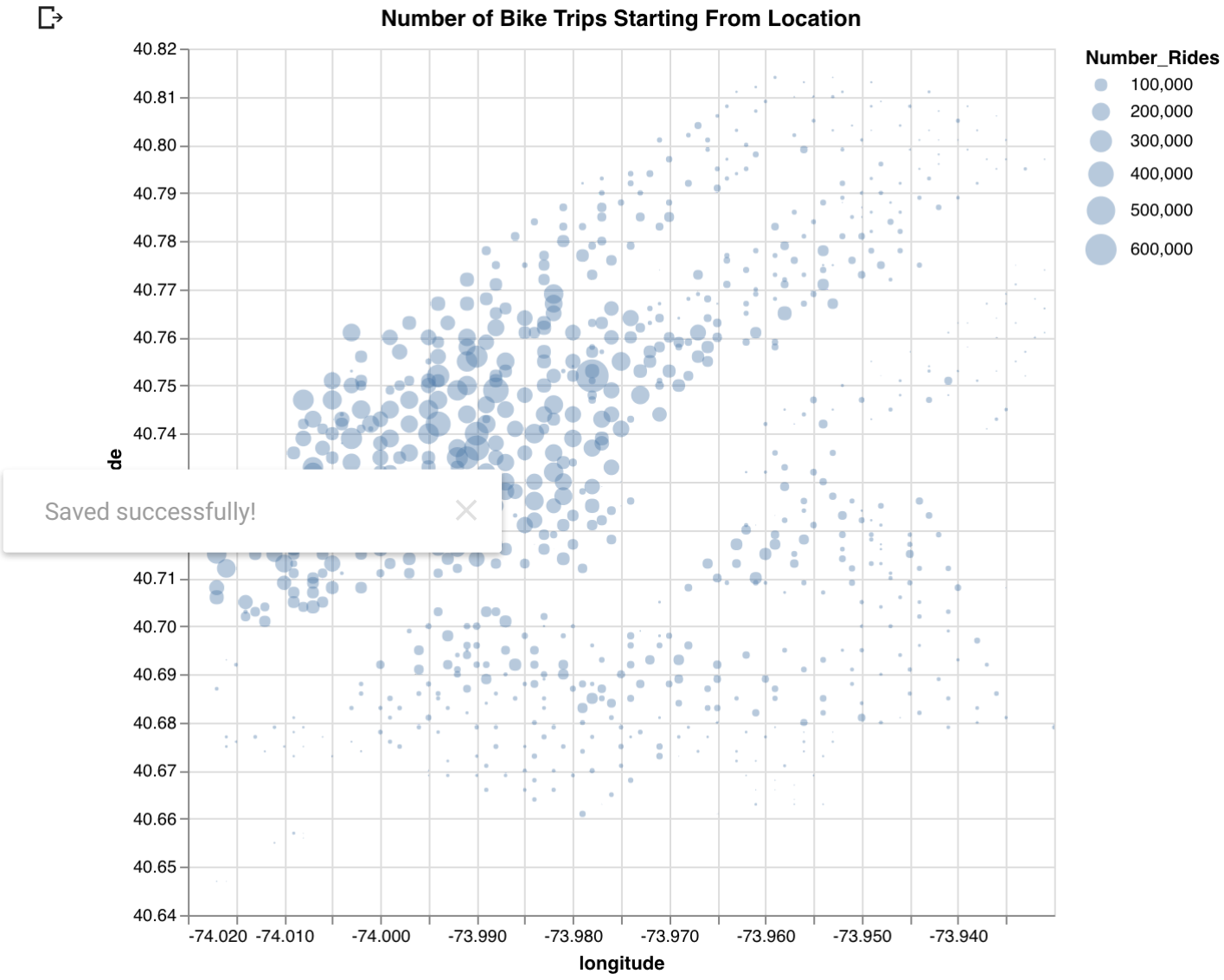
Usage Location

```
%%bigquery --project $project_id bike_trips_from

SELECT ROUND(start_station_latitude,3) latitude
     , ROUND(start_station_longitude,3) longitude
     , COUNT(*) AS Number_Rides
     , "Bike" Mode
FROM `bigquery-public-data.new_york_citibike.citibike_trips`
WHERE start_station_id IS NOT NULL
  AND birth_year IS NOT NULL
  AND end_station_longitude != 0
GROUP BY latitude, longitude
```

Saved successfully!

...	...	...	...	...
841	40.763	-73.983	118273	Bike
842	40.744	-73.983	166418	Bike
843	40.702	-73.983	30656	Bike
844	40.762	-73.983	125527	Bike

```
import altair as alt

d = (-74.02, -73.93)
r = (40.66, 40.81)

bike_trips_from_graph = alt.Chart(bike_trips_from).mark_circle(
    opacity = .4,
    clip=True,
).encode(
    x=alt.X('longitude:Q', scale=alt.Scale(domain=d)),
    y=alt.Y('latitude:Q', scale=alt.Scale(domain=r)),
    size = "Number_Rides:Q",
    tooltip = ["Mode", "Number_Rides"]
).properties(
    width=500,
    height=500,
    title="Number of Bike Trips Starting From Location"
)

bike_trips_from_graph
```



Saved successfully!

Export as SVG Export as PNG View Source Open in Vega Editor

Citi Bike stations cover Manhattan, Brooklyn, and Queens very well. Manhattan has the highest activity of bike rides.

Optimal Bike Redistribution Time

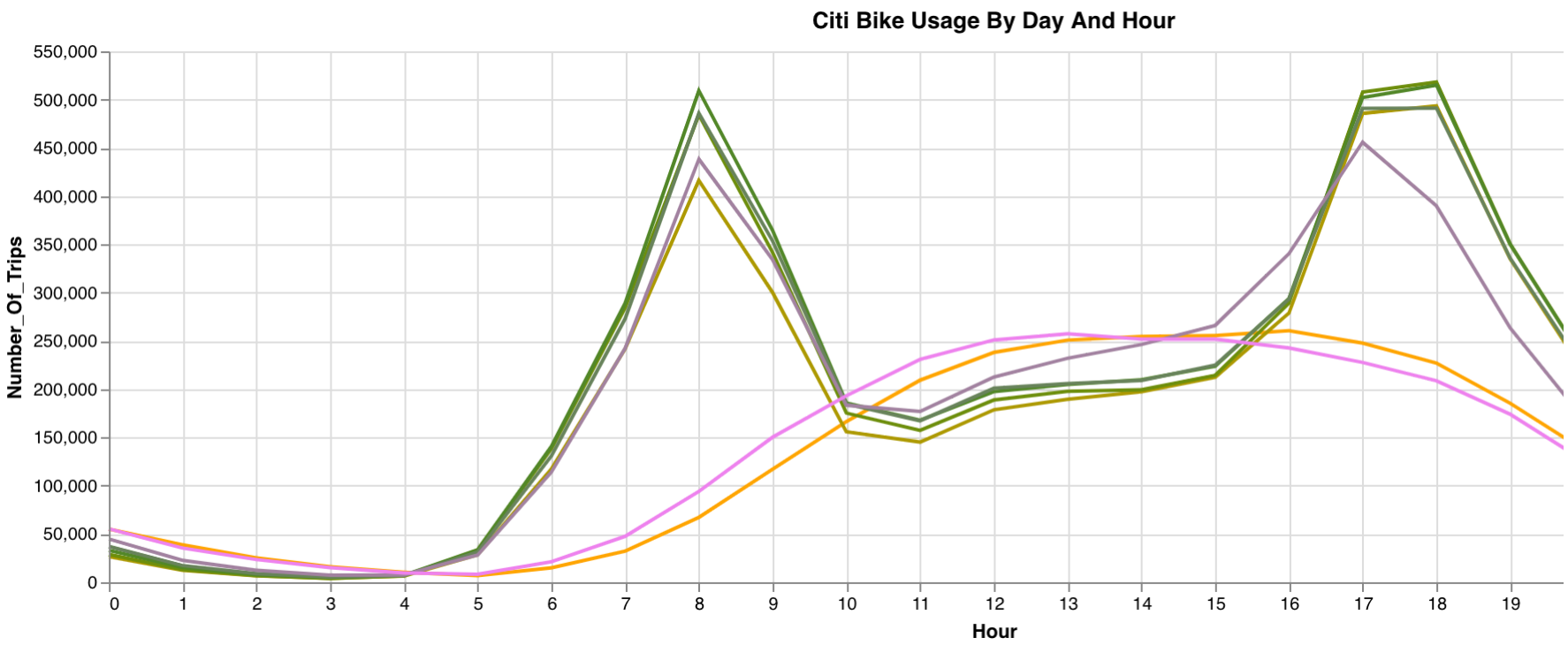
The first optimization problem that we will explore is when to redistribute bikes across the stations. To answer this, we will consider the optimal redistribution time to be that when there is the least bike activity. We can approximate overall bike activity by examining the hour in which the fewest bike rides begin. Then, we aggregate by the trips by day of the week.

```
%%bigquery --project $project_id week_hour_usage

SELECT EXTRACT(HOUR FROM starttime) Hour, EXTRACT(DAYOFWEEK FROM starttime) Day_Of_Week, COUNT(*) Number_Of_Trips
FROM `bigquery-public-data.new_york.citibike_trips`
WHERE start_station_id IS NOT NULL
  AND birth_year IS NOT NULL
  AND end_station_longitude != 0
GROUP BY Hour, Day_Of_Week
```

...	...	...	...
138	18	6	389887
139	9	6	334083
140	21	6	118269
141	10	6	183314
142	23	6	76918
143	0	6	44755
144	23	7	70968
145	4	7	9616
146	18	7	208721
147	11	7	230826
148	12	7	251005
149	3	7	15031
150	21	7	96310
151	0	7	55373
152	9	7	150287
153	15	7	251829
154	16	7	242718
155	5	7	8348
156	22	7	84372
157	1	7	35402
158	14	7	251825
159	17	7	227679
160	7	7	47473
161	2	7	21155
162	10	7	193268
163	20	7	125893
164	8	7	94052

Saved successfully!

```
import altair as alt

alt.Chart(week_hour_usage).mark_line(
).encode(
    x=alt.X('Hour:Q', axis = alt.Axis(values = [n for n in range(24)])),
    y=alt.Y('Number_Of_Trips:Q'),
    tooltip=['Day_Of_Week', "Number_Of_Trips"],
    color = alt.Color("Day_Of_Week:T", scale=alt.Scale(range=["orange", "green","violet"]))
).properties(
    title="Citi Bike Usage By Day And Hour",
    width=1000
)
```



Export as SVG Export as PNG View Source Open in Vega Editor

The least disruptive time to redistribute bikes would be 3 - 4 am on weekdays and 4-5 am on weekends (week starts on Sunday). If redistribution is necessary in the middle of the day, 10 - 11 am is optimal for weekdays, while there is no clear local minima for the weekends. However, there is likely some variance to this optimal redistribution time. It is likely that these times vary with other conditions (e.g. temperature, holidays, etc.).

Net Flow of Bike Stations

Over the course of a day, bikes move from station to station. We will explore which stations see the greatest disparity between incoming and outgoing bikes. We aggregate data by the location and the day of the week. We use median rather than mean becaause, each aggregate group contains only 52 data points, which is relatively small sample size.

Net Flow

```
%%bigquery --project $project_id bike_traffic

SELECT name Name
     , latitude Latitude
     , longitude Longitude
     , EXTRACT(DAYOFWEEK FROM a.Date) Day_Of_Week
     , APPROX_QUANTILES(Number_Incoming,1000)[OFFSET(500)] Average_Number_Incoming
     , APPROX_QUANTILES(Number_Outgoing,1000)[OFFSET(500)] Average_Number_Outgoing
     , APPROX_QUANTILES(Number_Incoming,1000)[OFFSET(500)] - APPROX_QUANTILES(Number_Outgoing,1000)[OFFSET(500)] Av
     , capacity Capacity
     , num_docks_available
     , num_bikes_available
     , IF(ABS(APPROX_QUANTILES(Number_Incoming,1000)[OFFSET(500)] - APPROX_QUANTILES(Number_Outgoing,1000)[OFFSET(5
FROM
  (
    SELECT station_id
         , COUNT(*) Number_Outgoing
         , EXTRACT(DATE FROM starttime) Date
    FROM `bigquery-public-data.new_york.citibike_trips`
       , `bigquery-public-data.new_york.citibike_stations`
    WHERE start_station_id = station_id
      AND start_station_id IS NOT NULL
      AND birth_year IS NOT NULL
      AND end_station_longitude != 0
    GROUP BY station_id, Date
  ) a,
  (
    SELECT station_id
         , COUNT(*) Number_Incoming
         , EXTRACT(DATE FROM starttime) Date
    FROM `bigquery-public-data.new_york.citibike_trips`
       , `bigquery-public-data.new_york.citibike_stations`
    WHERE end_station_id = station_id
      AND start_station_id IS NOT NULL
      AND birth_year IS NOT NULL
      AND end_station_longitude != 0
    GROUP BY station_id, Date
  ) b
```

Saved successfully!

```
       , longitude
       , name
       , capacity
       , num_docks_available
       , num_bikes_available
  FROM `bigquery-public-data.new_york.citibike_stations`
  ) c
WHERE a.station_id = b.station_id
  AND a.station_id = c.station_id
  AND a.Date = b.Date
  AND capacity != 0
GROUP BY name, latitude, longitude, Day_Of_Week, capacity, num_docks_available, num_bikes_available
```

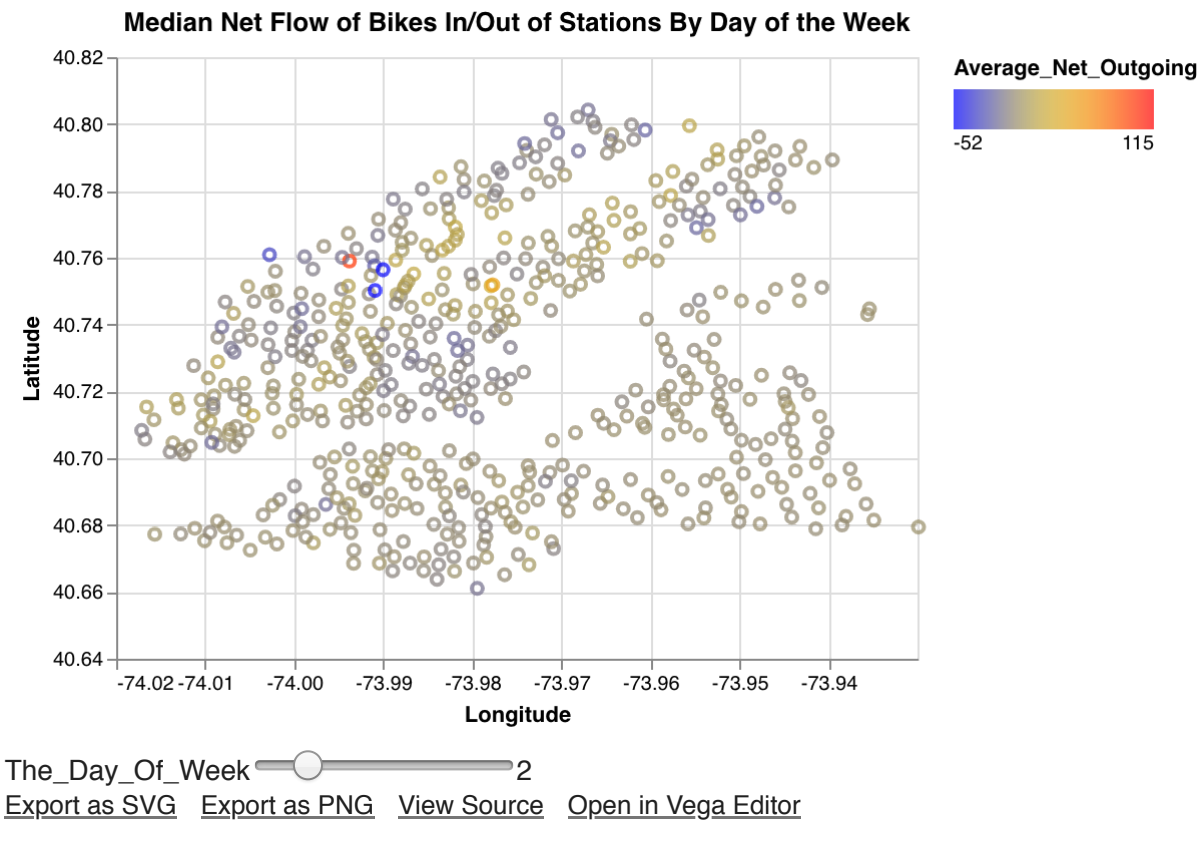
	Name	Latitude	Longitude	Day_Of_Week	Average_Number_Incoming	Average_Number_Outgoing	Average_
0	3 St & 3 Ave	40.675070	-73.987752	3	29	27	
1	3 St & 3 Ave	40.675070	-73.987752	1	52	50	
2	3 St & 3 Ave	40.675070	-73.987752	5	29	28	
3	3 St & 3 Ave	40.675070	-73.987752	2	37	34	
4	3 St & 3 Ave	40.675070	-73.987752	7	37	32	
5	3 St & 3 Ave	40.675070	-73.987752	4	26	20	
6	3 St & 3 Ave	40.675070	-73.987752	6	32	24	
7	Great Jones St	40.727434	-73.993790	3	143	137	

```
# Find the min and max values to restrict domain of colors
print(min(bike_traffic["Average_Net_Outgoing"]), max(bike_traffic["Average_Net_Outgoing"]))
```

```
-52 115
```

```
import altair as alt

d = (-74.02, -73.93)
r = (40.66, 40.81)

select_day = alt.selection_single(name="The", fields=['Day_Of_Week'], bind=alt.binding_range(min=1, max=7, step=1))

alt.Chart(bike_traffic).mark_point(
).encode( x=alt.X('Longitude:Q', scale = alt.Scale(domain = d))
        , y=alt.Y('Latitude:Q', scale = alt.Scale(domain = r))
        , color = alt.Color("Average_Net_Outgoing:Q", scale = alt.Scale(domain = (-52,115), range=["blue", "yellow"
        , tooltip=['Name', "Average_Number_Incoming", "Average_Number_Outgoing", "Average_Net_Outgoing"]
).add_selection(
    select_day
).transform_filter(
    select_day
).properties(
    title="Median Net Flow of Bikes In/Out of Stations By Day of the Week",
)
```

Saved successfully!



We are mostly concerned with the extremes on the spectrum (red and blue nodes). For most stations, the net difference between the median number of incoming bikes and the median number of outgoing bikes is small. However, bike stations like Pershing Square North loses 115 bikes over the course of an average Wednesday! On the other hand, the bike station at W 33 St & 7 Ave gains 52 bikes on an average Monday.

While we consider the raw number of the net movement, we should also keep in mind the different resources a station possesses to accomdate for the gain or loss of bikes.

Strain On Station Resources By Day Of the Week

Now, let us compare the net flow rate of bikes to the station's docking capacity. This will allow us to get a measure of how much strain is placed on the station's resources (both in terms of providing bikes and docking incoming bikes).

Unfortunately, we do not have past real time data about how many bikes were at a station on each day. Thus, we must define a station to be "strained" if the station's net flow of bikes is >= half of its capacity.

```
%%bigquery --project $project_id bike_strain

  SELECT station_id
       , name
       , capacity
       , num_bikes_available
       , num_docks_available
  FROM `bigquery-public-data.new_york.citibike_stations`
  WHERE capacity != 0
```

	station_id	name	capacity	num_bikes_available	num_docks_available
0	3476	Norman Ave & Leonard St	3	0	0
1	3548	Pacific St & Bedford Ave	25	0	0
2	3436	Greenwich St & Hubert St	31	0	0
3	316	Fulton St & William St	43	0	0
4	3225	Baldwin at Montgomery	14	0	13
5	3281	Leonard Gordon Park	14	0	14
6	3267	Morris Canal	14	0	14
7	3272	Jersey & 3rd	14	0	14
8	400	Pitt St & Stanton St	15	0	15
9	3198	Heights Elevator	18	0	17
10	3210	Pershing Field	18	0	18
11	3196	Riverview Park	18	0	18
12	3049	Cambridge Pl & Gates Ave	18	0	18
13	3430	Richardson St & N Henry St	19	0	19
14	3564	21 St & 36 Ave	19	0	19
15	3590	Carroll St & Franklin Ave	19	0	19
16	3048	Putnam Ave & Nostrand Ave	19	0	19
17	3653	31 St & 35 Ave	19	0	19
		Crescent St & 34 Ave	19	0	19
		21 St & 38 Ave	19	0	19
20	3525	23 Ave & 27 St	21	0	21
21	3546	Pacific St & Classon Ave	21	0	21
22	3563	28 St & 36 Ave	21	0	21
23	3047	Halsey St & Tompkins Ave	21	0	21
24	3356	Amsterdam Ave & W 66 St	21	0	21
25	3578	Park Pl & Franklin Ave	21	0	21
26	3212	Christ Hospital	22	0	22
27	3211	Newark Ave	22	0	22
28	3192	Liberty Light Rail	22	0	22
29	3194	McGinley Square	22	0	22
...	...	...	...	...	...

Saved successfully!

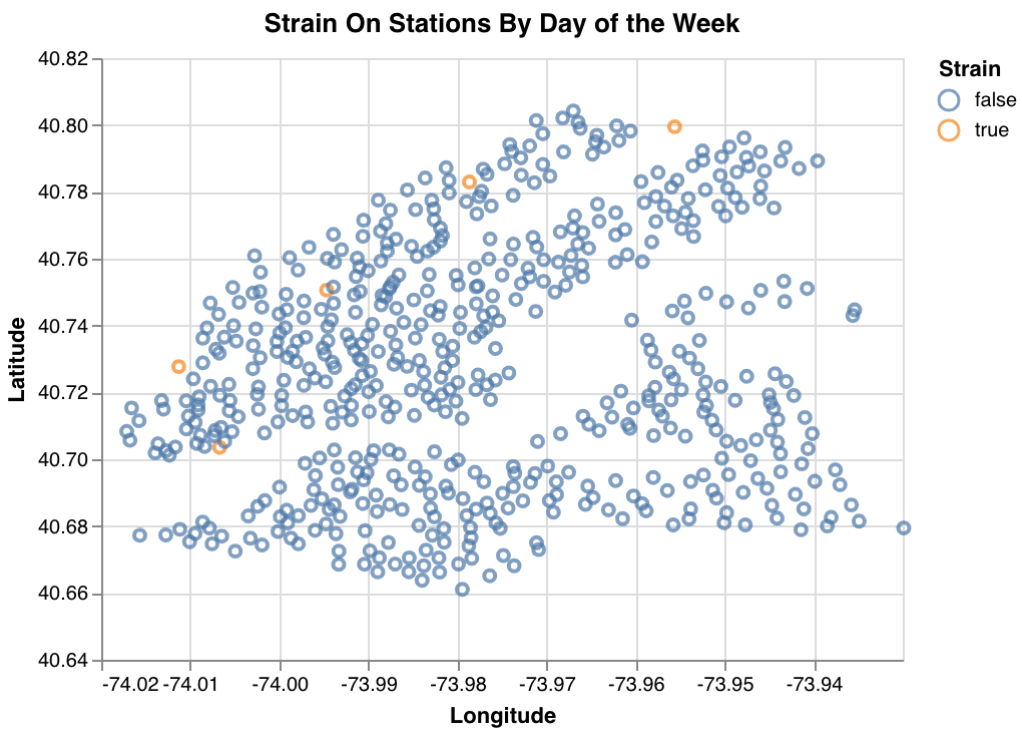
```
import statistics

print(statistics.mean(bike_strain["num_bikes_available"]), statistics.mean(bike_strain["num_docks_available"]), sta
```

```
6.0 19.0 29.0
```

Examining the mean bike and dock availabilities, updated on Sunday around 2 pm EST, shows that the definition of strain is more or less a good estimate of bike/dock availability. (Remember that 2pm Sunday is peak bike usage time so we expect fewer than normal bikes at a station)

```
import altair as alt

d = (-74.02, -73.93)
r = (40.66, 40.81)

select_day = alt.selection_single(name="The", fields=['Day_Of_Week'], bind=alt.binding_range(min=1, max=7, step=1))

alt.Chart(bike_traffic).mark_point(
).encode( x=alt.X('Longitude:Q', scale = alt.Scale(domain = d))
        , y=alt.Y('Latitude:Q', scale = alt.Scale(domain = r))
        , color = alt.Color("Strain")
        , tooltip=['Name', "Average_Net_Incoming:Q", "Capacity"]
).add_selection(
    select_day
).transform_filter(
```

```
    select_day
).properties(
    title="Strain On Stations By Day of the Week",
)
```



The_Day_Of_Week 1

Export as SVG Export as PNG View Source Open in Vega Editor

We notice that that the stations with the fewest or largest net flows of bikes are largely the same stations with the smallest or largest strains. On some days, stations have 3 and some almost 4 times as many incoming or outgoing bikes than available docks! For example, W 42 St & Dryer Avenue on a Wednesday has 3.5 times more net incoming bikes than available docks! City planners might want to consider adding additional docking stations and bikes to the blue and orange stations to compensate for the high traffic at those stations!

Taxi Ride Activity and Biking Activity

In this section, we explore what relation taxi rides might have to the usage of Citi Bike.

Ignoring taxi rides

Saved successfully!

now many of the taxi rides will be invalid. A ride may be invalid for the following reasons:

- Invalid pickup/dropoff location
- invalid price
- invalid pickup or dropoff time

Since the taxi datasets are large and we are dealing with percentages, we look just at the 2016 dataset.

```
%%bigquery --project $project_id taxi_invalid_trips

SELECT COUNT(*) / (SELECT COUNT(*) FROM `bigquery-public-data.new_york.tlc_yellow_trips_2016`) * 100 Percent_Taxi_T
FROM
`bigquery-public-data.new_york.tlc_yellow_trips_2016`
WHERE pickup_datetime IS NULL
  OR dropoff_datetime IS NULL
  OR pickup_datetime >= dropoff_datetime
  OR trip_distance = 0
  OR pickup_longitude = 0
  OR pickup_latitude = 0
  OR dropoff_longitude = 0
  OR dropoff_latitude = 0
  OR total_amount <= 0
  OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
```

	Percent_Taxi_Trips_Invalid
0	1.523762

Ignoring 1.5% of the taxi trips is nowhere as substantial as the 10% of bike rides, but still nontrivial.

Location of Taxi Trips

Now we visualize the starting location of taxi trips. There is no easy way to do this because we can only plot 5000 points and our query will return more than 5000. To solve this problem, we will randomly sample 5000 taxi pick up points.

```
%%bigquery --project $project_id taxi_trips_from

SELECT pickup_latitude Latitude
     , pickup_longitude Longitude
     , 1 Number_Rides
     , "Taxi" Mode
FROM `bigquery-public-data.new_york.tlc_yellow_trips_2016`
WHERE NOT(pickup_datetime IS NULL
  OR dropoff_datetime IS NULL
  OR pickup_datetime >= dropoff_datetime
  OR trip_distance = 0
  OR pickup_longitude = 0
```

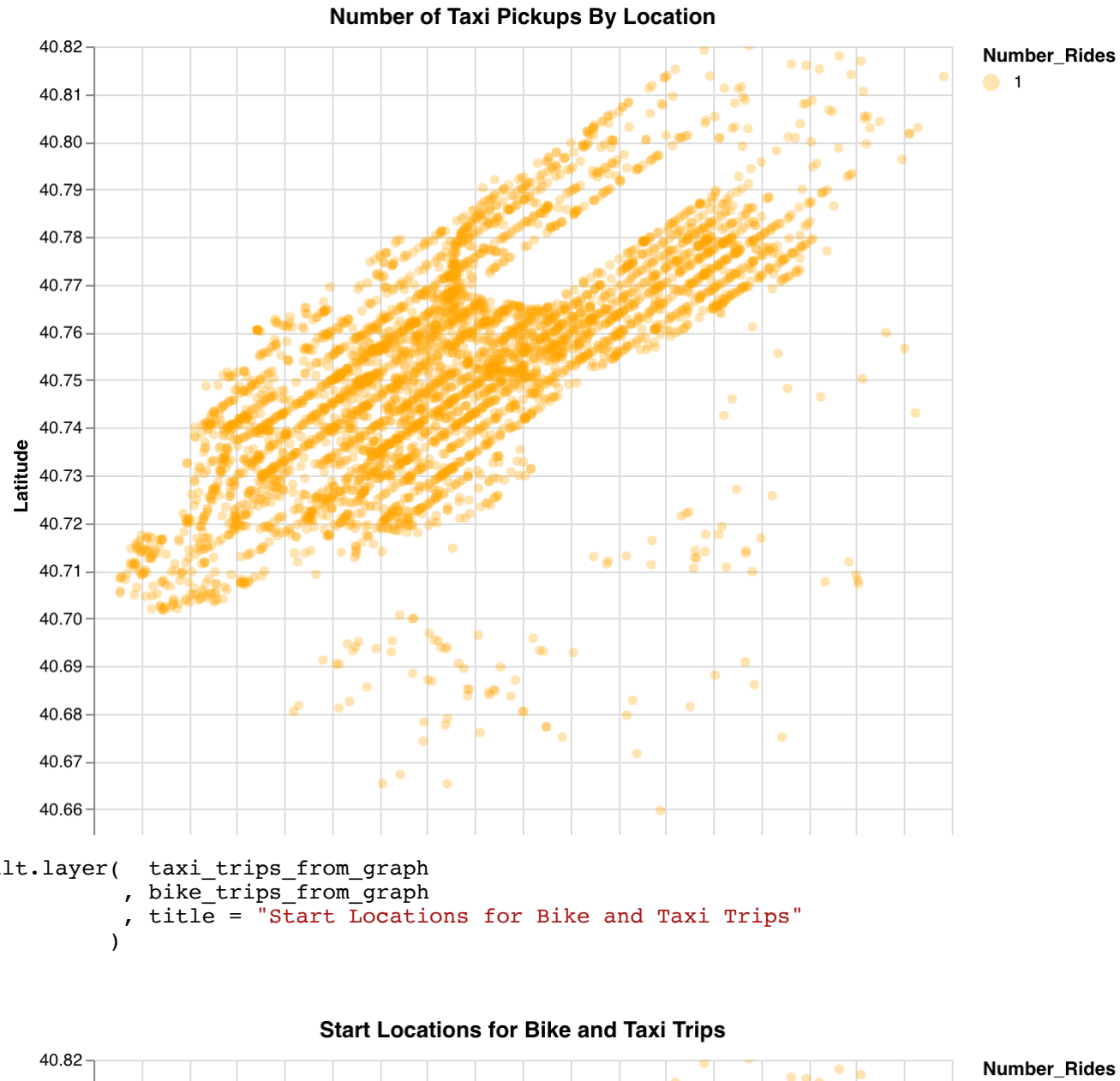
```
  OR pickup_latitude = 0
  OR dropoff_longitude = 0
  OR dropoff_latitude = 0
  OR total_amount <= 0
  OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
  )
ORDER BY RAND()
LIMIT 5000
```

	...	...	...	...	...
4970	40.763786	-73.977638	1	Taxi
4971	40.815208	-73.958992	1	Taxi
4972	40.711819	-74.015610	1	Taxi
4973	40.759411	-73.965202	1	Taxi
4974	40.758629	-73.988571	1	Taxi
4975	40.751049	-73.982437	1	Taxi
4976	40.755142	-74.000343	1	Taxi
4977	40.741455	-73.989616	1	Taxi
4978	40.711208	-74.016113	1	Taxi
4979	40.722649	-73.986099	1	Taxi
4980	40.751411	-73.993958	1	Taxi
4981	40.788040	-73.978378	1	Taxi
4982	40.727234	-73.993439	1	Taxi
4983	40.722630	-73.987244	1	Taxi
4984	40.779652	-73.944595	1	Taxi
4985	40.758804	-73.980515	1	Taxi
4986	40.713810	-74.014397	1	Taxi
4987	40.768570	-73.957718	1	Taxi
4988	40.646851	-73.789917	1	Taxi
4989	40.750832	-73.991219	1	Taxi
4990	40.750832	-73.991219	1	Taxi
4991	40.755623	-73.982338	1	Taxi
4992	40.743484	-73.979958	1	Taxi
4993	40.749821	-73.981377	1	Taxi
4994	40.713715	-73.951614	1	Taxi
4995	40.769951	-73.863350	1	Taxi
4996	40.744160	-73.991653	1	Taxi
4997	40.744019	-73.995819	1	Taxi
4998	40.730289	-73.989799	1	Taxi
4999	40.786858	-73.979347	1	Taxi

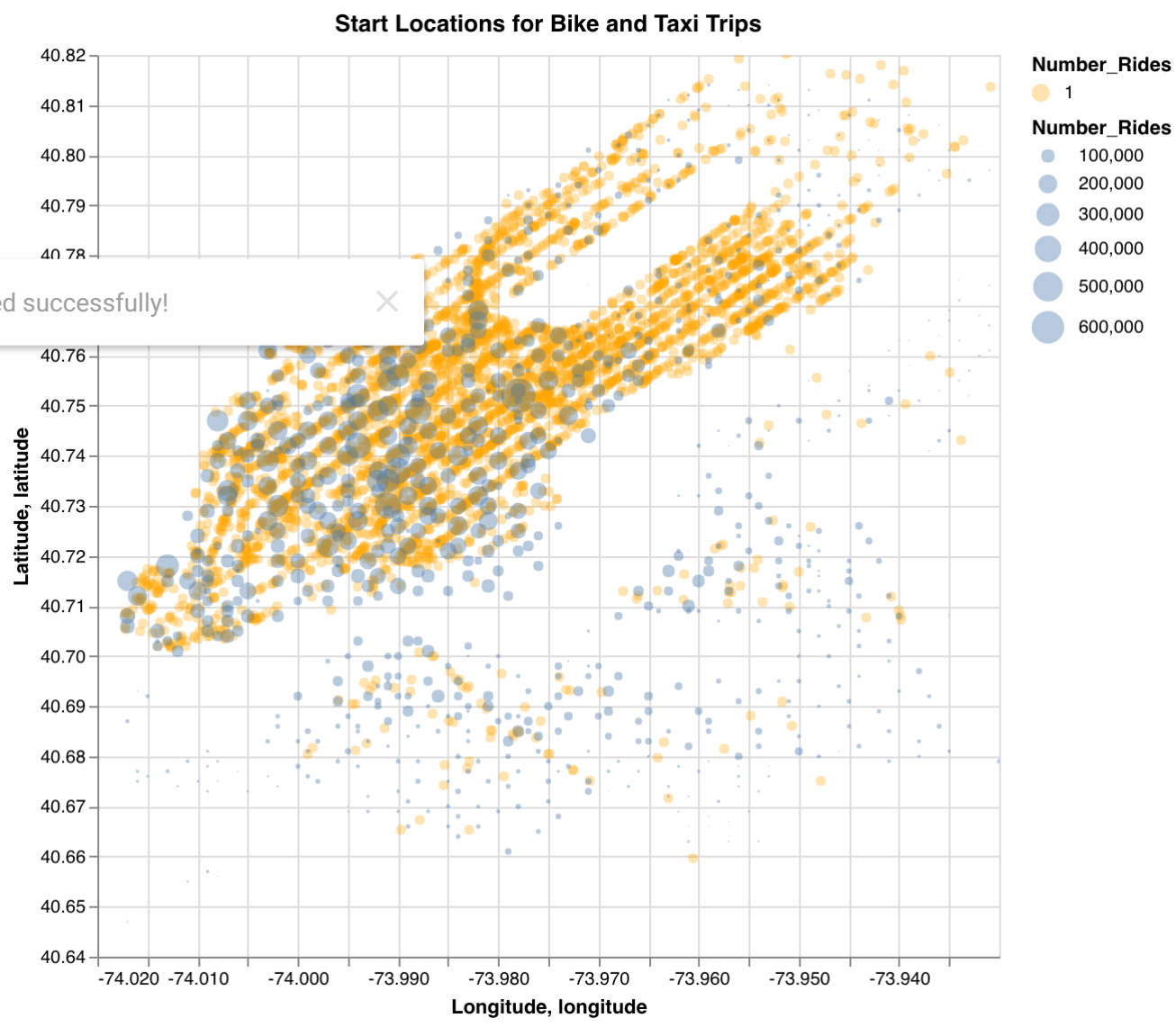
5000 rows × 4 columns

Saved successfully!

```
import altair as alt

d = (-74.02, -73.93)
r = (40.66, 40.81)

taxi_trips_from_graph = alt.Chart(taxi_trips_from).mark_circle(
    color = "orange",
    clip=True,
    opacity = .3
).encode(
    x=alt.X('Longitude:Q', scale=alt.Scale(domain=d)),
    y=alt.Y('Latitude:Q', scale=alt.Scale(domain=r)),
    color= alt.Color("Number_Rides:N", scale = alt.Scale(range=["orange"])),
).properties(
    width=500,
    height=500,
    title="Number of Taxi Pickups By Location"
)

taxi_trips_from_graph
```




```
g = alt.layer(  taxi_trips_from_graph
              , bike_trips_from_graph
              , title = "Start Locations for Bike and Taxi Trips"
             )
g
```



Export as SVG Export as PNG View Source Open in Vega Editor

We see that Manhattan is densely covered for both bikes and taxis. Bike rides in Brooklyn and Queens constitute a sizeable portion of the total bike rides. As a fraction of total rides, taxis seem to be more prevalent in Harlem, Upper East Side, and Upper West Side. We will compare the raw numbers in the next section.

Popularity of Bikes and Taxis When Biking Is Available

We want to see which mode of transportation is more popular in areas where a bike station **is present** (within 100 meters). We will examine the net difference in number of taxi rides and number of bike rides in an area. We do not use ratio between the two, because in many situations, we have only 1 taxi ride in an area of high bike activity or vice versa, which makes the graph not so helpful. Furthermore, we use just 2016 data to get an overall impression of the data.

```
%%bigquery --project $project_id bike_taxi_ratio

SELECT name Name
     , a.latitude Latitude
     , a.longitude Longitude
     , a.Day_Of_Week
     , Number_Taxi_Rides
```

```
      , Number_Bike_Rides
      , ABS(Number_Taxi_Rides - Number_Bike_Rides) Net_Difference
      , IF(Number_Taxi_Rides > Number_Bike_Rides, "Taxi > Bike", "Bike > Taxi") Count
FROM
(
SELECT ROUND(pickup_latitude,3) latitude
      , ROUND(pickup_longitude,3) longitude
      , COUNT(*) Number_Taxi_Rides
      , EXTRACT(DAYOFWEEK FROM pickup_datetime) Day_Of_Week
FROM `bigquery-public-data.new_york.tlc_yellow_trips_2015`
WHERE NOT(pickup_datetime IS NULL
  OR dropoff_datetime IS NULL
  OR pickup_datetime >= dropoff_datetime
  OR trip_distance = 0
  OR pickup_longitude = 0
  OR pickup_latitude = 0
  OR dropoff_longitude = 0
  OR dropoff_latitude = 0
  OR total_amount <= 0
  OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
  )
GROUP BY latitude, longitude, Day_Of_Week
) a,
(
SELECT ROUND(start_station_latitude, 3) latitude
      , ROUND(start_station_longitude,3) longitude
      , COUNT(*) AS Number_Bike_Rides
      , EXTRACT(DAYOFWEEK FROM starttime) Day_Of_Week
      , name
FROM `bigquery-public-data.new_york_citibike.citibike_trips`
    , `bigquery-public-data.new_york_citibike.citibike_stations`
WHERE NOT (start_station_id IS NULL
  OR end_station_id IS NULL)
  AND starttime BETWEEN DATETIME(TIMESTAMP "2015-01-01 00:00:00+00") AND DATETIME(TIMESTAMP "2016-01-01 00:00:00+0(
  AND start_station_id = station_id
GROUP BY latitude, longitude, name, Day_Of_Week
) b
WHERE a.latitude = b.latitude
  AND a.longitude = b.longitude
  AND a.Day_Of_Week = B.Day_Of_Week
```

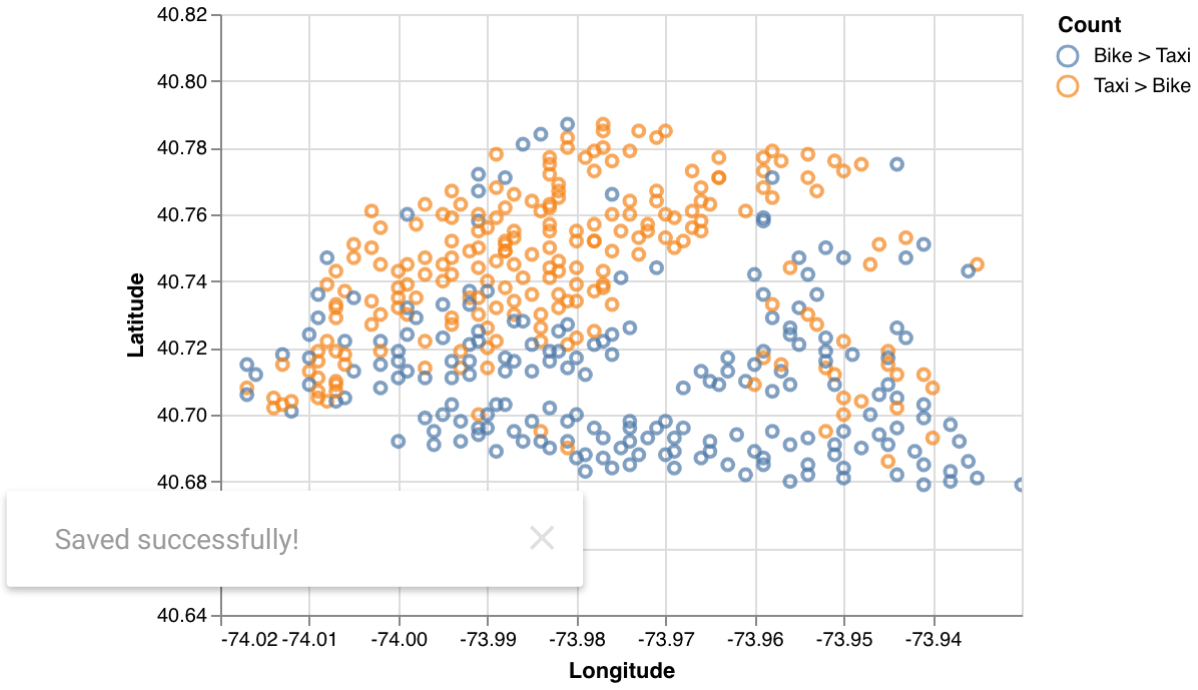
Saved successfully!

```
                Name  Latitude  Longitude  Day_Of_Week  Number_Taxi_Rides  Number_Bike_Rides  Net_Difference  Coun
```

```
print(min(bike_taxi_ratio["Net_Difference"]), max(bike_taxi_ratio["Net_Difference"]))
```

```
2 97592
```

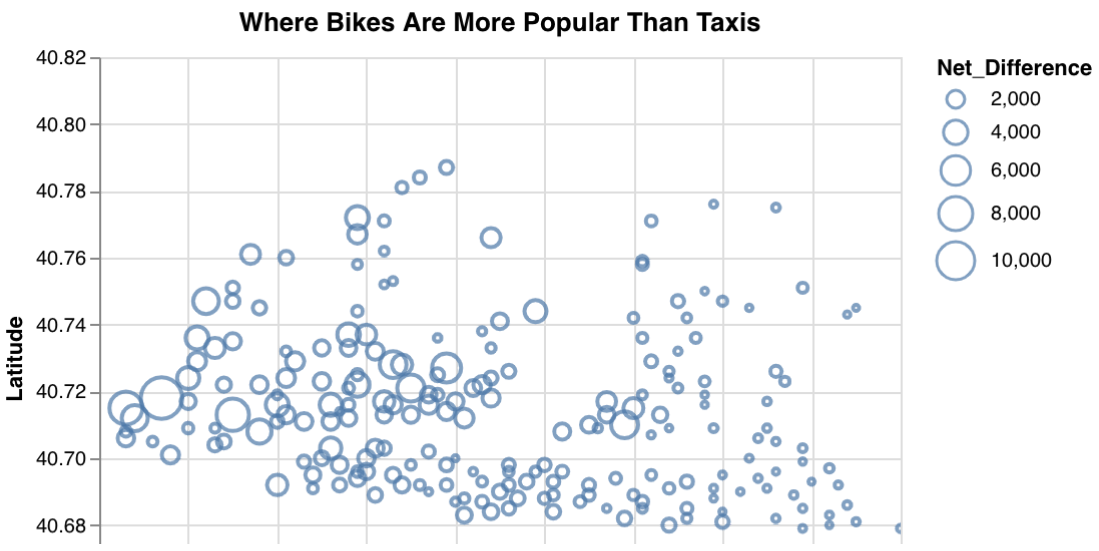
```
import altair as alt

d = (-74.02, -73.93)
r = (40.66, 40.81)


select_day = alt.selection_single(name="The", fields=['Day_Of_Week'], bind=alt.binding_range(min=1, max=7, step=1))

popularity_graph = alt.Chart(bike_taxi_ratio).mark_point(
    clip=True,
).encode(
  x=alt.X('Longitude:Q',scale=alt.Scale(domain=d)),
  y=alt.Y('Latitude:Q',scale=alt.Scale(domain=r)),
  color = alt.Color("Count"),
  tooltip = ['Name', "Number_Bike_Rides","Number_Taxi_Rides"]
).add_selection(
  select_day
).transform_filter(
  select_day
).properties(
  title = "More Popular Mode of Transportation"
)

popularity_graph
```



In areas where a bike station is present, taxis are more common on weekends, while bikes are more common on weekdays. Confirming our earlier suspicions, biking is more common than taxis in Brooklyn and Queens, when a station is present.

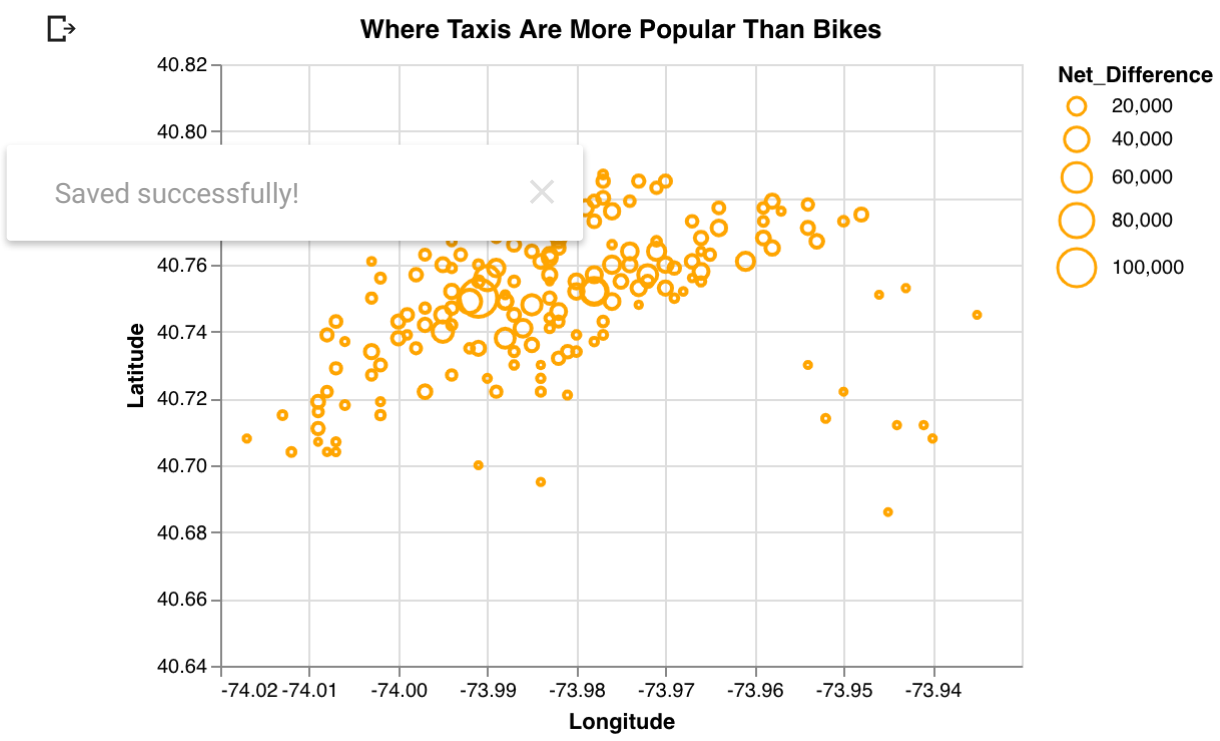
By How Much?

```
import altair as alt

d = (-74.02, -73.93)
r = (40.66, 40.81)


select_day = alt.selection_single(name="The", fields=['Day_Of_Week'], bind=alt.binding_range(min=1, max=7, step=1))

ratio_bike_graph = alt.Chart(bike_taxi_ratio).mark_point(
    clip=True,

).encode(
  x=alt.X('Longitude:Q',scale=alt.Scale(domain=d)),
  y=alt.Y('Latitude:Q',scale=alt.Scale(domain=r)),
  size = alt.Size("Net_Difference", scale= alt.Scale(domain = (1,10000))),
  tooltip = ['Name', "Number_Bike_Rides","Number_Taxi_Rides", "Net_Difference"]
).add_selection(
  select_day
).transform_filter(
  select_day
).properties(
  title = "Where Bikes Are More Popular Than Taxis"
).transform_filter(alt.FieldOneOfPredicate(field='Count', oneOf=['Bike > Taxi']))

ratio_bike_graph
```



The areas where bikes are substantially more popular than taxis are mainly in Queens and Brooklyn and along the East River. On the weekdays in some areas of Manhattan, there is more bike than taxi activity.

```
d = (-74.02, -73.93)
r = (40.66, 40.81)


select_day = alt.selection_single(name="The", fields=['Day_Of_Week'], bind=alt.binding_range(min=1, max=7, step=1))

ratio_taxi_graph = alt.Chart(bike_taxi_ratio).mark_point(
    clip=True,
    color = "orange",
    opacity = 1
).encode(
  x=alt.X('Longitude:Q',scale=alt.Scale(domain=d)),
  y=alt.Y('Latitude:Q',scale=alt.Scale(domain=r)),
  size = alt.Size("Net_Difference:Q",scale=alt.Scale(domain=(1,100000))),
  tooltip=["Name", "Number_Taxi_Rides", "Number_Bike_Rides", "Net_Difference"]

).add_selection(select_day).transform_filter(select_day).properties(
  title = "Where Taxis Are More Popular Than Bikes"
).transform_filter(alt.FieldOneOfPredicate(field='Count', oneOf=['Taxi > Bike']))

ratio_taxi_graph
```



The preceding graphs show the popularity of bike-share stations in Brooklyn, Queens, and Lower Manhattan and the continued domination of taxis in pper East Side and Upper West Side. Interestingly, the day of the week continues to play an important role in the observed patterns. Bikes are more common in Midtown on weekdays but taxis dominate on weekends. Another interesting trend is that bike rides are more common than taxi rides along bodies of water (East River and Hudson Bay). One future consideration might be to analyze how bike usage varies with distance to water.

Raw Counts of Taxi Rides Compared to Bike Usage

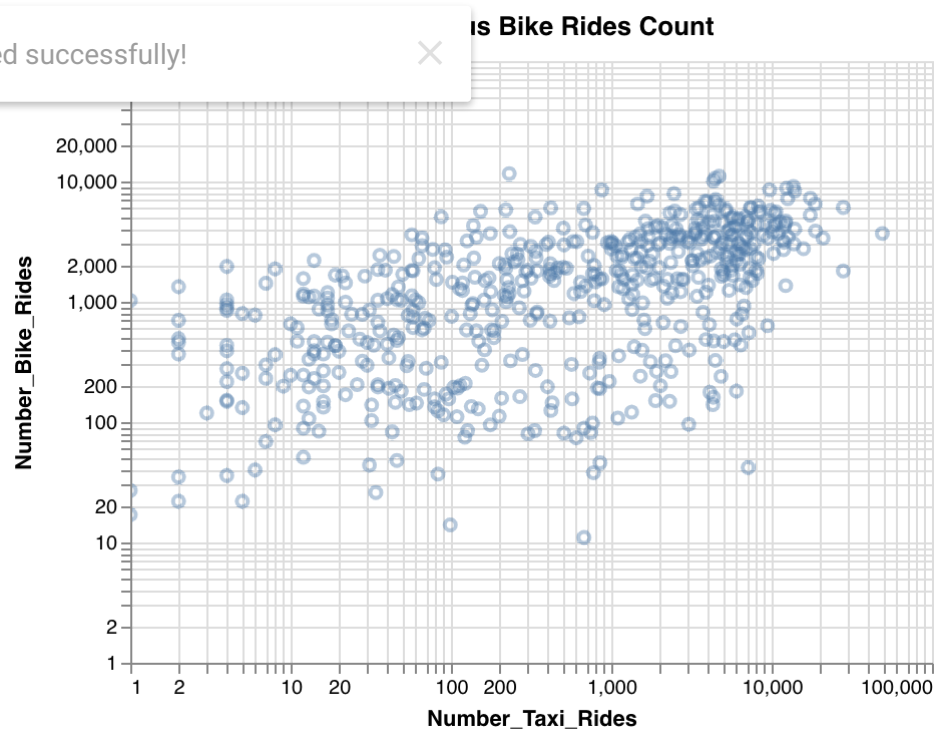
We will now do the same comparison, but ignoring the station location. We want to see if just the raw counts of taxi rides is enough to correlate to bike usage.

```
%%bigquery --project $project_id bike_taxi_ratio

SELECT name Name
     , a.latitude Latitude
     , a.longitude Longitude
     , a.Day_Of_Week
     , Number_Taxi_Rides
     , Number_Bike_Rides
     , ABS(Number_Taxi_Rides - Number_Bike_Rides) Net_Difference
     , IF(Number_Taxi_Rides > Number_Bike_Rides, "Taxi > Bike", "Bike > Taxi") Count
FROM
(
SELECT ROUND(pickup_latitude,3) latitude
     , ROUND(pickup_longitude,3) longitude
     , COUNT(*) Number_Taxi_Rides
     , EXTRACT(DAYOFWEEK FROM pickup_datetime) Day_Of_Week
```

```
FROM `bigquery-public-data.new_york.tlc_yellow_trips_2016`
WHERE NOT(pickup_datetime IS NULL
  OR dropoff_datetime IS NULL
  OR pickup_datetime >= dropoff_datetime
  OR trip_distance = 0
  OR pickup_longitude = 0
  OR pickup_latitude = 0
  OR dropoff_longitude = 0
  OR dropoff_latitude = 0
  OR total_amount <= 0
  OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
  )
GROUP BY latitude, longitude, Day_Of_Week
) a,
(
SELECT ROUND(start_station_latitude, 3) latitude
     , ROUND(start_station_longitude,3) longitude
     , COUNT(*) AS Number_Bike_Rides
     , EXTRACT(DAYOFWEEK FROM starttime) Day_Of_Week
     , name
FROM `bigquery-public-data.new_york_citibike.citibike_trips`
   , `bigquery-public-data.new_york_citibike.citibike_stations`
WHERE NOT (start_station_id IS NULL
  OR end_station_id IS NULL)
  AND starttime BETWEEN DATETIME(TIMESTAMP "2016-01-01 00:00:00+00") AND DATETIME(TIMESTAMP "2017-01-01 00:00:00+0(
  AND start_station_id = station_id
GROUP BY latitude, longitude, name, Day_Of_Week
) b
WHERE a.latitude = b.latitude
  AND a.longitude = b.longitude
  AND a.Day_Of_Week = B.Day_Of_Week
```

```
import altair as alt

select_day = alt.selection_single(name="The", fields=['Day_Of_Week'], bind=alt.binding_range(min=1, max=7, step=1))

popularity_graph = alt.Chart(bike_taxi_ratio).mark_point(
    clip=True,
    opacity = .4
).encode(
  x=alt.X("Number_Taxi_Rides:Q", scale=alt.Scale(domain = (1,100000),type="log")),
  y=alt.Y("Number_Bike_Rides:Q", scale=alt.Scale(domain = (1,100000),type="log")),
  tooltip=["Name", "Number_Taxi_Rides", "Number_Bike_Rides"]
).add_selection(
  select_day
).transform_filter(select_day).properties(
  title = "Taxi Versus Bike Rides Count"
)

popularity_graph
```

Saved successfully!



We notice a large cluster of points between 10,000 - 100,000 taxi rides and 10,000 - 100,000 bike rides. The data is pretty scattered but has remnants of a linear relationship. The number of taxi rides in an area may be a good predictor for Citi Bike usage.

Taxi Distance Compared to Bike Usage

Are people more likely to use Citi Bike when their destination is too far to walk but close enough to bike? Is average taxi trip distance for trips starting in that area a good measure of this (assumes that people are more or less going to the same destinations from that start point)?

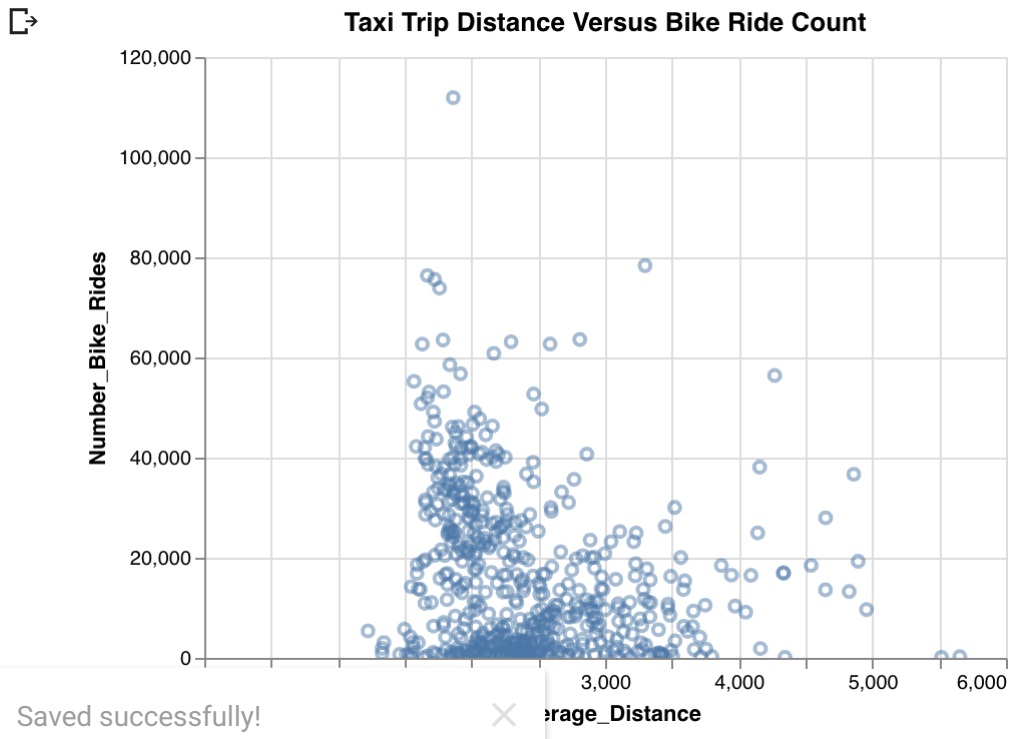
```
%%bigquery --project $project_id bike_taxi_speed

SELECT
  name Name ,
  a.latitude Latitude,
  a.longitude Longitude,
  Average_Distance,
  Average_Speed,
  Number_Bike_Rides
FROM (
  SELECT
    ROUND(pickup_latitude,3) latitude,
    ROUND(pickup_longitude,3) longitude,
    APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude,
```

```
          pickup_latitude),
        ST_GEOGPOINT(dropoff_longitude,
          dropoff_latitude)),1000)[OFFSET(500)] AS Average_Distance,
  APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude,
          pickup_latitude),
        ST_GEOGPOINT(dropoff_longitude,
          dropoff_latitude)) / timestamp_diff(dropoff_datetime, pickup_datetime, second), 1000) [OFFSET(500)] AS
FROM
  `bigquery-public-data.new_york.tlc_yellow_trips_2016`
WHERE
  NOT(pickup_datetime IS NULL
    OR dropoff_datetime IS NULL
    OR pickup_datetime >= dropoff_datetime
    OR trip_distance = 0
    OR pickup_longitude = 0
    OR ABS(pickup_longitude) > 180
    OR pickup_latitude = 0
    OR ABS(pickup_latitude) > 90
    OR dropoff_longitude = 0
    OR ABS(dropoff_longitude) > 180
    OR dropoff_latitude = 0
    OR ABS(dropoff_latitude) > 90
    OR total_amount <= 0
    OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
   )
GROUP BY
  latitude,
  longitude
  ) a,
(
SELECT
  ROUND(start_station_latitude, 3) latitude,
  ROUND(start_station_longitude,3) longitude,
  COUNT(*) AS Number_Bike_Rides,
  name
FROM
    `bigquery-public-data.new_york_citibike.citibike_trips`,
    `bigquery-public-data.new_york_citibike.citibike_stations`
  WHERE
    NOT (start_station_id IS NULL
      OR end_station_id IS NULL)
    AND starttime BETWEEN DATETIME(TIMESTAMP "2016-01-01 00:00:00+00")
    AND DATETIME(TIMESTAMP "2017-01-01 00:00:00+00")
    AND start_station_id = station_id
  GROUP BY
    latitude,
    longitude,
    name ) b
WHERE
  a.latitude = b.latitude
  AND a.longitude = b.longitude
```

Saved successfully!

...	...	...	...	...	...	...
522	E 85 St & York Ave	40.775	-73.948	2515.178290	3.902274	14163
523	E 110 St & Madison Ave	40.796	-73.948	1563.923395	4.317162	2136
524	Verona Pl & Fulton St	40.680	-73.948	2310.149049	3.766119	1634
525	Marcy Ave & Lafayette Ave	40.690	-73.948	2319.917945	4.456041	1910
526	Fulton St & Utica Ave	40.679	-73.930	1601.666909	4.044848	3061

```
import altair as alt

bike_taxi_distance_graph = alt.Chart(bike_taxi_speed).mark_point(
    clip=True,
    opacity = .5
).encode(
  x=alt.X("Average_Distance:Q"),
  y=alt.Y("Number_Bike_Rides:Q"),
  tooltip=["Name", "Average_Distance", "Number_Bike_Rides:Q"]
).properties(
  title = "Taxi Trip Distance Versus Bike Ride Count"
)

bike_taxi_distance_graph
```



Saved successfully!

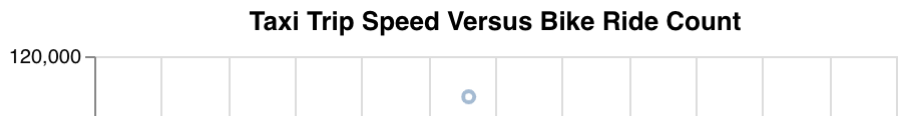
Export as SVG Export as PNG View Source Open in Vega Editor

Interestingly, we do see an inverse correlation between average taxi trip distance and biking popularity. The distribution is centered at roughly 2000 meters. 2000 meters = 1.25 miles, which is a distance that would be a somewhat lengthy walk in a city but reasonable biking distance.

Taxi Speed Versus Bike Ride Count

In this scenario, we will again assume that people are more or less travelling to similar areas, regardless of mode of transportation. Will more people bike if there is more traffic in an area?

```
import altair as alt

bike_taxi_speed_graph = alt.Chart(bike_taxi_speed).mark_point(
    clip=True,
    opacity = .5
).encode(
  x=alt.X("Average_Speed:Q"),
  y=alt.Y("Number_Bike_Rides:Q"),
  tooltip=["Name", "Average_Speed", "Number_Bike_Rides:Q"]
).properties(
  title = "Taxi Trip Speed Versus Bike Ride Count"
)

bike_taxi_speed_graph
```



Much like the taxi trip distance, taxi trip speed also has a negative correlation with number of bike rides in that area. Note that the center of the distribution, 3 m/s = 6.7 mph. According to Citi Bike, the average biking speed is 6.7 and 7.8 mph! Note that there could be many confounding variables, but a rudimentary analysis suggest that people might consider biking when it is faster than taxis.

Motor Vehicle Accidents Versus Biking

If an area has a high number of accidents, are the roads dangerous and will people be less likely to bike? Or do accidents happen when there are many cars on the road, which might push people to bike? We explore this question further.

```
%%bigquery --project $project_id accidents_bikes

SELECT
  Number_Accidents,
  APPROX_QUANTILES(Number_Bike_Rides,1000)[OFFSET(500)] Average_Number_Bike_Rides
FROM (
  SELECT
    ROUND(latitude,3) latitude,
    ROUND(longitude,3) longitude,
    COUNT(*) Number_Accidents
  FROM
    `bigquery-public-data.new_york.nypd_mv_collisions`
  WHERE DATETIME(timestamp) BETWEEN DATETIME(TIMESTAMP "2016-01-01 00:00:00+00")
    AND DATETIME(TIMESTAMP "2017-01-01 00:00:00+00")
  GROUP BY
    latitude,
    longitude
    ) a,
  (
  SELECT
    ROUND(start_station_latitude, 3) latitude,
    ROUND(start_station_longitude,3) longitude,
    COUNT(*) AS Number_Bike_Rides,
    name
  FROM
    `bigquery-public-data.new_york_citibike.citibike_trips`,
    `bigquery-public-data.new_york_citibike.citibike_stations`
  WHERE
    NOT (start_station_id IS NULL
      OR end_station_id IS NULL)
    AND starttime BETWEEN DATETIME(TIMESTAMP "2016-01-01 00:00:00+00")
    AND DATETIME(TIMESTAMP "2017-01-01 00:00:00+00")
    AND start_station_id = station_id
```

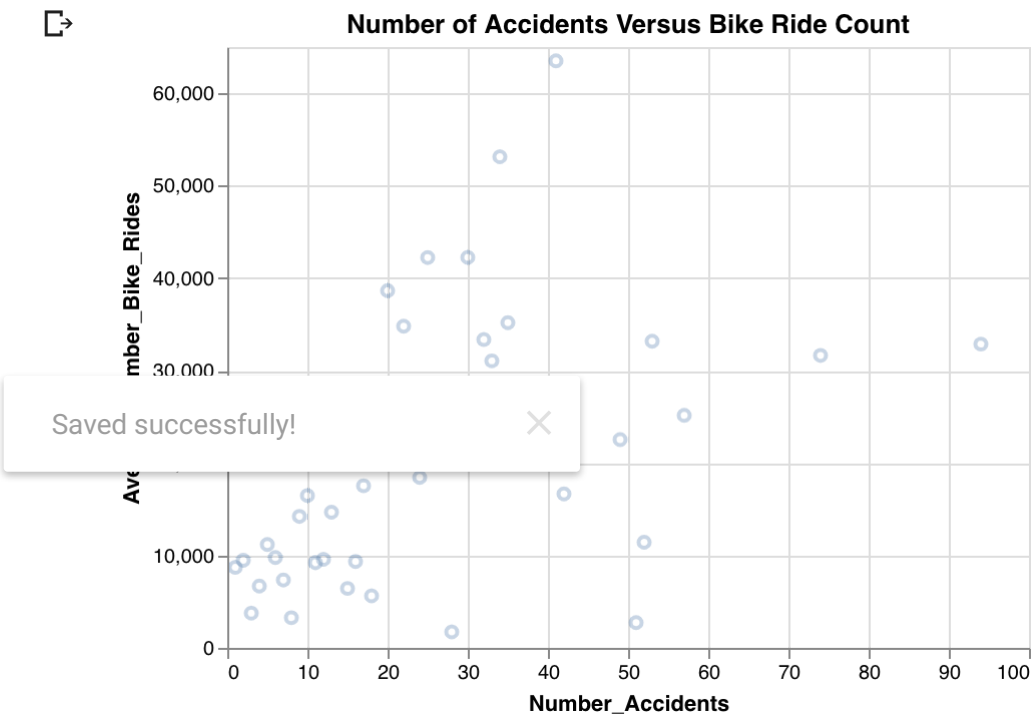
Saved successfully!

```
WHERE
  a.latitude = b.latitude
  AND a.longitude = b.longitude
GROUP BY
  Number_Accidents
```

12	5	11186
13	25	42232
14	1	8711
15	8	3266
16	9	14229
17	4	6687
18	23	20059
19	21	26218
20	29	24204
21	52	11428
22	16	9347

```
import altair as alt

accidents_bikes_graph = alt.Chart(accidents_bikes).mark_point(
    clip=True,
    opacity = .3
).encode(
  x=alt.X("Number_Accidents:Q"),
  y=alt.Y("Average_Number_Bike_Rides:Q"),
  tooltip=["Number_Accidents:Q", "Average_Number_Bike_Rides:Q"]
).properties(
  title = "Number of Accidents Versus Bike Ride Count"
)
```

```
accidents_bikes_graph
```



There seems to be a positive correlation between the number of accidents in an area and the station's bike ride activity. Are the bikers contributing to the accidents? Or do areas of high numbers of accidents just have more people moving about? Perhaps there is a clearer correlation if broken into smaller time frames. We will not do that here but we will incorporate it into our ML models below.

Trees

```
%%bigquery --project $project_id trees_bikes

SELECT
  name Name ,
  a.latitude Latitude,
  a.longitude Longitude,
  Number_Trees,
  Number_Bike_Rides
FROM (
  SELECT
    ROUND(latitude,3) latitude,
    ROUND(longitude,3) longitude,
    COUNT(*) Number_Trees
  FROM
    `bigquery-public-data.new_york.tree_census_2015`
  GROUP BY
    latitude,
    longitude
    ) a,
  (
  SELECT
    ROUND(start_station_latitude, 3) latitude,
    ROUND(start_station_longitude,3) longitude,
    COUNT(*) AS Number_Bike_Rides,
    name
  FROM
    `bigquery-public-data.new_york_citibike.citibike_trips`,
    `bigquery-public-data.new_york_citibike.citibike_stations`
  WHERE
    NOT (start_station_id IS NULL
      OR end_station_id IS NULL)
```

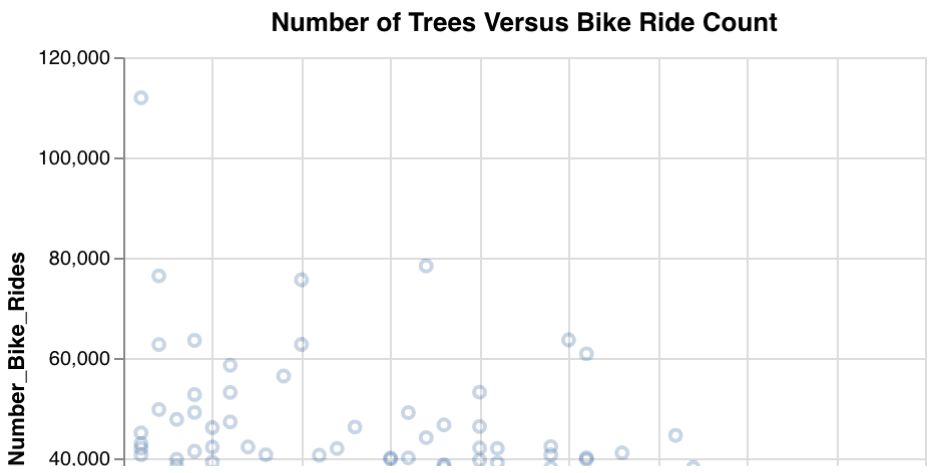
```
    AND starttime BETWEEN DATETIME(TIMESTAMP "2016-01-01 00:00:00+00")
    AND DATETIME(TIMESTAMP "2017-01-01 00:00:00+00")
    AND start_station_id = station_id
  GROUP BY
    latitude,
    longitude,
    name ) b
WHERE
  a.latitude = b.latitude
  AND a.longitude = b.longitude
```

	...	...	...	...	...	...
	481	E 103 St & Lexington Ave	40.790	-73.948	10	993
	482	Marcy Ave & Lafayette Ave	40.690	-73.948	12	1910
	483	Verona Pl & Fulton St	40.680	-73.948	12	1634
	484	Lorimer St & Broadway	40.704	-73.948	15	2549
	485	E 110 St & Madison Ave	40.796	-73.948	18	2136
	486	Nassau Ave & Newell St	40.725	-73.948	18	5626
	487	E 85 St & York Ave	40.775	-73.948	23	14163
	488	W 63 St & Broadway	40.772	-73.983	2	23975
	489	W 37 St & 5 Ave	40.750	-73.983	3	25640
	490	Broadway & W 53 St	40.763	-73.983	4	25046
	491	Rivington St & Ridge St	40.719	-73.983	6	13548
	492	Wyckoff St & 3 Ave	40.683	-73.983	9	644
	493	Bond St & Fulton St	40.690	-73.983	10	5032
	494	Bialystoker Pl & Delancey St	40.716	-73.983	10	16687
	495	W 45 St & 6 Ave	40.757	-73.983	11	25292
	496	Union St & 4 Ave	40.677	-73.983	13	1363
	497	W 70 St & Amsterdam Ave	40.777	-73.983	13	20402
	498	E 30 St & Park Ave S	40.744	-73.983	13	32538
			40.702	-73.983	16	6110
		W 67 St & Broadway	40.775	-73.983	17	21978
	501	Pacific St & Nevins St	40.685	-73.983	23	355
	502	W 100 St & Manhattan Ave	40.795	-73.965	9	480
	503	Willoughby Ave & Hall St	40.692	-73.965	11	8173
	504	Central Park W & W 96 St	40.791	-73.965	16	5905
	505	Lafayette Ave & St James Pl	40.689	-73.965	18	7133
	506	Broadway & Berry St	40.710	-73.965	19	10915
	507	3 Ave & E 62 St	40.763	-73.965	22	13771
	508	Tompkins Ave & Hopkins St	40.700	-73.947	7	2981
	509	Jackson Ave & 46 Rd	40.745	-73.947	7	3257
	510	3 Ave & E 100 St	40.788	-73.947	13	548

511 rows × 5 columns

Saved successfully!

```
import altair as alt

trees_bikes_graph = alt.Chart(trees_bikes).mark_point(
    clip=True,
    opacity=.3
).encode(
  x=alt.X("Number_Trees:Q"),
  y=alt.Y("Number_Bike_Rides:Q"),
  tooltip=["Number_Trees:Q", "Number_Bike_Rides:Q"]
).properties(
  title = "Number of Trees Versus Bike Ride Count"
)

trees_bikes_graph
```



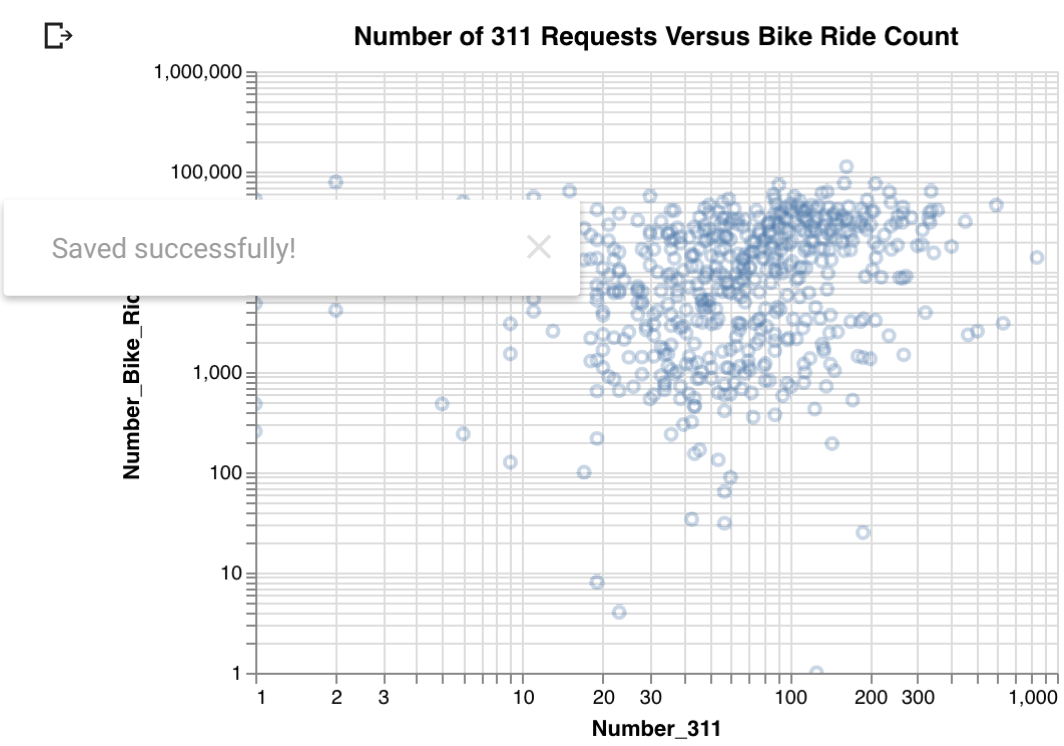
There does not seem to be a clear relationship between the number of trees and the number of bike rides. However, this does not mean a relationship is nonexistent. The relationship might be muddied by Central Park, or other parks with a high number of trees and located in an urban setting. Because of this, we will not include the number of trees in our ML model.

```
%%bigquery --project $project_id requests_bikes

SELECT
  name Name ,
  a.latitude Latitude,
  a.longitude Longitude,
  Number_311,
  Number_Bike_Rides
FROM (
  SELECT
    ROUND(latitude,3) latitude,
    ROUND(longitude,3) longitude,
    COUNT(*) Number_311
  FROM
    `bigquery-public-data.new_york.311_service_requests`
  WHERE
    DATETIME(created_date) BETWEEN DATETIME(TIMESTAMP "2016-01-01 00:00:00+00")
    AND DATETIME(TIMESTAMP "2017-01-01 00:00:00+00")
  GROUP BY
    latitude,
    longitude
    ) a,
  (
  SELECT
    ROUND(start_station_latitude, 3) latitude,
    ROUND(start_station_longitude,3) longitude,
    COUNT(*) AS Number_Bike_Rides,
    name
  FROM
                                    citibike.citibike_trips`,
                                    citibike.citibike_stations`
    NOT (start_station_id IS NULL
      OR end_station_id IS NULL)
    AND starttime BETWEEN DATETIME(TIMESTAMP "2016-01-01 00:00:00+00")
    AND DATETIME(TIMESTAMP "2017-01-01 00:00:00+00")
    AND start_station_id = station_id
  GROUP BY
    latitude,
    longitude,
    name ) b
WHERE
  a.latitude = b.latitude
  AND a.longitude = b.longitude
```

Saved successfully!

	Name	Latitude	Longitude	Number_311	Number_Bike_Rides
0	Atlantic Ave & Furman St	40.692	-74.000	12	11742
1	W 20 St & 8 Ave	40.743	-74.000	104	31269
2	Clinton St & 4 Place	40.678	-74.000	36	1055
3	St James Pl & Pearl St	40.711	-74.000	48	8711
4	Howard St & Centre St	40.719	-74.000	94	27219
5	Washington Pl & 6 Ave	40.732	-74.000	37	21172
6	W 13 St & 7 Ave	40.738	-74.000	157	19628
7	Greenwich Ave & Charles St	40.735	-74.000	97	33611
8	President St & Henry St	40.683	-74.000	34	739
9	Bayard St & Baxter St	40.716	-74.000	30	25223
10	E 6 St & Avenue B	40.725	-73.982	38	27446
11	5 St & 6 Ave	40.670	-73.982	54	622

```
import altair as alt

requests_bikes_graph = alt.Chart(requests_bikes).mark_point(
    clip=True,
    opacity = .3
).encode(
  x=alt.X("Number_311:Q", scale=alt.Scale(type="log")),
  y=alt.Y("Number_Bike_Rides:Q", scale=alt.Scale(type="log")),
  tooltip=["Name", "Number_311:Q", "Number_Bike_Rides:Q"]
).properties(
  title = "Number of 311 Requests Versus Bike Ride Count"
)

requests_bikes_graph
```



Export as SVG Export as PNG View Source Open in Vega Editor

Generally speaking, there seems to be more bike usage in areas of higher 311 calls. Possibly, the number of 311 call is a result of higher population, and more population affects the number of bike rides. However, the relationship is not clear-cut enough to include in our predictive model, especially not in a linear regression we will be using.

Machine Learning: Predict Number of Bikes Starting From Station

In this section, we will use a linear regression to predict the number of bike trips originating from a station on a given day. To begin, we use features such as

- Station (training dataimplicitly captures details about that station)
- Day of the week
- Rain
- Average Temperature
- Number of taxi trips originating from the same location

We split our data in train (60%), validation (20%), and test sets (20%). We use the FARM_FINGERPRINT hash function to ensure that no inputs overlap and our training data is deterministic, even when adding additional features.

Version 1: Train

```
%%bigquery --project $project_id
CREATE OR REPLACE MODEL `ml.bike_count_v1`
OPTIONS(model_type='linear_reg') AS
(
  SELECT Number_Outgoing AS label
```

```
    , CAST(a.station_id AS STRING) station_id
    , CAST(EXTRACT(DAYOFWEEK FROM a.Date) AS STRING) Day_Of_Week
    , IF(rain_drizzle = "1", True, False) rain_drizzle
    , (temp - 32)*5/9 temp
    , Number_Taxi_Trips
FROM
(
  SELECT station_id
       , ROUND(latitude,3) latitude
       , ROUND(longitude,3) longitude
       , COUNT(*) Number_Outgoing
       , EXTRACT(DATE FROM starttime) Date
       , capacity Capacity
  FROM `bigquery-public-data.new_york.citibike_trips`
     , `bigquery-public-data.new_york.citibike_stations`
  WHERE start_station_id = station_id
    AND start_station_id IS NOT NULL
    AND birth_year IS NOT NULL
    AND end_station_longitude != 0
    AND capacity != 0
  GROUP BY station_id
         , DATE
         , capacity
         , latitude
         , longitude
  ) a,
  (
    SELECT DATE(CAST(year AS INT64), CAST(mo AS INT64), CAST(da AS INT64)) AS Date
         , rain_drizzle
         , temp
    FROM (
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2016`

          UNION ALL
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2015`

          UNION ALL
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2014`

          UNION ALL

          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2013`
       )
       WHERE wban = '94789'
  ) b,
  (
    SELECT ROUND(pickup_latitude,3) latitude
         , ROUND(pickup_longitude,3) longitude
                                     up_datetime) Date
                                     rips

    (SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2016`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2015`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2014`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2013`
    )
    GROUP BY Date, latitude, longitude
  ) c
WHERE MOD(ABS(FARM_FINGERPRINT(FORMAT_DATE("%x",a.Date))), 10) < 6
    AND a.Date = b.Date
    AND a.Date = c.Date
    AND a.latitude = c.latitude
    AND a.longitude = c.longitude
)
```

Saved successfully!

Version 1: Training Info

```
%%bigquery --project $project_id

SELECT
  *
FROM
  ML.TRAINING_INFO(MODEL `ml.bike_count_v1`)
```

training_run iteration loss eval_loss duration_ms learning_rate

Version 1: Evaluate on Validation

```
%%bigquery --project $project_id
SELECT
  *
FROM
  ML.EVALUATE(MODEL `ml.bike_count_v1`,
(
  SELECT Number_Outgoing AS label
       , CAST(a.station_id AS STRING) station_id
       , CAST(EXTRACT(DAYOFWEEK FROM a.Date) AS STRING) Day_Of_Week
       , IF(rain_drizzle = "1", True, False) rain_drizzle
       , (temp - 32)*5/9 temp
       , Number_Taxi_Trips
  FROM
  (
    SELECT station_id
         , ROUND(latitude,3) latitude
         , ROUND(longitude,3) longitude
         , COUNT(*) Number_Outgoing
         , EXTRACT(DATE FROM starttime) Date
         , capacity Capacity
    FROM `bigquery-public-data.new_york.citibike_trips`
       , `bigquery-public-data.new_york.citibike_stations`
    WHERE start_station_id = station_id
      AND start_station_id IS NOT NULL
      AND birth_year IS NOT NULL
      AND end_station_longitude != 0
      AND capacity != 0
    GROUP BY station_id
           , DATE
           , capacity
           , latitude
           , longitude
  ) a,
  (
    SELECT DATE(CAST(year AS INT64), CAST(mo AS INT64), CAST(da AS INT64)) AS Date
         , rain_drizzle
         , temp
    FROM (
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2016`

          UNION ALL

          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2015`


                                                        ata.noaa_gsod.gsod2014`

          UNION ALL

          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2013`
     )
     WHERE wban = '94789'
  ) b,
  (
    SELECT ROUND(pickup_latitude,3) latitude
         , ROUND(pickup_longitude,3) longitude
         , EXTRACT(DATE FROM pickup_datetime) Date
         , COUNT(*) Number_Taxi_Trips
    FROM
    (SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2016`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2015`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2014`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2013`
    )
    GROUP BY Date, latitude, longitude
  ) c
  WHERE MOD(ABS(FARM_FINGERPRINT(FORMAT_DATE("%x",a.Date))), 10) >= 6
    AND MOD(ABS(FARM_FINGERPRINT(FORMAT_DATE("%x",a.Date))), 10) < 8
    AND a.Date = b.Date
    AND a.Date = c.Date
    AND a.latitude = c.latitude
    AND a.longitude = c.longitude
)
)
```

Saved successfully!

	mean_absolute_error	mean_squared_error	mean_squared_log_error	median_absolute_error	r2_score	explai
0	23.695257	1151.390185	0.418539	17.28949	0.677509	

We are off to a good start. Let us add in a few more features to see if we can improve the performance further.

Version 2: Train

We add several features to our model

- the number of motor-vehicle accidents at a station on a give date
- The borough the station is located in (if available)
- The total docking capacity at a station
- The average taxi trip speed originating from that station
- the depth of snow

Notice that we do not include taxi trip distance for trips originating from the station because speed is calculated from distance and including both would likely be redundant.

```
%%bigquery --project $project_id
CREATE OR REPLACE MODEL `ml.bike_count_v2`
OPTIONS(model_type='linear_reg') AS
(
  SELECT Number_Outgoing AS label
       , CAST(a.station_id AS STRING) station_id
       , CAST(EXTRACT(DAYOFWEEK FROM a.Date) AS STRING) Day_Of_Week
       , rain_drizzle
       , temp
       , Number_Taxi_Trips
       , IF(Number_Accidents is NULL, 0, Number_Accidents) Number_Accidents
       , IF(borough is NULL, "UNK", borough) borough
       , capacity
       , Average_Speed
       , sndp
  FROM
  (
    SELECT station_id
         , ROUND(latitude,3) latitude
         , ROUND(longitude,3) longitude
         , COUNT(*) Number_Outgoing
         , EXTRACT(DATE FROM starttime) Date
         , capacity Capacity
    FROM `bigquery-public-data.new_york.citibike_trips`
       , `bigquery-public-data.new_york.citibike_stations`
    WHERE start_station_id = station_id
      AND start_station_id IS NOT NULL
      AND birth_year IS NOT NULL
      AND end_station_longitude != 0
      AND capacity != 0
    GROUP BY station_id
          , DATE
          , capacity
          , latitude
          , longitude
  ) a,
  (
    SELECT DATE(CAST(year AS INT64), CAST(mo AS INT64), CAST(da AS INT64)) AS Date
                                  True, False) rain_drizzle

                                 ndp) sndp
    FROM (
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2016`

          UNION ALL

          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2015`

          UNION ALL
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2014`

          UNION ALL

          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2013`
     )
     WHERE wban = '94789'
  ) b,
  (
    SELECT ROUND(pickup_latitude,3) latitude
         , ROUND(pickup_longitude,3) longitude
         , EXTRACT(DATE FROM pickup_datetime) Date
         , COUNT(*) Number_Taxi_Trips
         , APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude, pickup_latitude), ST_GEOGPOINT(dropoff_lor
         , APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude, pickup_latitude), ST_GEOGPOINT(dropoff_lor
    FROM
    (SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2016`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2015`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2014`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2013`
    )
    WHERE NOT(pickup_datetime IS NULL
          OR dropoff_datetime IS NULL
          OR pickup_datetime >= dropoff_datetime
          OR trip_distance = 0
          OR pickup_longitude = 0
          OR ABS(pickup_longitude) > 180
          OR pickup_latitude = 0
          OR ABS(pickup_latitude) > 90
          OR dropoff_longitude = 0
          OR ABS(dropoff_longitude) > 180
          OR dropoff_latitude = 0
```

Saved successfully!

```
            OR ABS(dropoff_latitude) > 90
            OR total_amount <= 0
            OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
            )
      GROUP BY Date, latitude, longitude
    ) c
    LEFT OUTER JOIN
    (
      SELECT ROUND(latitude,3) latitude
           , ROUND(longitude,3) longitude
           , COUNT(*) Number_Accidents
           , borough
           , EXTRACT(DATE FROM timestamp) Date
      FROM `bigquery-public-data.new_york.nypd_mv_collisions`
      GROUP BY Date, latitude, longitude, borough
    ) d ON a.Date = d.Date
        AND a.latitude = d.latitude
        AND a.longitude = d.longitude
    WHERE MOD(ABS(FARM_FINGERPRINT(FORMAT_DATE("%x",a.Date))), 10) < 6
      AND a.Date = b.Date
      AND a.Date = c.Date
      AND a.latitude = c.latitude
      AND a.longitude = c.longitude
)
```

Version 2: Training Info

```
%%bigquery --project $project_id

SELECT
  *
FROM
  ML.TRAINING_INFO(MODEL `ml.bike_count_v2`)
```

	training_run	iteration	loss	eval_loss	duration_ms	learning_rate
0	0	4	1111.004074	1166.502644	7943	0.8
1	0	3	1114.197627	1169.805760	8037	0.8
2	0	2	1135.384484	1188.118356	7247	0.8
3	0	1	1474.491137	1530.288698	7701	0.4
4	0	0	2376.923217	2420.943982	4000	0.2

Saved successfully!

data.

Version 2: Evaluation on Validation Set

```
%%bigquery --project $project_id
SELECT
  *
FROM
  ML.EVALUATE(MODEL `ml.bike_count_v2`,
(
  SELECT Number_Outgoing AS label
       , CAST(a.station_id AS STRING) station_id
       , CAST(EXTRACT(DAYOFWEEK FROM a.Date) AS STRING) Day_Of_Week
       , rain_drizzle
       , temp
       , Number_Taxi_Trips
       , IF(Number_Accidents is NULL, 0, Number_Accidents) Number_Accidents
       , IF(borough is NULL, "UNK", borough) borough
       , capacity
       , Average_Speed
       , sndp
FROM
(
  SELECT station_id
       , ROUND(latitude,3) latitude
       , ROUND(longitude,3) longitude
       , COUNT(*) Number_Outgoing
       , EXTRACT(DATE FROM starttime) Date
       , capacity Capacity
  FROM `bigquery-public-data.new_york.citibike_trips`
     , `bigquery-public-data.new_york.citibike_stations`
  WHERE start_station_id = station_id
    AND start_station_id IS NOT NULL
    AND birth_year IS NOT NULL
    AND end_station_longitude != 0
    AND capacity != 0
  GROUP BY station_id
         , DATE
         , capacity
         , latitude
         , longitude
  ) a,
  (
    SELECT DATE(CAST(year AS INT64), CAST(mo AS INT64), CAST(da AS INT64)) AS Date
         , IF(rain_drizzle = "1", True, False) rain_drizzle
         , (temp - 32)*5/9 temp
         , IF(sndp = 999.9, 0 , sndp) sndp
    FROM (
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2016`

          UNION ALL
```

```
        SELECT *
        FROM `bigquery-public-data.noaa_gsod.gsod2015`

        UNION ALL
        SELECT *
        FROM `bigquery-public-data.noaa_gsod.gsod2014`

        UNION ALL

        SELECT *
        FROM `bigquery-public-data.noaa_gsod.gsod2013`
   )
   WHERE wban = '94789'
) b,
(
  SELECT ROUND(pickup_latitude,3) latitude
       , ROUND(pickup_longitude,3) longitude
       , EXTRACT(DATE FROM pickup_datetime) Date
       , COUNT(*) Number_Taxi_Trips
       , APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude, pickup_latitude), ST_GEOGPOINT(dropoff_lor
       , APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude, pickup_latitude), ST_GEOGPOINT(dropoff_lor
  FROM
  (SELECT *
  FROM
    `bigquery-public-data.new_york.tlc_yellow_trips_2016`
  UNION ALL
  SELECT *
  FROM
    `bigquery-public-data.new_york.tlc_yellow_trips_2015`
  UNION ALL
  SELECT *
  FROM
    `bigquery-public-data.new_york.tlc_yellow_trips_2014`
  UNION ALL
  SELECT *
  FROM
    `bigquery-public-data.new_york.tlc_yellow_trips_2013`
  )
  WHERE NOT(pickup_datetime IS NULL
        OR dropoff_datetime IS NULL
        OR pickup_datetime >= dropoff_datetime
        OR trip_distance = 0
        OR pickup_longitude = 0
        OR ABS(pickup_longitude) > 180
        OR pickup_latitude = 0
        OR ABS(pickup_latitude) > 90
        OR dropoff_longitude = 0
        OR ABS(dropoff_longitude) > 180
        OR dropoff_latitude = 0
        OR ABS(dropoff_latitude) > 90
        OR total_amount <= 0
        OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
        )
                                                    itude
```

Saved successfully!

```
    (
      SELECT ROUND(latitude,3) latitude
           , ROUND(longitude,3) longitude
           , COUNT(*) Number_Accidents
           , borough
           , EXTRACT(DATE FROM timestamp) Date
      FROM `bigquery-public-data.new_york.nypd_mv_collisions`
      GROUP BY Date, latitude, longitude, borough
    ) d ON a.Date = d.Date
      AND a.latitude = d.latitude
      AND a.longitude = d.longitude
   WHERE MOD(ABS(FARM_FINGERPRINT(FORMAT_DATE("%x",a.Date))), 10) >= 6
    AND MOD(ABS(FARM_FINGERPRINT(FORMAT_DATE("%x",a.Date))), 10) < 8
    AND a.Date = b.Date
    AND a.Date = c.Date
    AND a.latitude = c.latitude
    AND a.longitude = c.longitude
  )
)
```

	mean_absolute_error	mean_squared_error	mean_squared_log_error	median_absolute_error	r2_score	explai
0	23.565149	1135.105864	0.399521	17.213496	0.682194	

We have a slight improvement but not by much. We may have simply just hit a wall, since linear regressions cannot easily capture the complexities between features. We will go ahead and run the same model on the held - out test data.

Version 2: Evaluate on Test Data

```
%%bigquery --project $project_id
SELECT
  *
FROM
  ML.EVALUATE(MODEL `ml.bike_count_v2`,
(
  SELECT Number_Outgoing AS label
       , CAST(a.station_id AS STRING) station_id
       , CAST(EXTRACT(DAYOFWEEK FROM a.Date) AS STRING) Day_Of_Week
       , rain_drizzle
       , temp
       , Number_Taxi_Trips
       , IF(Number_Accidents is NULL, 0, Number_Accidents) Number_Accidents
       , IF(borough is NULL, "UNK", borough) borough
       , capacity
       , Average_Speed
       , sndp
```

```
FROM
(
  SELECT station_id
       , ROUND(latitude,3) latitude
       , ROUND(longitude,3) longitude
       , COUNT(*) Number_Outgoing
       , EXTRACT(DATE FROM starttime) Date
       , capacity Capacity
  FROM `bigquery-public-data.new_york.citibike_trips`
     , `bigquery-public-data.new_york.citibike_stations`
  WHERE start_station_id = station_id
    AND start_station_id IS NOT NULL
    AND birth_year IS NOT NULL
    AND end_station_longitude != 0
    AND capacity != 0
  GROUP BY station_id
         , DATE
         , capacity
         , latitude
         , longitude
  ) a,
  (
    SELECT DATE(CAST(year AS INT64), CAST(mo AS INT64), CAST(da AS INT64)) AS Date
         , IF(rain_drizzle = "1", True, False) rain_drizzle
         , (temp - 32)*5/9 temp
         , IF(sndp = 999.9, 0 , sndp) sndp
    FROM (
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2016`

          UNION ALL

          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2015`

          UNION ALL
          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2014`

          UNION ALL

          SELECT *
          FROM `bigquery-public-data.noaa_gsod.gsod2013`
     )
     WHERE wban = '94789'
  ) b,
  (
    SELECT ROUND(pickup_latitude,3) latitude
         , ROUND(pickup_longitude,3) longitude
         , EXTRACT(DATE FROM pickup_datetime) Date
         , COUNT(*) Number_Taxi_Trips
         , APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude, pickup_latitude), ST_GEOGPOINT(dropoff_lor
         , APPROX_QUANTILES(ST_DISTANCE(ST_GEOGPOINT(pickup_longitude, pickup_latitude), ST_GEOGPOINT(dropoff_lor
```

Saved successfully!

```
      `bigquery-public-data.new_york.tlc_yellow_trips_2016`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2015`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2014`
    UNION ALL
    SELECT *
    FROM
      `bigquery-public-data.new_york.tlc_yellow_trips_2013`
    )
    WHERE NOT(pickup_datetime IS NULL
          OR dropoff_datetime IS NULL
          OR pickup_datetime >= dropoff_datetime
          OR trip_distance = 0
          OR pickup_longitude = 0
          OR ABS(pickup_longitude) > 180
          OR pickup_latitude = 0
          OR ABS(pickup_latitude) > 90
          OR dropoff_longitude = 0
          OR ABS(dropoff_longitude) > 180
          OR dropoff_latitude = 0
          OR ABS(dropoff_latitude) > 90
          OR total_amount <= 0
          OR (pickup_longitude = dropoff_longitude AND pickup_latitude = dropoff_latitude)
          )
      GROUP BY Date, latitude, longitude
    ) c
    LEFT OUTER JOIN
    (
      SELECT ROUND(latitude,3) latitude
           , ROUND(longitude,3) longitude
           , COUNT(*) Number_Accidents
           , borough
           , EXTRACT(DATE FROM timestamp) Date
      FROM `bigquery-public-data.new_york.nypd_mv_collisions`
      GROUP BY Date, latitude, longitude, borough
    ) d ON a.Date = d.Date
        AND a.latitude = d.latitude
        AND a.longitude = d.longitude
    WHERE MOD(ABS(FARM_FINGERPRINT(FORMAT_DATE("%x",a.Date))), 10) >=8
      AND a.Date = b.Date
      AND a.Date = c.Date
      AND a.latitude = c.latitude
      AND a.longitude = c.longitude
  )
)
```

	mean_absolute_error	mean_squared_error	mean_squared_log_error	median_absolute_error	r2_score	explai

Our evaluation on the test data yielded even better results than evaluation on the training data, meaning our data split worked as expected. We did not overfit.

```
%%bigquery --project $project_id
SELECT AVG(avg) Average_Outgoing
FROM(
SELECT AVG(Number_Bike_Rides) avg
FROM
  (
  SELECT COUNT(*) Number_Bike_Rides, EXTRACT(DATE FROM starttime) Date
  FROM `bigquery-public-data.new_york.citibike_trips`
  GROUP BY start_station_id, Date
  )
GROUP BY Date
)
```

	Average_Outgoing
0	74.3507

Considering that there is an average of 74 bikes that leave a station on an average day, our model is not superb but still reasonable.

Conclusion

We answered several optimization questions regarding Citi Bike usage in New York. Namely, we discovered the best times of each day to redistribute bikes. We analyzed which bikes had the greatest disparities between incoming and outgoing bikes. We analyzed which bike stations were being starved of resources (either in the form of bikes or docks).

Furthermore, we explored interesting correlations between taxi data and bike data, including the raw number of taxi trips in an area, the average distance of those trips, and the average speed of those trips.

Other questions had less exciting answers. There was a vague positive correlationship between the number of accidents in an area and bike activity. From what we could tell, trees do not have a clear correlation to bike activity. Neither really do 311 servicee requests.

In the ML portion of this project, we constructed a model to predict the number of outgoing bike rides a station would have in a given day, based on factors including but not limited to taxi usage, weather, accidents, location, resources, etc.

One significant limitation of my approach was that I used "crow flies" distance in all my calculations, where Manhattan distance may be a more ...ce and speed.

Saved successfully!

There are many more exciting parts of this project to be explored. For example, I would examine proximity to water, population density, and neighborhood crime in future considerations.