UChicago | MSCA 31012
Data Engineering Platforms for Analytics

June 6, 2019

Annie Qurat ul ain
WooJong Choi
Tam Nguyen
Markus Wehr

Outline

- Executive Summary
- Business Use Case
- Relational database and tools
- Data Analysis and Visualization
- Tableau Visualization
- Summary

Executive Summary

INTRODUCTION

Goal
Invest US$50 million to:
- Expand stations to all **50 city wards**
- Add **175** stations and **10,500** bikes

2021

2019 - 2020
- 2019: More than **20k rides** per day in peak seasons.
- March 2019, **Lyft** took over Divvy
- Early 2020: Plan to pass **20 millionth rides** mark.

2017 -2018

Second expansion
(107 new stations)
Provided its 15 millionth rides in 2018

2015 - 2016

First expansion
(175 new stations)

Officially launched
in June 2013
(75 stations and 750 bikes)

2013

DIVVY

Bikeshare system

6,000 bikes

608 stations

Chicagoans' regular mode of transportation

RESEARCH OBJECTIVES

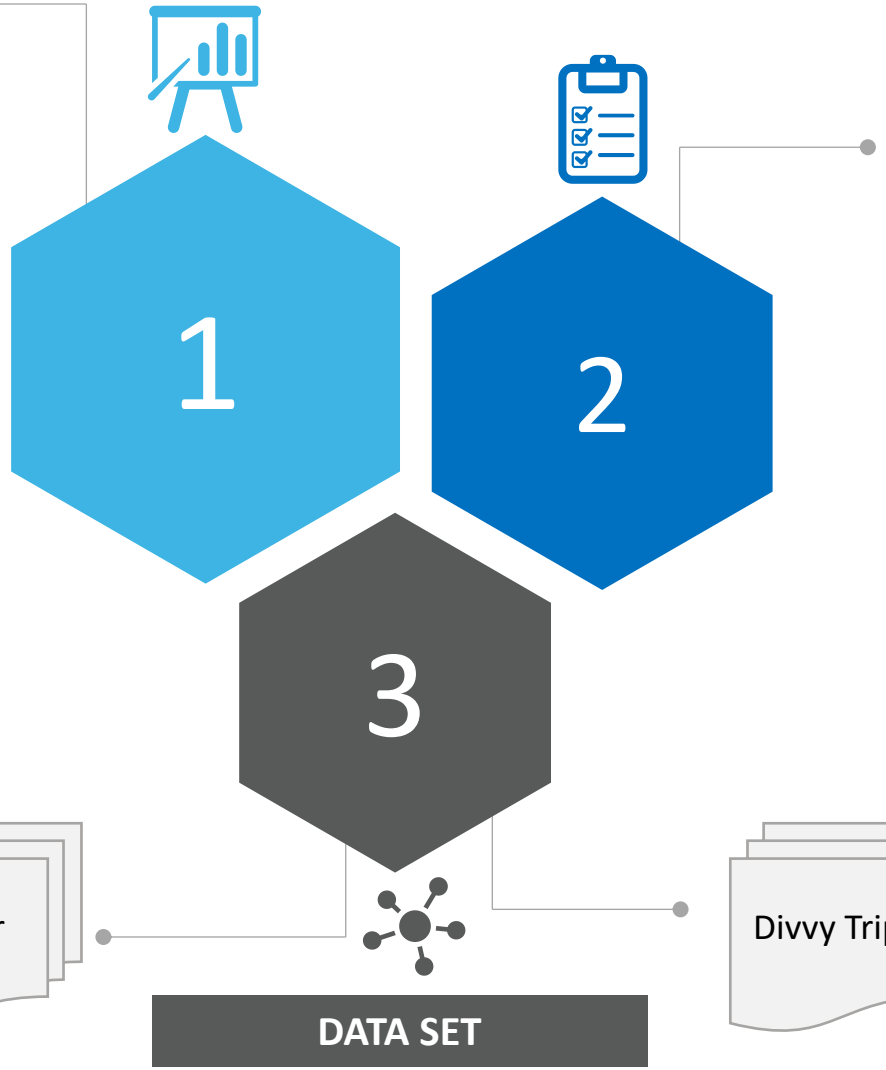
- To assist with the expansion plan, our team developed a relational database that will enable quick response and analysis on the current state Divvy operations in regard to ridership, station locations and various other factors affecting them. And:
  - Provide methodologies and various tools used in the process
  - Provide data analysis and visualization
  - Put forward a future state blueprint for the new stations and bikes allocation process

Business use case

PROPOSED FINDING

Our final deliverables will enable Divvy leadership to:

- Understand current ridership and station locations
- Understand various factors that impact ridership. i.e
  - Demographic
  - Traffic volume
  - Bike racks / lanes
  - Weather
- Develop dashboards and KPIs to gauge overall business / operation performance
- Plan for future station & bikes allocation



METHODOLOGY

- Develop a scoring model to determine optimal number of stations and bikes by zip codes based on various factors
- Visualize findings from analysis - trends, outliers, patterns and predictions

Data Source

Dataset	Source		File Format	Size
Trip	Divvy	https://www.divvybikes.com/system-data	CSV	> 1mil rows
Station	City of Chicago	https://data.cityofchicago.org/Transportation/Divvy-Bicycle-Stations/bbyy-e7gq	CSV	> 600 rows
Station_zip	Divvy	https://feeds.divvybikes.com/stations/stations.json	JSON	> 600 rows
Weather	National Weather Service Forecast Office	https://w2.weather.gov/climate/xmacis.php?wfo=lot	CSV	> 12k rows
Bike racks	City of Chicago	https://data.cityofchicago.org/Transportation/Bike-Racks/cbyb-69xx	CSV	> 5k rows
Population	City of Chicago	https://catalog.data.gov/dataset?res_format=CSV&organization=city-of-chicago	CSV	< 100 rows
Bike route	City of Chicago	https://data.cityofchicago.org/Transportation/Bike-Routes/3w5d-sru8	CSV	< 1k rows
Zip code	Chicago Data Type	http://robparal.blogspot.com/2013/07/chicago-community-area-and-zip-code.html	CSV	< 100 rows

Relational Database and Tools

Fact and dimensional table

Table Name	Table Type	Cardinality	Additional Details
fact_trip	Fact Table	M:1 Relationship with Station and Weather Table	Contains information about each trip including the start/end station, total time, age, gender of the customer
dim_station	Dimensional Table	1:M relationship with Fact Table	Contains information like station address, total number of docks available, date the station became available.
dim_weather	Dimensional Table	1:M relationship with Fact Table	Contains temperature, rain/snow, wind information in hourly format. Also, contains the sunset and sunrise time.
dim_population	Dimensional Table	1:M relationship with Location Table	Contains information about the population (age, gender) demographics zip wise.
dim_location	Dimensional Table	M:1 relationship with Population Table	Contains the location of all the stations, traffic routes, bike routes. Zip code is a must have for each address.
dim_traffic	Dimensional Table	1:M relationship with Location Table	Contains the traffic flow information daily including the direction (Northbound, Southbound, Westward, Eastward) on streets.
dim_bike_racks	Dimensional Table	1:M relationship with Location Table	Contains information about the non-divvy bike racks scattered across Chicago city
dim_bike_lane	Dimensional Table	1:1 relationship with Location Table	Contains information about the bike routes in the city, including their length and the streets they run on.

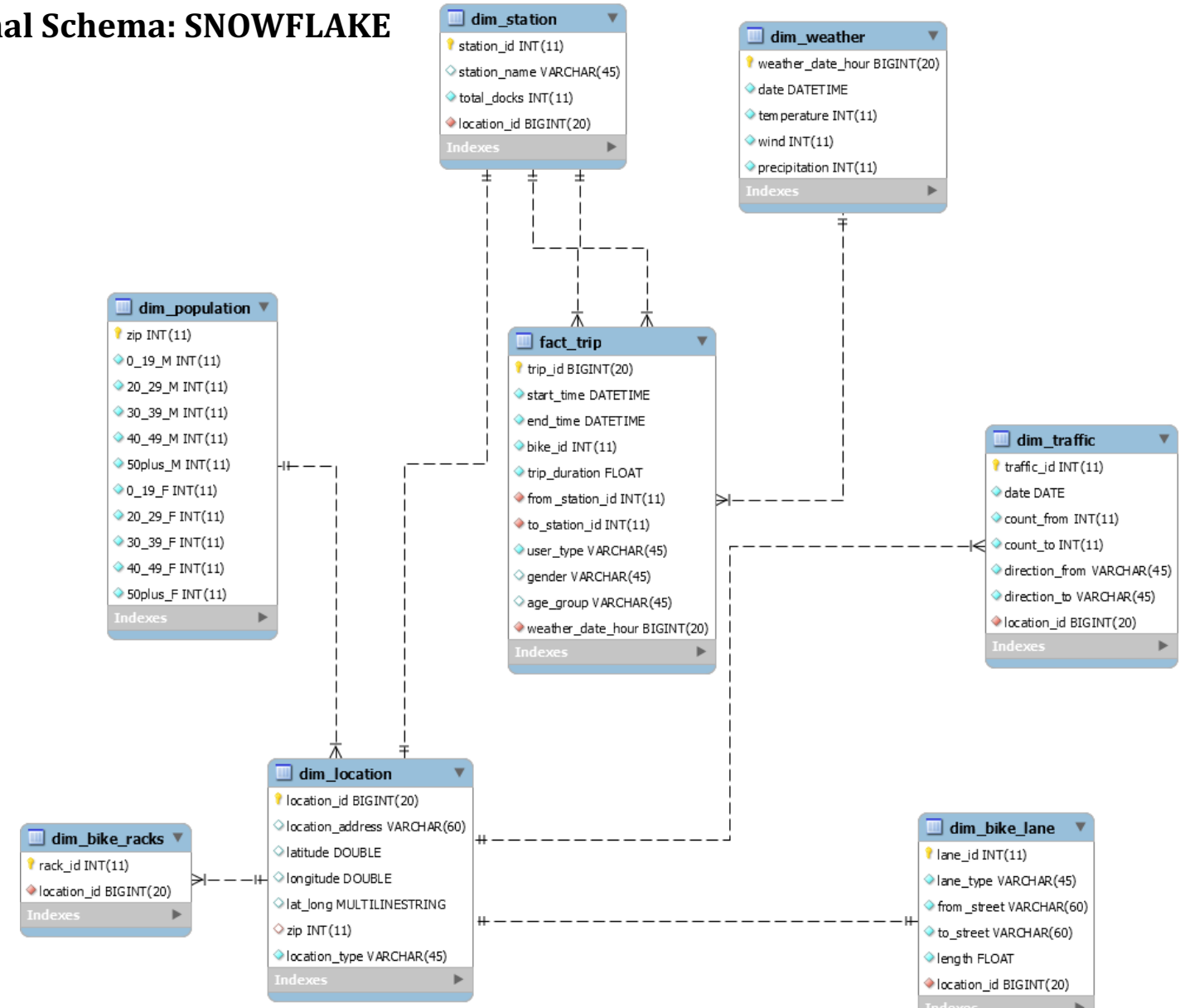
Fact table joined with Dimension tables provides interesting insights into how variables interact. Fact Table can be sliced by time and diced by stations, gender and age variables.

Database Design: Enhanced Entity Relational Diagram

DDL

DML

Dimensional Schema: SNOWFLAKE



Tools

1

Data

Data from various sources

2

ETL Process

3

Data Warehouse

Google Cloud Platform

4

Data Analysis

5

Data Visualization

6

Findings and Recommendations

Data extraction, Cleaning, Normalization

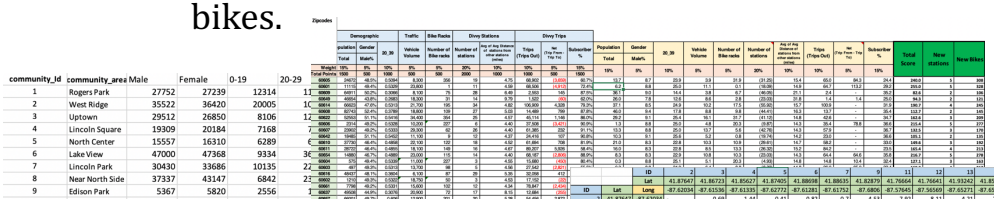
- Create and and load database
- Produce queries to support project's analysis purpose

```
# Number of trips by hour by weekay and weekend.
SELECT
CASE WHEN dayname(start_time) IN ("Saturday", "Sunday") THEN "Weekend" ELSE "Weekday" END AS DateType,
HOUR(start_time) AS TimeOfDay,
COUNT(trip_id) as NoOfTrips
FROM fact_trip
GROUP BY TimeOfDay, DateType
ORDER BY TimeOfDay DESC;
```

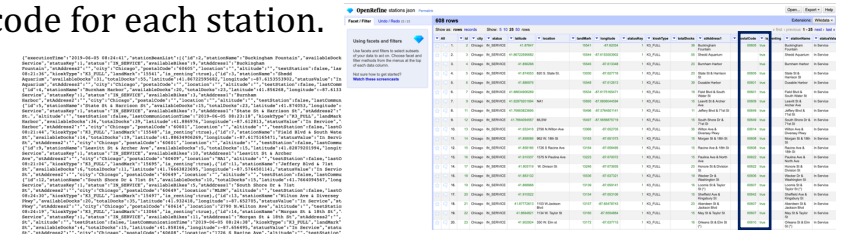
```
#8. Number of TripIn per zip
SELECT
    dl.zip,
    COUNT(ft.to_station_id) AS TripIn
FROM
    fact_trip ft
    INNER JOIN dim_station ds ON ds.station_id = ft.to_stati
    LEFT JOIN dim_location dl   ON dl.location_id = ds.locat
GROUP BY zip
ORDER BY TripIn DESC;
```

```
SELECT
TripFrom.station_id,
TripFrom.stationName AS stationName,
TripFrom.TimeofDay AS timeFrom,
TripTo.TimeofDay AS timeTo,
TripFrom.tripFrom,
TripTo.tripTo,
(TripFrom.tripFrom - TripTo.tripTo) AS NetTrip,
(TripFrom.totalDocks - TripFrom.tripFrom + TripTo.tripTo) AS docksNeededgaps
FROM
((SELECT
ds.station_id,
ds.station_name AS stationName,
ds.total_docks AS totalDocks,
HOUR(ft.start_time) AS TimeOfDay,
COUNT(ft.from_station_id) as tripFrom
FROM fact_trip ft
INNER JOIN dim_station ds
ON ds.station_id = ft.from_station_id
GROUP BY ds.station_id, TimeOfDay
ORDER BY ds.station_id,TimeOfDay ASC) AS TripFrom
INNER JOIN
((SELECT
ds.station_id,
ds.station_name AS stationName,
ds.total_docks AS totalDocks,
HOUR(ft.end_time) AS TimeOfDay,
COUNT(ft.to_station_id) as tripTo
FROM fact_trip ft
INNER JOIN dim_station ds
ON ds.station_id = ft.to_station_id
GROUP BY ds.station_id, TimeOfDay
ORDER BY ds.station_id,TimeOfDay ASC) AS TripTo
ON TripFrom.station_id = TripTo.station_id
WHERE TripFrom.TimeOfDay= TripTo.TimeOfDay;
```

- Clean all dimensional tables to import to mySQL
- Analyze descriptive data: customer profiling, zip, stations
- Build the scoring system for research objectives' purpose: add more stations and bikes.



- Import, clean, and extract real-time station data from Divvy to get the zip code for each station.



- Get the zipcode using longitude and latitude for dim_location table
- Estimated the distance between trips
- Stack the distance data to produce an adaptable format for tableau visualization purpose
- Conduct some correlation between trips and other factors: weekday, bike racks, weather…

```
install.packages("revgeo")
library(revgeo)

dataPath <- "/Users/tammy/Tammy/UChicago/Data_Engineering_Platform/FinalProject/Processing/"

#Rack1
rack1 <- read.csv(paste(dataPath,"rack1.csv",sep="/"))
rack1_zip <- revgeo(longitude=rack1$longitude, latitude=rack1$latitude,
                    output='frame', item='zip')
write.table(rack1_zip, file = paste(dataPath,'rack1_zip.csv',sep = '/'), row.names = F)

#Rack2
rack2 <- read.csv(paste(dataPath,"rack2.csv",sep="/"))
rack2_zip <- revgeo(longitude=rack2$longitude, latitude=rack2$latitude,
                    output='frame', item='zip')
write.table(rack2_zip, file = paste(dataPath,'rack2_zip.csv',sep = '/'), row.names = F)

#Rack3
rack3 <- read.csv(paste(dataPath,"rack3.csv",sep="/"))
rack3_zip <- revgeo(longitude=rack3$longitude, latitude=rack3$latitude,
                    output='frame', item='zip')
write.table(rack3_zip, file = paste(dataPath,'rack3_zip.csv',sep = '/'), row.names = F)
```

```
```{r distance}
#library(geosphere)
#distance<-c()
#for(i in 1:5){
#  distance<-c(distance,distVincentyEllipsoid(c(geo$lat1[i],geo$long1[i]),c(geo$lat2[i],geo$long2[i])))
#}
```
```

```
```{r distance}
distance<-read.csv("C:/Users/wehrm/OneDrive/Documents/UChicago/Quarter 1/Data Engineering Platforms/Final project/Data/distance.csv",header=TRUE,sep=',')
```
```

```
```{r distance}
dim_distance<-data.frame(geo,distance)
head(dim_distance)
```

Average distance per station

```{r distance}
avg_distance<-aggregate(dim_distance[,11],list(dim_distance$station_id), mean)
head(avg_distance)
```
```

```
```{r distance}
mySQL<-"SELECT
  fr5.station_id,
  fr5.zip,
  fr5.trip_id,
  fr5.latitude AS lat1,
  fr5.longitude AS long1,
  fr5.station_id,
  fr5.zip,
  fr5.trip_id,
  fr5.latitude AS lat2,
  fr5.longitude AS long2
FROM
(SELECT
  ds.station_id,
  dl.zip,
  ft.trip_id,
  dl.latitude,
  dl.longitude
FROM
  dim_location dl
  INNER JOIN
  dim_station ds ON dl.location_id=ds.location_id
  INNER JOIN
  fact_trip ft ON ds.station_id=ft.from_station_id) AS fr5
  INNER JOIN
(SELECT
  ds.station_id,
  dl.zip,
  ft.trip_id,
  dl.latitude,
  dl.longitude
FROM
  dim_location dl
  INNER JOIN
  dim_station ds ON dl.location_id=ds.location_id
  INNER JOIN
  fact_trip ft ON ds.station_id=ft.to_station_id) AS fr5 ON fr5.trip_id=fr5.trip_id
WHERE
  fr5.station_id != fr5.station_id;"
geo<-dbGetQuery(connection, myQuery)
head(geo)
```
```

- Construct fact_trip table to import to my SQL:
  - Calculate the age group of Divvy users
  - Add in new column as a foreign key using in mySQL.

```
In [34]: def age_groups(age):
    if age < 20:
        return '0_19'
    elif 20 <= age < 30:
        return '20-29'
    elif 30 <= age < 40:
        return '30-39'
    elif 40 <= age < 50:
        return '40-49'
    elif 50 <= age:
        return '50+'

fact_trip1['age_group'] = fact_trip1['age'].apply(age_groups)

fact_trip1['age_group'].value_counts(sort=False)

Out[34]: 20-29    267525
         30-39    346995
         0_19       1342
         50+      129777
         40-49    126001
         Name: age_group, dtype: int64

In [35]: fact_trip1.head()

Out[35]:
   trip_id  start_time  end_time  bike_id  trip_duration  \
0  18005527  2018-04-01 00:04:44  2018-04-01 00:13:03  3819  499.0
1  18005528  2018-04-01 00:06:42  2018-04-01 00:27:07  5098  1,225.0
2  18005529  2018-04-01 00:07:19  2018-04-01 00:23:19  5165  960.0
3  18005530  2018-04-01 00:07:23  2018-04-01 00:14:47  3851  424.0
4  18005531  2018-04-01 00:10:23  2018-04-01 00:22:12  5965  709.0

   from_station_id  to_station_id  user_type  gender  birthyear  age  \
0               20            22  Customer     NaN        NaN  NaN
1              157           171  Subscriber  Male     1994.0  25.0
2              157           190  Subscriber  Male     1965.0  54.0
3              106           106  Customer    NaN        NaN  NaN
4              228           219  Subscriber  Male     1983.0  36.0
```

```
In [37]: fact_trip1['date','time']=fact_trip['start_time'].str.split(expand=True)
         fact_trip1['hour']=fact_trip['start_time'].dt.hour
         fact_trip1['day']=fact_trip['start_time'].dt.weekday_name
         fact_trip1['weather_date_hour']=fact_trip['month']+fact_trip['day']+fact_trip['year']+fact_trip['hour']
         fact_trip1.head()

         /Users/tammy/anaconda3/lib/python3.7/site-packages/pandas/core/frame.py:3140: SettingWithCopyWarning:
         A value is trying to be set on a copy of a slice from a DataFrame.
         Try using .loc[row_indexer,col_indexer] = value instead

         See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
           self[k1] = value[k2]

Out[37]:
   trip_id  start_time  end_time  bike_id  trip_duration  from_station_id  to_station_id  user_type  gender  age_group  date  time  year  month  day  hour
0  18005527  2018-04-01 00:04:44  2018-04-01 00:13:03  3819  499.0  22  171  Subscriber  Male  20-29  2018-04-01  00:04:44  2018  04  01  00
1  18005528  2018-04-01 00:06:42  2018-04-01 00:27:07  5098  1,225.0  157  190  Subscriber  Male  50+  2018-04-01  00:06:42  2018  04  01  00
2  18005529  2018-04-01 00:07:19  2018-04-01 00:23:19  5165  960.0  106  106  Customer  NaN  NaN  2018-04-01  00:07:19  2018  04  01  00
3  18005530  2018-04-01 00:07:23  2018-04-01 00:14:47  3851  424.0  241  171  Subscriber  Male  20-29  2018-04-01  00:07:23  2018  04  01  00
4  18005531  2018-04-01 00:10:23  2018-04-01 00:22:12  5965  709.0  228  219  Subscriber  Male  30-39  2018-04-01  00:10:23  2018  04  01  00
```

```
In [15]: fact_trip = fact_trip[['trip_id', 'start_time', 'end_time', 'bike_id', 'trip_duration',
                                'from_station_id', 'to_station_id',
                                'user_type', 'gender', 'age_group', 'weather_date_hour']]

         fact_trip.head()

Out[15]:
   trip_id  start_time  end_time  bike_id  trip_duration  from_station_id  to_station_id  user_type  gender  age_group  weather_date_hour
0  18005527  2018-04-01 00:04:44  2018-04-01 00:13:03  3819  499.0  22  171  Subscriber  Male  20-29  40101400
```

Sample Queries

Net influx per station and hour

```sql
SELECT
    TripFrom.station_id,
    TripFrom.stationName AS stationName,
    TripFrom.TimeofDay AS tripTime,
    TripFrom.tripFrom,
    TripTo.tripTo,
    (TripFrom.tripFrom - TripTo.tripTo) AS NetTrip
FROM
    (SELECT
        ds.station_id,
        ds.station_name AS stationName,
        ds.total_docks AS totalDocks,
        HOUR(ft.start_time) AS TimeOfDay,
        COUNT(ft.from_station_id) as tripFrom
    FROM
        fact_trip ft
            INNER JOIN
        dim_station ds ON ds.station_id = ft.from_station_id
    GROUP BY
        ds.station_id, TimeOfDay
    ORDER BY
        ds.station_id,TimeOfDay ASC) AS TripFrom
        INNER JOIN
     (SELECT
         ds.station_id,
         ds.station_name AS stationName,
         ds.total_docks AS totalDocks,
         HOUR(ft.end_time) AS TimeOfDay,
         COUNT(ft.to_station_id) as tripTo
     FROM
         fact_trip ft
             INNER JOIN
         dim_station ds ON ds.station_id = ft.to_station_id
     GROUP BY
         ds.station_id, TimeOfDay
     ORDER BY ds.station_id,TimeOfDay ASC) AS TripTo ON TripFrom.station_id = TripTo.station_id
WHERE TripFrom.TimeofDay = TripTo.TimeofDay;
```

Average distance travelled per station and zip code

```sql
SELECT

    FrS.station_id,
    FrS.trip_id,
    FrS.latitude AS lat1,
    FrS.longitude AS long1,
    TrS.station_id,
    TrS.trip_id,
    TrS.latitude AS lat2,
    TrS.longitude AS long2
FROM
(SELECT
    ds.station_id,
    ft.trip_id,
    dl.latitude,
    dl.longitude
FROM
    dim_location dl
        INNER JOIN
    dim_station ds ON dl.location_id=ds.location_id
        INNER JOIN
    fact_trip ft ON ds.station_id=ft.from_station_id) AS FrS
    INNER JOIN
(SELECT
    ds.station_id,
    ft.trip_id,
    dl.latitude,
    dl.longitude
FROM
    dim_location dl
        INNER JOIN
    dim_station ds ON dl.location_id=ds.location_id
        INNER JOIN
    fact_trip ft ON ds.station_id=ft.to_station_id) AS TrS ON FrS.trip_id=TrS.trip_id
WHERE
    FrS.station_id != TrS.station_id;
```

Data Analysis and Visualization

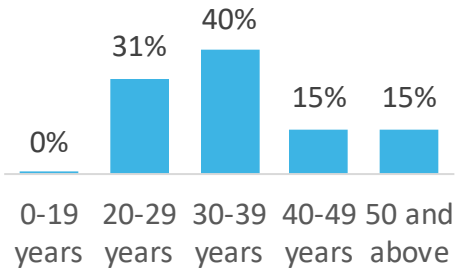
Customer Profiling

Users Type

Gender

74% 26%

Age Group



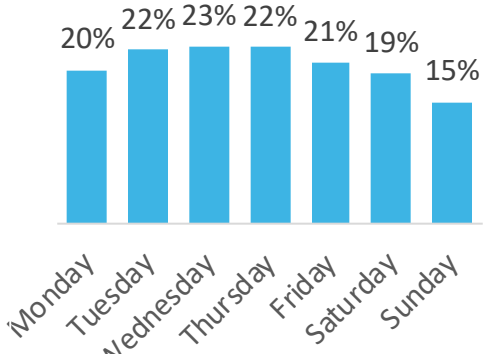
Average Distance Travel

0.95 miles

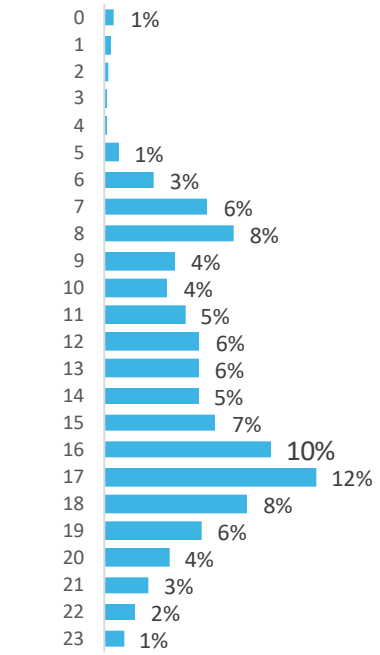
Average Trip Duration

24 minutes/trip

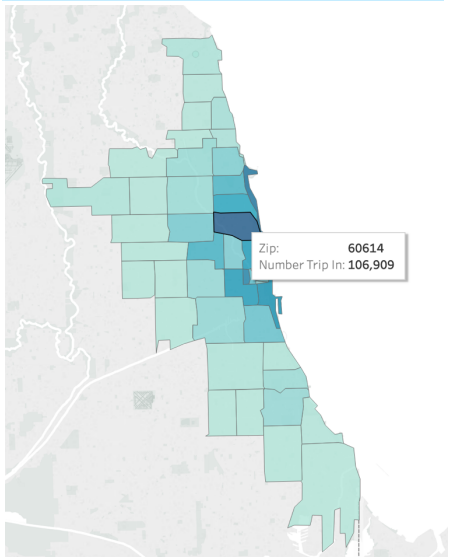
Trip by day



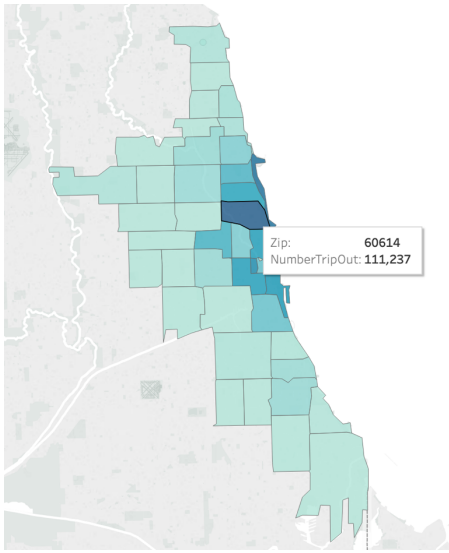
Trip by hour



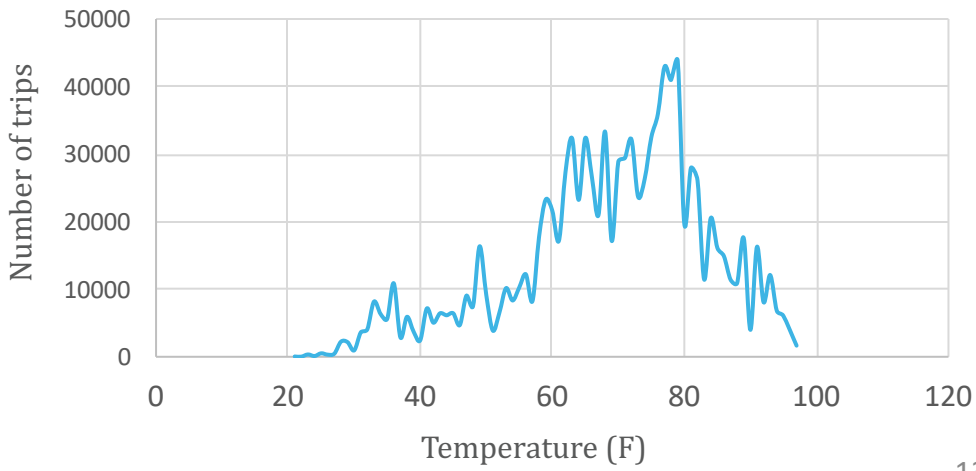
Trip-in by area



Trip-out by area



Trip by weather



Base: Divvy Trips data Q2-2018 (N = 1.05 million)

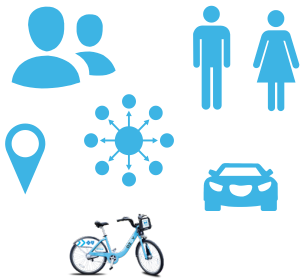
Findings by zip code

Zipcode Analysis

	Population	Gender		Age										Traffic	Bike Racks	Divvy Stations				Divvy Trips				
	Total	Male%	Female%	0_19_M	20_29_M	30_39_M	40_49_M	50plus_M	0_19_F	20_29_F	30_39_F	40_49_F	50plus_F	Vehicle Volume	Number of Bike racks	Number of stations	Total # of docks	Avg # of docks	Avg of Avg Distance of stations from other stations (miles)	Trips Out	Trips In	Net	% of total trips	Subscriber %
60605	24672	48.5%	51.5%	0.0678	0.138	0.1094	0.0565	0.1134	0.0719	0.1451	0.1169	0.0603	0.1206	8,300	356	19	550	28.9	4.75	68,902	65,243	(3,659)	6.33%	60.7%
60601	11115	49.4%	50.6%	0.0639	0.1702	0.093	0.0505	0.1103	0.0716	0.1744	0.0953	0.0517	0.113	23,800	1	11	364	33.1	4.59	68,506	63,594	(4,912)	6.23%	72.4%
60609	64911	50.2%	49.8%	0.1753	0.079	0.0748	0.0623	0.11	0.1737	0.0785	0.0743	0.062	0.11	8,100	75	28	314	11.2	6.49	2,553	2,698	145	0.25%	87.5%
60649	46654	43.8%	56.2%	0.1172	0.0612	0.0563	0.0539	0.1434	0.1503	0.0785	0.0723	0.0769	0.184	18,300	31	14	170	12.1	9.79	1,522	1,462	(60)	0.14%	62.0%
60614	66623	47.6%	52.4%	0.0751	0.1642	0.0888	0.0527	0.0951	0.0827	0.191	0.0975	0.0581	0.1049	21,700	195	34	638	18.8	4.82	106,909	111,237	4,328	10.29%	73.3%
60608	82743	52.4%	47.6%	0.1509	0.1084	0.0693	0.0658	0.1067	0.1355	0.0868	0.0618	0.0598	0.1004	19,800	109	27	370	13.7	5.03	14,489	15,288	799	1.41%	87.8%
60622	52553	51.5%	48.3%	0.0882	0.1414	0.1354	0.0592	0.0868	0.0845	0.1353	0.1295	0.0566	0.0831	34,400	354	25	465	18.6	4.57	45,114	46,260	1,146	4.31%	86.0%
60606	2314	49.2%	50.8%	0.0843	0.1517	0.1106	0.0519	0.0938	0.0869	0.1564	0.1141	0.0536	0.0968	10,200	#N/A	6	170	28.3	4.40	37,508	34,087	(3,421)	3.38%	90.9%
60607	23302	49.2%	50.8%	0.0867	0.1485	0.1137	0.0522	0.0906	0.0897	0.1535	0.1176	0.054	0.0936	29,300	62	26	453	17.4	4.40	61,385	61,617	232	5.80%	91.1%
60642	19485	51.1%	48.9%	0.0858	0.1424	0.1361	0.059	0.0874	0.0821	0.1364	0.1303	0.0565	0.094	11,100	9	12	217	18.1	4.37	24,418	24,523	107	2.31%	90.8%
60610	37730	46.4%	53.6%	0.0397	0.1339	0.0915	0.0533	0.1456	0.0459	0.1547	0.1057	0.0615	0.1682	22,100	122	18	403	22.4	4.52	61,694	62,402	708	5.86%	81.9%
60611	28722	46.4%	53.6%	0.0394	0.1337	0.0915	0.0533	0.146	0.0456	0.1545	0.1058	0.0616	0.1687	18,100	149	16	372	23.3	4.67	89,207	95,133	5,926	8.70%	58.4%
60654	14880	46.7%	53.3%	0.0421	0.1342	0.0941	0.0536	0.1426	0.0476	0.1534	0.1072	0.0613	0.1638	23,000	115	14	356	25.4	4.40	68,187	65,381	(2,806)	6.30%	88.9%
60604	575	49.4%	50.6%	0.0696	0.1704	0.0939	0.0504	0.1096	0.0713	0.1739	0.0957	0.0522	0.113	#N/A	#N/A	3	85	28.3	4.55	15,680	15,230	(450)	1.46%	70.4%
60603	497	49.3%	50.7%	0.0704	0.169	0.0926	0.0503	0.1107	0.0724	0.1751	0.0946	0.0523	0.1127	13,700	38	5	135	27.0	4.56	27,542	24,721	(2,821)	2.47%	
60616	48437	48.1%	51.9%	0.0956	0.0954	0.0776	0.0622	0.1499	0.1025	0.1036	0.0838	0.0668	0.1626	6,100	87	29	447	15.4	5.35	32,058	32,470	412	3.05%	
60602	1210	49.3%	50.7%	0.0702	0.1702	0.0926	0.0504	0.1099	0.0719	0.1744	0.095	0.0521	0.1132	#N/A	#N/A	3	77	25.7	4.53	17,152	17,130	(22)	1.62%	
60661	7736	49.2%	50.8%	0.0882	0.1464	0.1157	0.0524	0.0887	0.0913	0.1514	0.1196	0.0542	0.0916	15,600	102	12	354	29.5	4.34	78,847	76,413	(2,434)	7.33%	93
60637	49508	44.9%	55.1%	0.1343	0.0778	0.061	0.0545	0.121	0.1663	0.094	0.0748	0.0672	0.1484	20,900	72	17	264	15.5	8.15	12,684	12,429	(255)	1.19%	81
60657	66001	49.7%	50.3%	0.0525	0.1862	0.1358	0.0675	0.0532	0.188	0.1165	0.0565	0.0885	0.1295	12,900	201	20	371	18.6	5.28	54,456	57,328	2,872	5.28%	80.5%
60647	87297	50.4%	49.6%	0.1187	0.1217	0.1112	0.0616	0.0911	0.1172	0.1193	0.109	0.0606	0.0837	10,600	204	24	376	15.7	5.43	26,516	27,351	835	2.54%	82.6%
60612	33478	48.4%	51.6%	0.1148	0.1163	0.0933	0.0559	0.0979	0.1248	0.1217	0.1041	0.0598	0.1054	29,500	76	19	335	17.6	4.68	13,200	12,552	(648)	1.22%	91.3%
60615	40608	44.9%	55.1%	0.0991	0.0974	0.0659	0.0529	0.1336	0.1239	0.1159	0.0909	0.0656	0.1648	10,800	64	11	177	16.1	7.54	10,823	10,907	84	1.03%	76.7%
60618	92089	50.2%	49.8%	0.1238	0.0956	0.1017	0.0701	0.1107	0.1223	0.095	0.1014	0.0697	0.1097	18,700	193	21	308	14.7	5.95	10,973	11,778	805	1.07%	85.2%
60613	46285	50.4%	49.6%	0.0602	0.164	0.1156	0.0628	0.1016	0.0783	0.1625	0.1133	0.0612	0.0992	11,600	76	23	426	18.5	5.90	43,431	44,295	864	4.14%	79.5%
60653	29912	43.2%	56.8%	0.1143	0.0672	0.0641	0.0549	0.1315	0.1517	0.0872	0.0846	0.0725	0.1721	19,900	59	3	123	13.7	6.62	2,101	2,277	176	0.21%	73.3%
60625	78654	50.0%	50.0%	0.119	0.0959	0.1004	0.0676	0.1173	0.1179	0.0958	0.1007	0.0677	0.1178	23,800	286	19	285	15.0	6.85	11,857	12,247	390	1.14%	87.8%
60640	65796	51.8%	48.2%	0.0779	0.1109	0.1109	0.0783	0.1399	0.0727	0.1031	0.1032	0.0729	0.1301	16,700	205	14	282	20.1	6.71	22,613	23,060	447	2.16%	80.4%
60619	63630	43.6%	56.4%	0.1177	0.0521	0.0497	0.0572	0.1535	0.152	0.0674	0.0642	0.074	0.2063	17,800	31	17	163	10.8	9.86	638	684	46	0.06%	77.6%
60634	74302	49.1%	50.9%	0.1233	0.0657	0.0711	0.0693	0.1588	0.1273	0.0721	0.0736	0.0708	0.165	21,800	112	1	19	19.0	5.33	3,164	3,693	529	0.52%	79.8%
60626	50144	50.4%	49.6%	0.1136	0.1076	0.0951	0.0715	0.1165	0.1116	0.1057	0.0935	0.0703	0.1145	7,100	145	15	233	15.5	8.92	7,235	7,295	60	0.69%	84.5%
60621	35916	44.8%	55.2%	0.1517	0.0654	0.0505	0.0543	0.1253	0.187	0.0806	0.0623	0.067	0.1553	15,000	28	12	130	10.8	8.28	395	381	(14)	0.04%	86.3%
60624	36109	46.6%	53.4%	0.161	0.0633	0.0607	0.0589	0.1163	0.1844	0.0732	0.0694	0.0674	0.1334	11,800	35	8	88	11.0	5.71	339	320	(19)	0.03%	68.1%
60623	92112	53.7%	46.4%	0.1948	0.0931	0.0851	0.0674	0.1001	0.1638	0.0834	0.0775	0.0579	0.0898	13,700	109	6	66	11.0	5.88	560	588	28	0.06%	82.0%
60645	45280	49.5%	50.5%	0.1342	0.0763	0.078	0.0683	0.1404	0.1369	0.0777	0.0774	0.0696	0.1433	11,400	55	7	95	12.1	8.99	1,296	1,297	1	0.12%	77.2%
60659	38109	49.1%	50.9%	0.1325	0.0713	0.0704	0.0661	0.1507	0.1372	0.0739	0.0729	0.0685	0.1565	44,000	19	4	44	11.0	7.79	601	687	86	0.06%	75.5%
60660	42757	51.2%	48.8%	0.0844	0.1033	0.0959	0.0791	0.1496	0.0806	0.0982	0.0912	0.0753	0.1424	35,200	42	5	91	18.2	7.57	5,215	5,686	471	0.51%	79.7%
60641	71668	49.6%	50.4%	0.1358	0.0791	0.0803	0.0637	0.1312	0.1377	0.0802	0.0815	0.0708	0.1398	33,900	81	6	78	13.0	7.21	1,173	1,110	(63)	0.11%	77.2%
60630	54099	49.2%	50.8%	0.1214	0.0715	0.0766	0.0715	0.1439	0.1253	0.0733	0.0789	0.0741	0.1563	12,300	37	2	26	13.0	7.28	240	239	9	0.02%	73.5%
60651	64273	47.3%	52.7%	0.1552	0.0734	0.0646	0.0538	0.1201	0.1724	0.0814	0.0717	0.0667	0.1348	34,900	31	4	44	11.0	6.25	804	795	(9)	0.08%	70.0%
60644	48852	45.3%	54.1%	0.1412	0.0647	0.0573	0.0594	0.136	0.1667	0.0764	0.0677	0.0701	0.1605	9,800	43	13	139	10.7	6.89	278	259	(19)	0.03%	69.4%
60636	40323	46.5%	53.4%	0.1587	0.0671	0.0527	0.0587	0.1287	0.1818	0.0769	0.0603	0.0673	0.1479	18,200	20	8	80	10.0	8.10	105	109	4	0.01%	76.2%
60617	94161	46.5%	53.5%	0.1439	0.0596	0.0573	0.0595	0.1458	0.1643	0.0681	0.0654	0.0674	0.1596	8,600	190	6	62	10.3	10.84	191	189	(2)	0.02%	90.1%
60201	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	9	135	15.0	11.62	3,783	3,729	(54)	0.36%	77.3%
60202	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	3	45	15.0	10.36	1,141	1,243	102	0.11%	80.5%
60208	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	2	30	15.0	11.80	1,876	1,850	(26)	0.18%	80.5%
60639	39411	49.5%	50.5%	0.1627	0.0802	0.0752	0.0642	0.1128	0.1657	0.0815	0.0764	0.0655	0.1166	12,500	45	1	11	11.0	6.04	200	191	(9)	0.02%	75.5%

Factors considered for analysis. For each zip code we found:

- Total population
- Male & Female %
- %s of different age groups
- Vehicle volume
- Number of bike racks
- Number of stations
- Number of docks
- Avg of avg distance of stations from other stations
- Trip Out, Trip In, Net
- % of subscribers



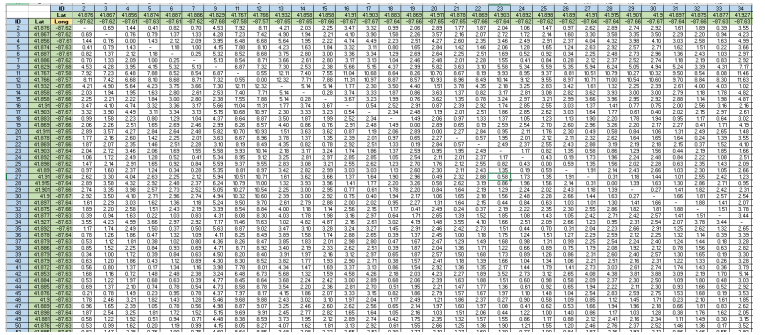
Sample zip code findings

Zip code	60605
Total Population	24672
Male%	0.4852
Female%	0.5148
0_19_M	0.0678
20_29_M	0.138
30_39_M	0.1094
40_49_M	0.0565
50plus_M	0.1134
0_19_F	0.0719
20_29_F	0.1451
30_39_F	0.1169
40_49_F	0.0603
50plus_F	0.1206
Vehicle Volume	8300
Number of Bike racks	356
Number of stations	19
Total # of docks	550
Avg # of docks	28.94736842
Avg of Avg Distance of stations from other stations (miles)	4.754563244
Trips Out	68902
Trips In	65243
Net	-3659
% of total trips	0.063305521
Subscriber %	0.606542626

Sample calculation

**** Sample calculation***

- Distance between stations



- Latitude & Longitude for 608 existing stations
- Used complex formula involving trigonometry to find distances
  - *=IFERROR(6371*ACOS(COS(RADIANS(90-$B563))*COS(RADIANS(90-VS$2))+SIN(RADIANS(90-$B563))*SIN(RADIANS(90-VS$2))*COS(RADIANS($C563-VS$3)))/1.609,0)*
- Resulting in over 30k distance values for each pair of stations

Score based approach

Current station locations (Before expansion plan)

Where are the stations?

- CTA, Metra stations
- employment centers, shopping districts, medical centers, schools
- other popular destinations.

How were the locations chosen?

- population density
- business permits
- other stations in the surrounding network.

Our scoring methodology

- When Divvy first launched, it focused more on the popular destinations (tourist attraction areas, shopping centers, offices etc.)
- The expansion plan is focused more on expanding to the areas where there are currently *no Divvy stations*
- Priority = underserved communities (in terms of number of Divvy stations).
- Score based system for the allocation of the stations and the bikes taking into consideration the below factors. New station allocation determined based on overall score (i.e. higher score = more stations)

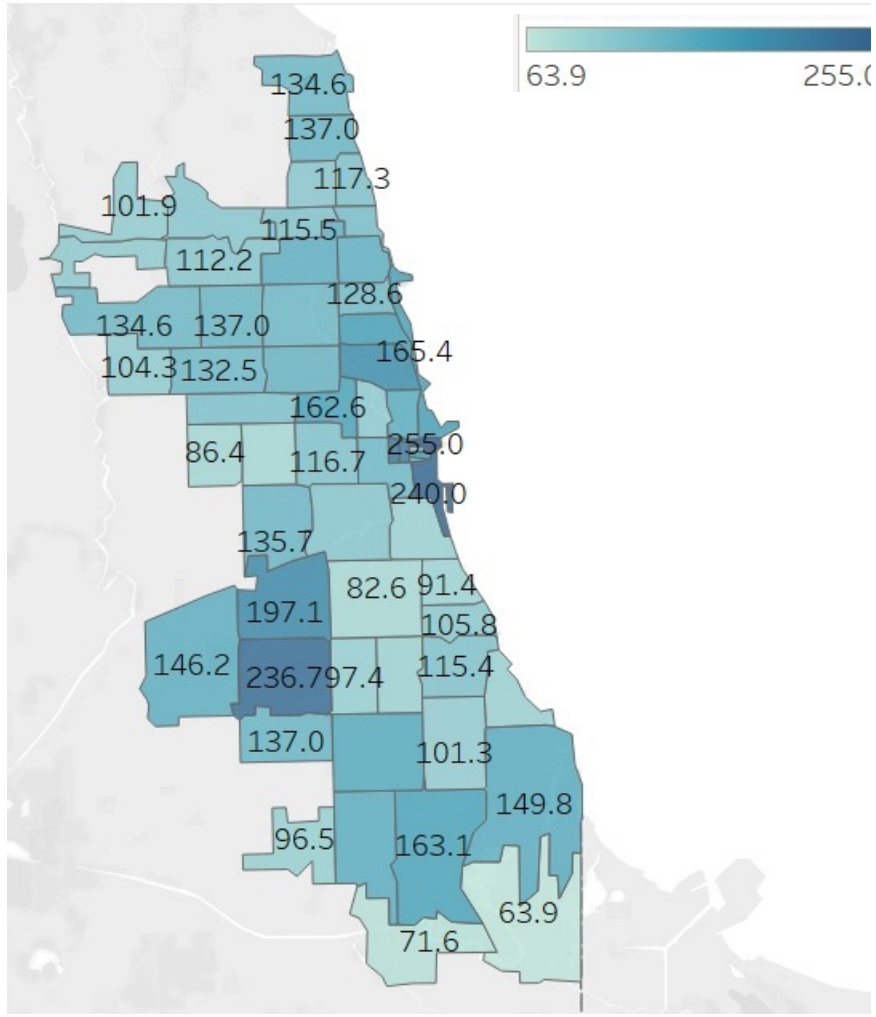
Category	Score Description		Weight	Comments
Divvy Stations (existing)	less number of stations = more points	↓ ↑	20%	More weight assigned to zip codes with no stations. Points deducted to zip codes with stations
Trips (Trips Out)	more number of trips = more points	↑ ↑	10%	-
Net (Trip From - Trip To)	lower Net value = more points	↓ ↑	5%	Points only added to zip codes with a negative net value
Subscriber%	higher % of subscribers = more points	↑ ↑	15%	-
Population Total	higher population = more points	↑ ↑	15%	-
Male%	higher male % = more points	↑ ↑	5%	-
20_39 Age Group	higher % of 20_39 age group = more points	↑ ↑	10%	-
Average Distance to other stations	higher avg distance to other stations = more points	↑ ↑	10%	-
Traffic	higher vehicle volume = more points	↑ ↑	5%	-
Bike racks	more number of bike racks (bike friendliness score) = more points	↑ ↑	5%	-

Scores by zip code

Scoring by zip calculation

	Demographic			Traffic	Bike Racks	Divvy Stations		Divvy Trips																	
	Population	Gender	20_39	Vehicle Volume	Number of Bike racks	Number of stations	Avg of Avg Distance of stations from other stations (miles)	Trips (Trips Out)	Net (Trip From - Trip To)	Subscriber %	Population	Gender	20_39	Vehicle Volume	Number of Bike racks	Number of stations	Avg of Avg Distance of stations from other stations (miles)	Trips (Trips Out)	Net (Trip From - Trip To)	Subscriber %	Total Score	New stations	New Bikes
	Total	Male%									Total	Male%												
Weight	15%	5%	10%	5%	5%	20%	10%	10%	5%	15%	15%	5%	10%	5%	5%	20%	10%	10%	5%	15%			
Total Points	1500	500	1000	500	500	2000	1000	1000	500	1500													
60605	24672	48.5%	0.5094	8,300	356	19	4.75	68,902	(3,659)	60.7%	13.7	8.7	23.9	3.9	31.9	(31.25)	15.4	65.0	84.3	24.4	240.0	5	308
60601	11115	49.4%	0.5329	23,800	1	11	4.59	68,506	(4,312)	72.4%	6.2	8.8	25.0	11.1	0.1	(18.09)	14.9	64.7	113.2	29.2	255.0	5	328
60609	64911	50.2%	0.3066	8,100	75	28	6.49	2,553	145	87.5%	36.1	9.0	14.4	3.8	6.7	(46.05)	21.1	2.4	-	35.2	82.6	2	106
60649	46654	43.8%	0.2683	18,300	31	14	9.79	1,522	(60)	62.0%	26.0	7.8	12.6	8.6	2.8	(23.03)	31.8	1.4	1.4	25.0	94.3	2	121
60614	66823	47.6%	0.5313	21,700	195	34	4.82	106,909	4,328	79.3%	37.1	8.5	24.9	10.2	17.5	(55.92)	15.7	100.9	-	31.9	190.7	4	245
60608	82743	52.4%	0.3789	18,800	109	27	5.03	14,489	799	87.8%	46.0	9.4	17.8	8.8	9.8	(44.41)	16.3	13.7	-	35.4	112.7	2	145
60622	52553	51.1%	0.5416	34,400	354	25	4.57	45,114	1,146	86.0%	29.2	9.1	25.4	16.1	31.7	(41.12)	14.8	42.6	-	34.7	162.6	3	209
60606	2314	49.2%	0.5328	10,200	227	6	4.40	37,508	(3,421)	90.9%	1.3	8.8	25.0	4.8	20.3	(9.87)	14.3	35.4	78.8	36.6	215.4	5	277
60607	23902	49.2%	0.5333	28,300	62	26	4.40	61,385	232	91.1%	13.3	8.8	25.0	13.7	5.6	(42.76)	14.3	57.9	-	36.7	132.5	3	170
60642	18485	51.1%	0.5452	11,100	9	12	4.37	24,416	107	90.8%	10.3	9.1	25.6	5.2	0.8	(19.74)	14.2	23.0	-	36.6	105.1	2	135
60610	37730	46.4%	0.4958	22,100	122	18	4.52	61,694	708	81.9%	21.0	8.3	22.8	10.3	10.8	(29.61)	14.7	58.2	-	33.0	149.6	3	192
60611	28722	46.4%	0.4855	18,100	149	16	4.67	69,207	5,926	58.4%	16.0	8.3	22.8	8.5	13.3	(26.32)	15.2	84.2	-	23.5	165.4	4	213
60654	14880	46.7%	0.4889	23,000	115	14	4.40	68,187	(2,806)	88.9%	8.3	8.3	22.9	10.8	10.3	(23.03)	14.3	64.4	64.6	35.8	216.7	5	278
60604	575	49.4%	0.5339	11,000	227	3	4.55	15,680	(450)	80.4%	0.3	8.8	25.1	5.1	20.3	(4.93)	14.8	14.8	10.4	32.4	127.1	3	163
60603	497	49.3%	0.5313	13,700	98	5	4.56	27,542	(2,821)	55.5%	0.3	8.8	24.9	6.4	8.8	(8.22)	14.8	26.0	65.0	22.4	169.2	4	217
60616	48437	48.1%	0.3604	6,100	97	29	5.35	32,058	412	77.6%	27.0	8.6	16.9	2.9	7.8	(47.70)	17.4	30.3	-	31.2	94.3	2	121
60602	1210	49.3%	0.5322	18,750	50	3	4.53	17,152	(22)	79.3%	0.7	8.8	25.0	8.8	4.4	(4.93)	14.7	16.2	0.5	31.9	106.1	2	136
60661	7798	49.2%	0.5331	15,600	102	12	4.34	78,847	(2,434)	93.3%	4.3	8.8	25.0	7.3	9.1	(19.74)	14.1	74.4	56.1	37.6	217.0	5	279
60637	49508	44.9%	0.3076	20,900	72	17	8.15	12,684	(255)	81.4%	27.5	8.0	14.4	9.8	6.4	(27.96)	26.5	12.0	5.9	32.8	115.4	2	148
60657	66001	49.7%	0.606	12,900	201	20	5.28	54,456	2,872	80.5%	36.7	8.9	28.4	6.0	18.0	(32.89)	17.1	51.4	-	32.4	166.2	4	214
60647	87297	50.4%	0.4612	10,600	204	24	5.43	26,516	835	82.6%	48.6	9.0	21.6	5.0	18.3	(39.47)	17.6	25.0	-	33.3	138.9	3	178
60612	33478	48.4%	0.4414	29,500	76	19	4.68	13,200	(648)	91.3%	18.6	8.7	20.7	13.8	6.8	(31.25)	15.2	12.5	14.9	36.8	116.7	2	150
60615	40608	44.9%	0.3601	10,800	64	11	7.54	10,823	84	76.7%	22.6	8.0	16.9	5.1	5.7	(18.09)	24.5	10.2	-	30.9	105.8	2	136
60618	92089	50.2%	0.3937	18,700	193	21	5.95	10,973	805	85.2%	51.2	9.0	18.5	8.7	17.3	(34.54)	19.3	10.4	-	34.3	134.2	3	172
60613	48285	50.4%	0.556	11,600	76	23	5.90	43,431	864	79.5%	26.9	9.0	26.1	5.4	6.8	(37.83)	19.2	41.0	-	32.0	128.6	3	165
60653	29912	43.2%	0.3031	19,900	59	9	6.62	2,101	176	73.3%	16.6	7.7	14.2	9.3	5.3	(14.80)	21.5	2.0	-	29.5	91.4	2	117
60625	78654	50.0%	0.3928	23,800	286	19	6.85	11,857	390	87.8%	43.8	8.9	18.4	11.1	25.6	(31.25)	22.2	11.2	-	35.4	145.5	3	187
60640	65796	51.8%	0.4281	16,700	205	14	6.71	22,613	447	80.4%	36.6	9.3	20.1	7.8	18.4	(23.03)	21.8	21.3	-	32.4	144.6	3	186
60619	63830	43.6%	0.2334	17,800	31	17	9.86	638	46	77.6%	35.5	7.8	11.0	8.3	2.8	(27.96)	32.0	0.6	-	31.2	101.3	2	130
60634	74302	49.1%	0.2865	21,800	112	1	5.33	3,164	529	79.8%	41.3	8.8	13.4	10.2	10.0	(1.64)	17.3	3.0	-	32.1	134.6	3	173
60626	50144	50.4%	0.4019	7,100	145	15	8.92	7,235	60	84.5%	27.9	9.0	18.9	3.3	13.0	(24.67)	29.0	6.8	-	34.0	117.3	3	151
60621	35916	44.8%	0.2588	15,000	28	12	8.28	395	(14)	86.3%	20.0	8.0	12.1	7.0	2.5	(19.74)	26.9	0.4	0.3	34.8	92.3	2	119
60624	38109	46.6%	0.2786	11,800	35	8	5.71	339	(19)	68.1%	21.2	8.3	13.1	5.5	3.1	(13.16)	18.5	0.3	0.4	27.4	84.8	2	109
60623	92112	53.7%	0.3391	13,700	109	6	5.88	560	28	82.0%	51.3	9.6	15.9	6.4	9.8	(9.87)	19.1	0.5	-	33.0	135.7	3	174
60645	45280	49.5%	0.3074	11,400	55	7	8.99	1,296	1	77.2%	25.2	8.9	14.4	5.3	4.9	(11.51)	29.2	1.2	-	31.1	108.7	2	140
60659	38109	49.1%	0.2885	44,000	19	4	7.79	601	86	75.5%	21.2	8.8	13.5	20.6	1.7	(6.58)	25.3	0.6	-	30.4	115.5	2	148
60660	42757	51.2%	0.3886	35,200	42	5	7.57	5,215	471	79.7%	23.8	9.2	18.2	16.5	3.8	(8.22)	24.6	4.9	-	32.1	124.8	3	160
60641	71668	49.6%	0.3211	39,900	81	6	7.21	1,173	(63)	77.2%	39.9	8.9	15.1	18.7	7.3	(9.87)	23.4	1.1	1.5	31.1	137.0	3	176
60630	54099	49.2%	0.3003	12,300	37	2	7.28	230	9	73.5%	30.1	8.8	14.1	5.8	3.3	(3.29)	23.6	0.2	-	29.6	112.2	2	144
60651	64273	47.3%	0.2911	34,900	31	4	6.25	804	(9)	70.0%	35.8	8.5	13.7	16.3	2.8	(6.58)	20.3	0.8	0.2	28.2	119.9	3	154
60644	40852	45.9%	0.2661	9,800	49	13	6.89	278	(19)	69.4%	27.1	8.2	12.5	4.6	4.4	(21.38)	22.4	0.3	0.4	28.0	86.4	2	111
60636	40923	46.6%	0.257	18,200	20	8	8.10	105	4	76.2%	22.8	8.3	12.1	8.5	1.8	(13.16)	26.3	0.1	-	30.7	97.4	2	125
60617	84161	46.5%	0.2504	8,600	190	6	10.84	191	(2)	90.1%	46.8	8.3	11.8	4.0	17.0	(9.87)	35.2	0.2	0.0	36.3	149.8	3	192
60201	#N/A	#N/A	#N/A	#N/A	#N/A	9	11.62	3,783	(54)	77.9%	#N/A	#N/A	#N/A	#N/A	#N/A	(14.80)	37.7	3.6	1.2	31.4	134.6	3	173
60202	#N/A	#N/A	#N/A	#N/A	#N/A	3	10.36	1,141	102	80.5%	#N/A	#N/A	#N/A	#N/A	#N/A	(4.93)	33.6	1.1	-	32.4	137.0	3	176
60208	#N/A	#N/A	#N/A	#N/A	#N/A	2	11.80	1,876	(26)	80.5%	#N/A	#N/A	#N/A	#N/A	#N/A	(3.29)	38.3	1.8	0.6	32.4	91.4	2	117
60639	90411	49.5%	0.3133	12,500	45	1	6.04	200	(9)	75.5%	50.3	8.8	14.7	5.8	4.0	(1.64)	19.6	0.2	0.2	30.4	132.5	3	170
60628	72206	44.8%	0.2338	12,500	103	-	-	-	-	-	40.2	8.0	11.0	5.8	9.2	88.89	-	-	-	-	163.1	3	210
60643	49957	45.9%	0.2177	27,800	93	-	-	-	-	-	27.8	8.2	10.2	13.0	8.3	80.89	-	-	-	-	148.4	3	191
60620	72220	44.0%	0.2335	12,300	55	-	-	-	-	-	40.2	7.9	11.0	5.8	4.9	93.45	-	-	-	-	153.2	3	197
60655	28552	48.3%	0.2528	7,300	46	-	-	-	-	-	15.9	8.6	11.9	3.4	4.1	52.58	-	-	-	-	96.5	2	124
60633	8726	49.8%	0.2484	4,400	18	-	-	-	-	-	4.9	8.9	11.7	2.1	1.6	34.83	-	-	-	-	63.9	1	82
60629	113921	49.3%	0.3115	14,600	157	-	-	-	-	-	63.4	8.8	14.6	6.8	14.1	128.37	-	-	-	-	236.7	5	304
60652	40965	48.0%	0.2614	33,300	35	-	-	-	-	-	22.8	8.6	12.3	15.6	3.1	74.66	-	-	-	-	137.0	3	176
60707	18043	47.9%	0.2801	22,300	59	-	-	-	-	-	10.0	8.6	13.1	10.4	5.3	56.83	-	-	-	-	104.3	2	134
60632	91332	51.1%	0.322	27,800	18	-	-	-	-	-	50.8	9.1	15.1	13.0	1.6	107.38	-	-	-	-	197.1	4	253
60631	28206	47.6%	0.2335	12,300	61	-	-	-	-	-	15.7	8.5	11.0	5.8	5.5	55.55	-	-	-	-	101.9	2	131
60646	27162	48.1%	0.2161	27,600	44	-	-	-	-	-	15.1	8.6	10.1	12.9	3.9	60.71	-	-	-	-	111.4	2	143
60656	26870	48.7%	0.2907	17,500	20	-	-	-	-	-	15.0	8.7	13.6	8.2	1.8	56.61	-	-	-	-	103.9	2	133
60638	50977	48.7%	0.2743	28,700	35	-	-	-	-	-	28.4	8.7	12.9	13.4	3.1	79.64	-	-	-	-	146.2	3	188
60827	6489	41.1%	0.2688	18,636	4	-	-	-	-	-	3.6	7.4	12.5	9.7	0.4	38.99	-	-	-	-	71.5	2	92

Scoring by zip visualization



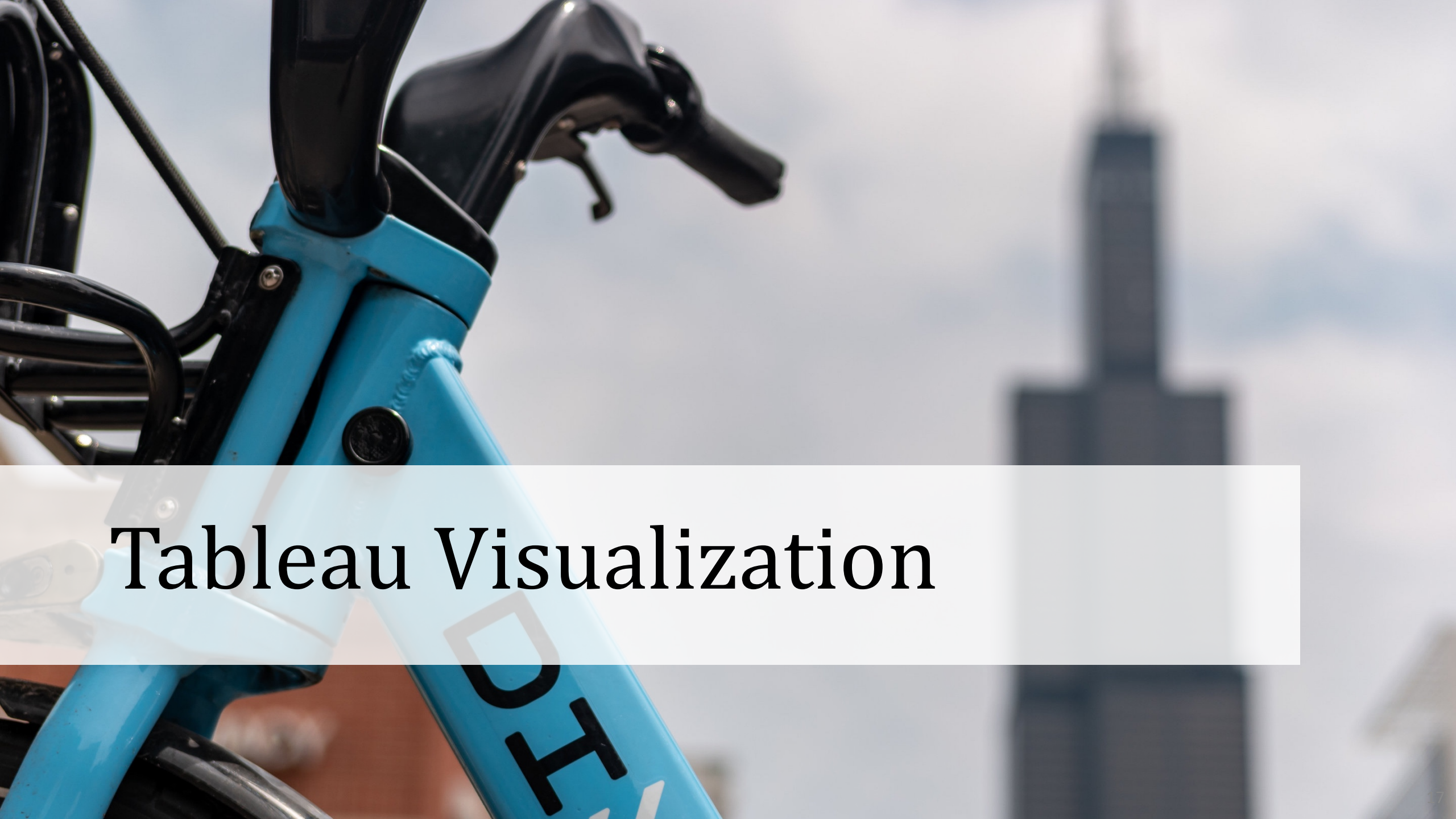
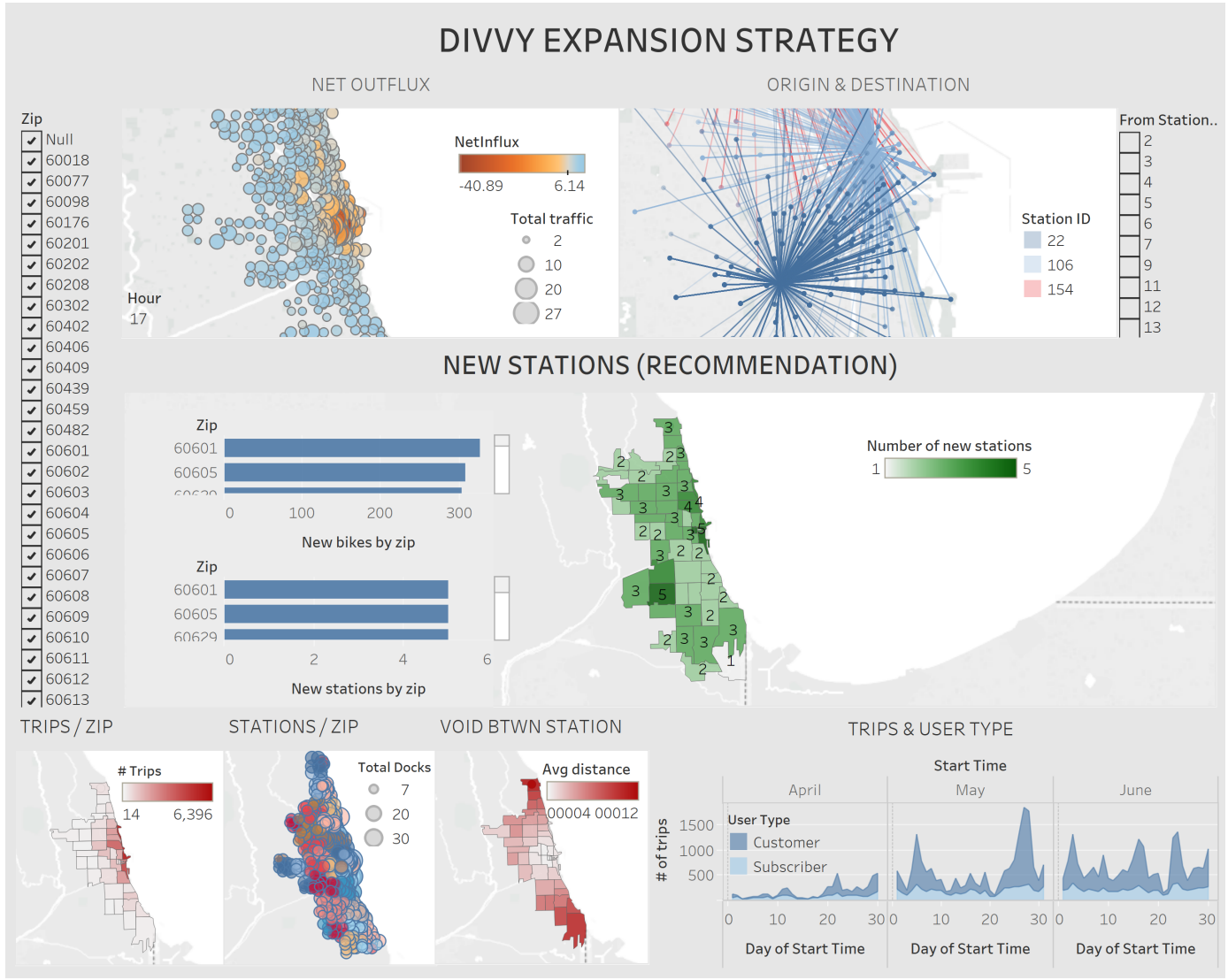


Tableau Visualization

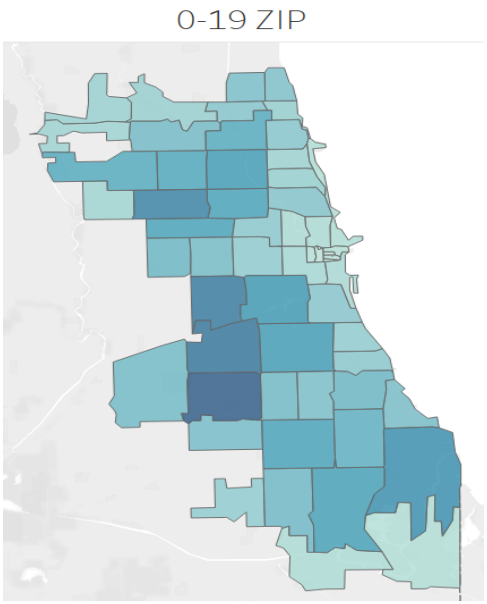
Dashboard

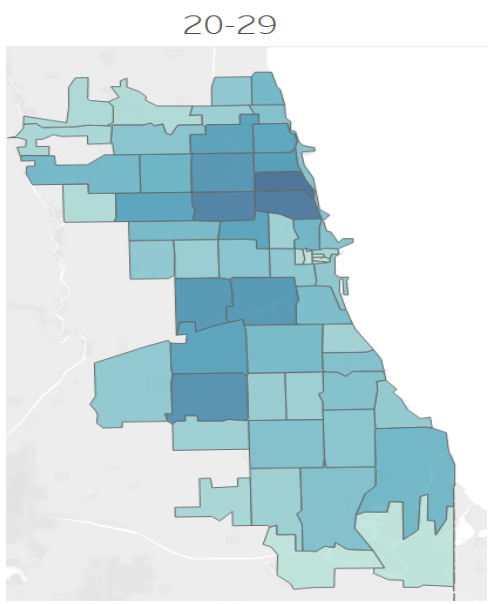


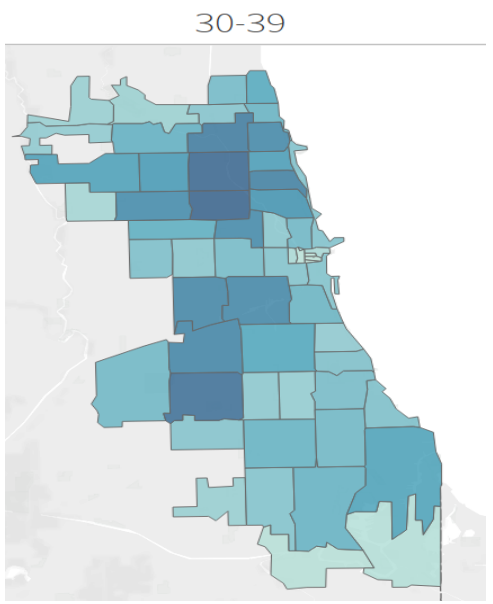
Derived recommendation from trip and zip demographics:

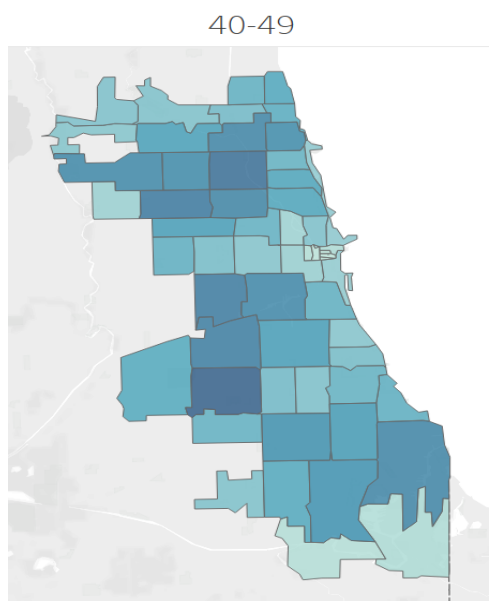
- ***Net Outflux:*** Number of bikes stalled minus number of bikes taken for each station and filtered by hour
- ***Origin & Destination:*** All destinations of the trips taken from a respective station
- ***New Stations (Recommendation):*** Suggested number of new stations per zip code, based on the previously described scoring methodology (+ Number of suggested new bikes and stations per zip code as bar chart)
- ***Trips / Zip:*** Average number of trips started in a respective zip code
- ***Stations / Zip:*** All divvy stations filtered by zip code (color wise) and number of docks (bubble size)
- ***Void Btwn Station:*** Average distance in 100 meters between stations within one zip code
- ***Trips & User Type:*** Number of trips taken filtered by subscribers and non-subscribers ('customers')

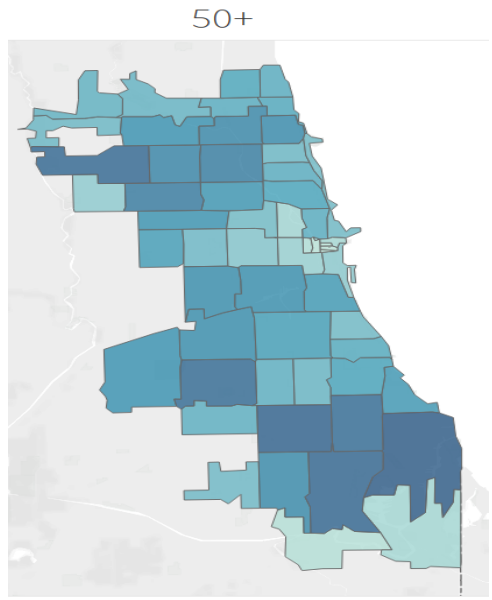
Demographics by Zip Code



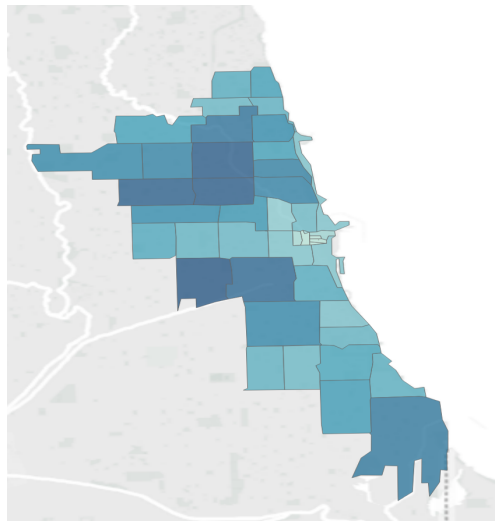




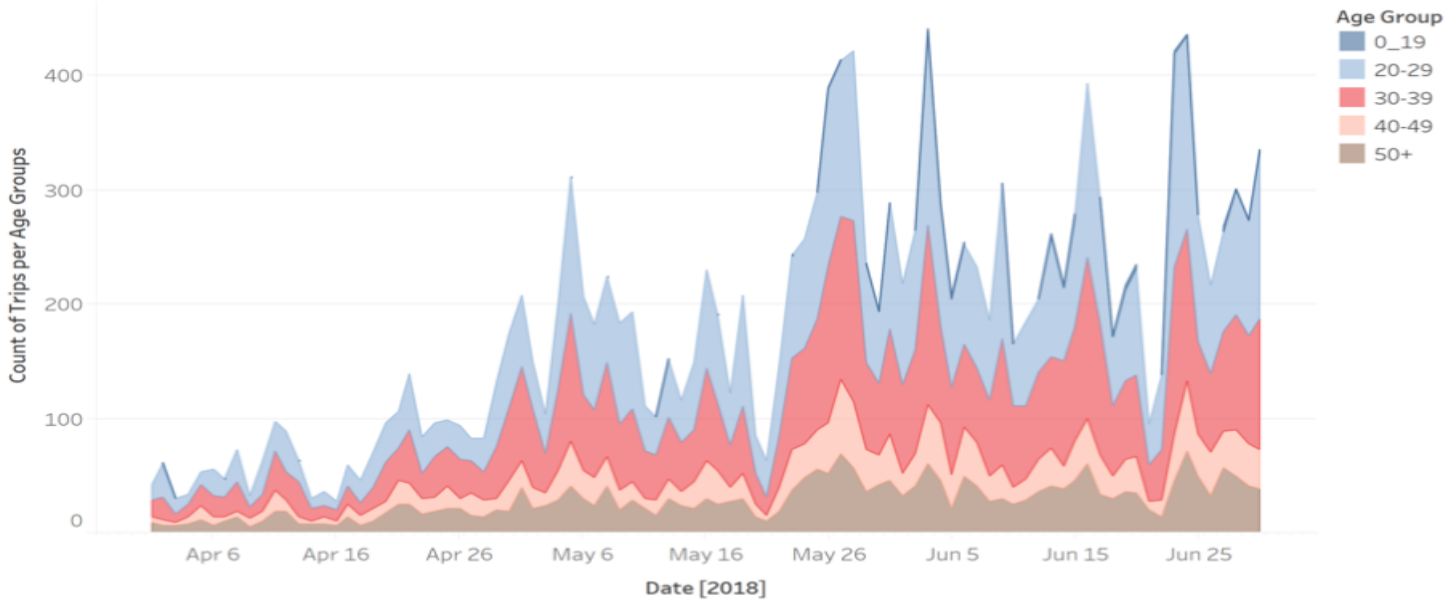




Total population by zip



Number of trips taken by age groups



Summary

Summary

Recommendations and Future Vision:

- Increase stations in ZIPs father from downtown Chicago based on scoring variables to serve the needs of local residents better
- Allocate more bikes to stations with higher net outflux (especially during summer)
- More advance analysis based on variables like customer feedback, commercial footprints, real estate bike scores etc.
- Capitalize on the existing bike rack network in Chicago
- Expand to OLTP framework to support real time trip information.
- Scaling out to support the ever increasing data repository.

Lessons Learned:

- Choose your data sources carefully, every data source has its own conventions and business case.
- Make sure geographic data from different sources is coherent.
- Don't over normalize for OLAP - keep it simple!
- Split up data sources / use views for faster processing in tableau.
- Excel is a very powerful tool.

THANK YOU!