Project 1 Notebook

Class: W205 - Data Engineering

```
Student: Blair Jones

Date: 27 Sept 2020
```

Context: ¶

```
The objective of this project is to work with Lyft Bay Wheels, the company running Bay Area Bikeshare, to try to increase
ridership.  The company may offer deals through the mobile app to do so.

This analysis examines public data provided by Lyft Bay Wheels to identify business opportunities which Lyft Bay Wheels ma
y explore.
```

Lyft Bay Wheels seeks answers to two specific questions:

```
- What are the 5 most popular trips identified as "commuter trips"?
- What are recommendations for offers to increase ridership?
```

In [1]:
```
# Setup required libraries
import pandas as pd
import matplotlib.pyplot as plt
```

Section 1 - What are the 5 most popular trips identified as "commuter trips"?

Let the definition of a "commuter trip" be the following:

- It will have a Subscriber type of "Subscriber" in the table bikeshare_trips. "Commuting trips" are regular trips. Due to the nature of the pricing for subscriptions vs pay-by-the-hour rides, trips which are for commuting purposes are much more likely to be under a subscription pricing plan.
- All days of the week will be considered. It is impossible to differentiate between riders who are "commuting" or not solely based on the day of the week.
- All times of the day will be considered. It is impossible to differentiate between riders who are "commuting" or not solely based on the time of day.
- All trips that are longer than 5 minutes in duration are included.
- Approximately 18% of subscriber trips are less than 5 minutes in duration. This requires further refinement to differentiate between "commuting" trips and "other" trips. In this context, "other" can include casual rides, shopping trips, etc.
  - In aggregate, the top "trips" as determined by the total number of rides between two stations in the service area are all in San Francisco.
  - According to census data for San Francisco (see REF1), the average commuter walk is 15.8 minutes. With an average walking speed of 3 mph (see REF3), the walker will travel a distance of 0.8 miles. If the average commuter biker's pace is 15.5 mph (see REF2), the rider can cover the same distance in 3.1 minutes.
  - How many trips can reasonably be deemed as "not commuter" trips due to their duration being less than 3 minutes?

Total number of trips in dataset

In [2]:
```
%%bigquery --use_rest_api
SELECT COUNT(*)
FROM `bigquery-public-data.san_francisco.bikeshare_trips`
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

Out[2]:

	f0_
0	983648

Total number of "subscriber" trips in dataset

In [3]:
```
%%bigquery --use_rest_api
SELECT COUNT(*)
FROM `bigquery-public-data.san_francisco.bikeshare_trips`
WHERE subscriber_type = 'Subscriber'
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

Out[3]:

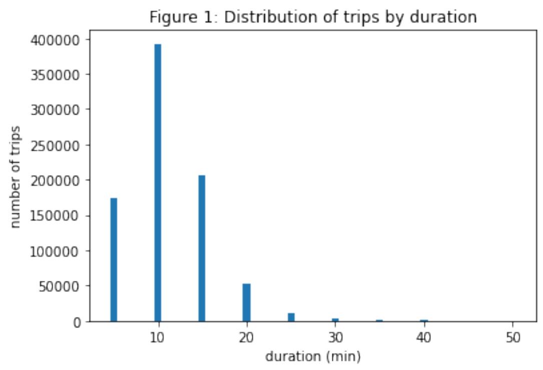
	f0_
0	846839

Number of "subscriber" trips less than 3.1 minutes in duration

In [4]:
```
%%bigquery df_trip_durations_overall --use_rest_api
WITH data AS (
    SELECT duration_sec
    FROM `bigquery-public-data.san_francisco.bikeshare_trips`
    WHERE subscriber_type = 'Subscriber'
)
SELECT ranges, COUNT(duration_sec) as trip_count
FROM data, UNNEST([5*60, 10*60, 15*60, 20*60, 25*60, 30*60, 35*60, 40*60, 45*60, 50*60]) AS ranges
WHERE data.duration_sec > ranges - 5*60
  AND data.duration_sec <= ranges
GROUP BY ranges
ORDER BY ranges
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

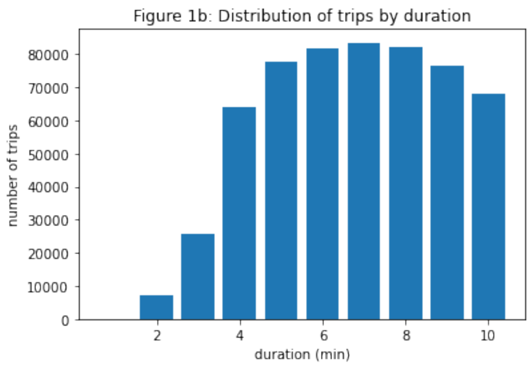
In [5]:
```
fig1, ax1 = plt.subplots()
x1 = df_trip_durations_overall['ranges']/60
y1 = df_trip_durations_overall['trip_count']
ax1.bar(x1, y1)
ax1.set(xlabel='duration (min)', ylabel='number of trips',
       title='Figure 1: Distribution of trips by duration')

plt.show()
```



More granular look at trips with duration less than 5 minutes

In [6]:
```
%%bigquery df_trip_durations_overall --use_rest_api
WITH data AS (
    SELECT duration_sec
    FROM `bigquery-public-data.san_francisco.bikeshare_trips`
    WHERE subscriber_type = 'Subscriber'
)
SELECT ranges, COUNT(duration_sec) as trip_count
FROM data, UNNEST([1*60, 2*60, 3*60, 4*60, 5*60, 6*60, 7*60, 8*60, 9*60, 10*60]) AS ranges
WHERE data.duration_sec > ranges - 1*60
  AND data.duration_sec <= ranges
GROUP BY ranges
ORDER BY ranges
```

In [7]:
```
fig2, ax2 = plt.subplots()
x2 = df_trip_durations_overall['ranges']/60
y2 = df_trip_durations_overall['trip_count']
ax2.bar(x2, y2)
ax2.set(xlabel='duration (min)', ylabel='number of trips',
       title='Figure 1b: Distribution of trips by duration')

plt.show()
```



Roughly 30,000 trips are less than 3 minutes in duration. These will be excluded form the classification of a "commuting trip".

References for the definition of "commuting trips":

- REF1: https://www.governing.com/gov-data/transportation-infrastructure/commute-time-averages-drive-public-transportation-bus-rail-by-metro-area.html (https://www.governing.com/gov-data/transportation-infrastructure/commute-time-averages-drive-public-transportation-bus-rail-by-metro-area.html)
- REF2: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiQhK_Dsf_rAhUJR6wKHSEHAtgQFjACegQIExAF&url=https%3A%2F%2Fbicycles.stackexchange.com%
is-a-reasonable-speed-for-long-distances-on-a-bike&usg=AOvVaw1s4SJBwBgu16L5CjvLysL2 (https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiQhK_Dsf_rAhUJR6wKHSEHAtgQFjACegQIExAF&url=https%3A%2F%2Fbicycles.stackexchange.com%
is-a-reasonable-speed-for-long-distances-on-a-bike&usg=AOvVaw1s4SJBwBgu16L5CjvLysL2)
- REF3: https://en.wikipedia.org/wiki/Walking (https://en.wikipedia.org/wiki/Walking)
- REF4: https://medium.com/data-tale/how-far-do-people-travel-in-bike-sharing-systems-faf0295bc75a (https://medium.com/data-tale/how-far-do-people-travel-in-bike-sharing-systems-faf0295bc75a)

Top 5 "commuting trips"

Parts of this query are inspired by a response to this StackOverflow post: https://stackoverflow.com/questions/42521180/sum-consolidate-combine-reversed-pairs-in-big-query (https://stackoverflow.com/questions/42521180/sum-consolidate-combine-reversed-pairs-in-big-query)

In [8]:
```
%%bigquery df_top_5 --use_rest_api
SELECT endpoint_1_id as station1, endpoint_2_id as station2, trip_count as number_of_trips,
       B.name as station1_name, B.landmark as station1_landmark,
       C.name as station2_name, C.landmark as station2_landmark
FROM (
  WITH pairs AS (
    SELECT
      (SELECT STRING_AGG(CAST(endpoint AS STRING) ORDER BY endpoint)
        FROM UNNEST(ARRAY[start_station_id, end_station_id]) AS endpoint
      ) AS endpoints,
      COUNT(trip_id) AS trip_count
    FROM `bigquery-public-data.san_francisco.bikeshare_trips`
    WHERE subscriber_type = "Subscriber"
      AND duration_sec >= 3 * 60
    GROUP BY endpoints
  )
  SELECT
    REGEXP_EXTRACT(endpoints, r"(\w+),") AS endpoint_1_id,
    REGEXP_EXTRACT(endpoints, r",(\w+)") AS endpoint_2_id,
    trip_count
  FROM pairs
  ORDER BY trip_count DESC
  LIMIT 5
) as A
JOIN (
  SELECT name, station_id, landmark
  FROM `bigquery-public-data.san_francisco.bikeshare_stations`
) as B
ON A.endpoint_1_id = CAST(B.station_id AS STRING)
JOIN (
  SELECT name, station_id, landmark
  FROM `bigquery-public-data.san_francisco.bikeshare_stations`
) as C
ON A.endpoint_2_id = CAST(C.station_id AS STRING)
ORDER BY number_of_trips DESC
```

The top 5 trips in the Lyft service area, as defined by the criteria for a "commuting trip" are:

In [9]:
```
df_top_5
```

Out[9]:

	station1	station2	number_of_trips	station1_name	station1_landmark	station2_name	station2_landmark
0	65	69	13979	Townsend at 7th	San Francisco	San Francisco Caltrain 2 (330 Townsend)	San Francisco
1	50	61	13263	Harry Bridges Plaza (Ferry Building)	San Francisco	2nd at Townsend	San Francisco
2	55	70	10788	Temporary Transbay Terminal (Howard at Beale)	San Francisco	San Francisco Caltrain (Townsend at 4th)	San Francisco
3	50	70	10689	Harry Bridges Plaza (Ferry Building)	San Francisco	San Francisco Caltrain (Townsend at 4th)	San Francisco
4	51	70	10525	Embarcadero at Folsom	San Francisco	San Francisco Caltrain (Townsend at 4th)	San Francisco

NOTE: This analysis was re-run to include trips with duration less than 3 minutes. The results were identical.

Section 2 - Recommendations for offers to increase ridership

A core hypothesis is that promotional activities influence ridership levels. We did not have access to promotional data to determine the causality between different types of promotions and ridership levels.

Instead, we looked at other operational areas for indicators where ridership could be increased. We used historical data on bike stations status, bike trips taken and bike station logistics to answer several key questions:

- Question 1: Which stations are under-utilized (i.e. have many bikes available)? This may signal an opportunity to use special promotions around those stations to increase utilization.
- Question 2: Which stations are over-utilized (i.e. have few bikes available)? This may signal a need to add stations or bikes in the service area.
- Question 3: Are there rider zip codes that are under-served (i.e. there few stations in the zip code compared to the number of trips)? How many rider zip codes are served from existing stations zip codes for SUBSCRIBERS (i.e. better data quality?)? This may signal an opportunity to expand the service area into new locations.
- Question 4: How many riders who match the subscriber type of 'Customer' use bikes in the range of 30-45 minutes? This could be a new segment for a new 45-minute subscription.
- Question 5: What are the top most-ridden bikes and where are they ridden? If this looks like a "thing", make a promotional game, for example, people who find and ride this bike, contributing to a pattern are eligible to win a prize, or a donation to charity is made from the proceeds.

Question 1: Which stations are under-utilized?

We seek to identify stations which are underutilized by looking at the availability of bikes at those stations, averaged over time.

Ideally this data is then considered in the context of population density and commuting patterns around those stations. However this is beyond the scope of this analysis.

We include only weekdays and day-time hours for this analysis.

In [10]:
```
%%bigquery df_under_util --use_rest_api
    SELECT ranges, count(station_id) as station_count
    FROM (
      SELECT station_id,
             status_date,
             status_hour,
             AVG(availability) as avg_availability
      FROM (
        SELECT station_id,
                EXTRACT(DATE FROM time) as status_date,
                EXTRACT(HOUR FROM time) as status_hour,
                EXTRACT(MINUTE FROM time) as status_minute,
                EXTRACT(DAY FROM time) as status_day,
                (bikes_available / (bikes_available + docks_available)) as availability,
        FROM `bigquery-public-data.san_francisco.bikeshare_status`
        WHERE bikes_available > 0 and docks_available > 0
      )
      WHERE status_hour >= 7
          and status_hour <= 20
          and status_day NOT IN (6, 7)
      GROUP BY station_id, status_date, status_hour
      ORDER BY station_id ASC, status_date DESC, status_hour DESC
    ),
    UNNEST([.1, .2, .3, .4, .5, .6, .7, .8, .9, 1]) AS ranges
    WHERE avg_availability > ranges - .1
      AND avg_availability <= ranges
    GROUP BY ranges
    ORDER BY ranges
```

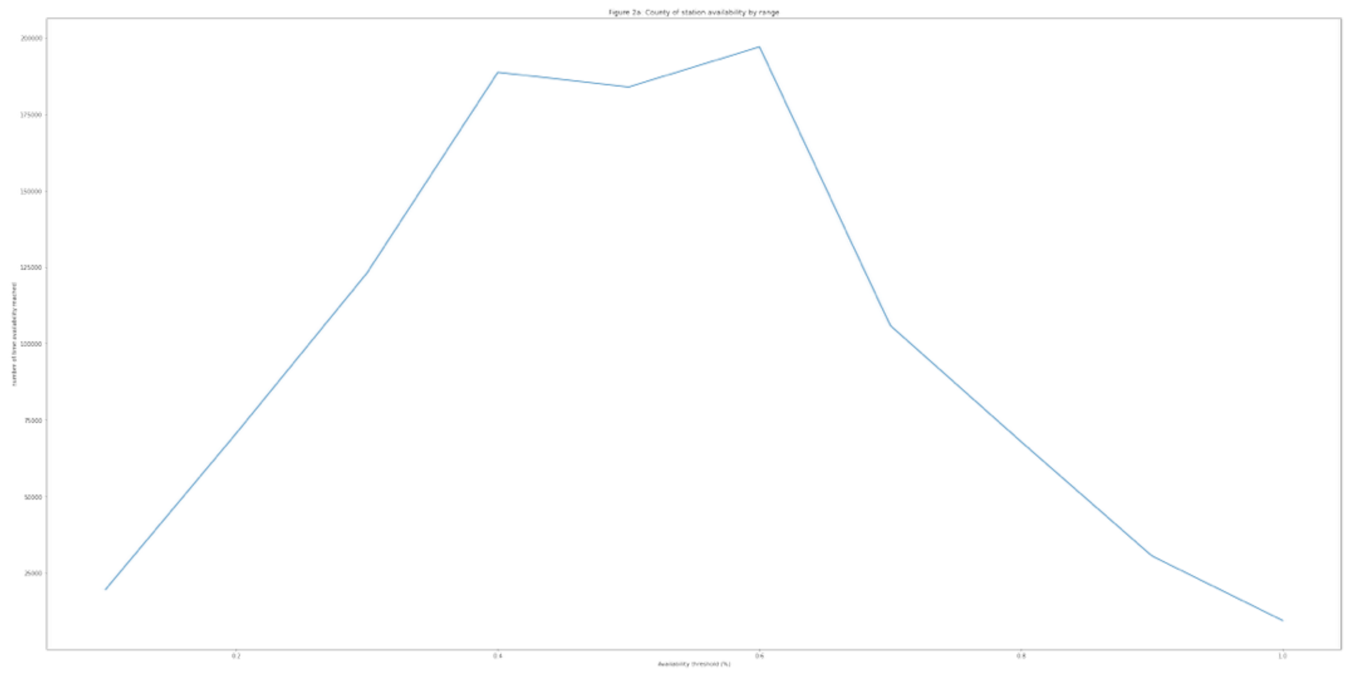
```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

In [11]:
```
df_under_util.head()
```

Out[11]:

	ranges	station_count
0	0.1	19628
1	0.2	70725
2	0.3	123165
3	0.4	188739
4	0.5	183940

In [12]:
```
fig_under, ax_un = plt.subplots(figsize=(40, 20))
x_un = df_under_util['ranges']
y_un = df_under_util['station_count']

ax_un.plot(x_un, y_un)
ax_un.set(xlabel='Availability threshold (%)', ylabel='number of time availability reached',
       title='Figure 2a: County of station availability by range')

plt.show()
```



In [13]:
```
df_under_util.describe()
```

Out[13]:

	ranges	station_count
count	10.000000	10.000000
mean	0.550000	99726.700000
std	0.302765	71694.766612
min	0.100000	9490.000000
25%	0.325000	39949.000000
50%	0.550000	88338.500000
75%	0.775000	168746.250000
max	1.000000	197054.000000

Analysis

From Figure 2a we can see that the overal station availability shows a roughly normal distribution centered on 55% availability.

Lower availability is generally desirable in order to maximize yield on working product, in this case bikes. Similar and other industries routinely set lower targets for availabiltity of their working product, such as 20-30%.

Therefore there is significant room for optimization of the availability of the existing bike inventory.

Recommendation

Create a detailed inventory of stations with higher-than-average availability and explore targeted marketing to attract new riders.

Question 2: Which stations are over-utilized?

We seek to identify stations which are overutilized by looking at the utilization of docks at those stations, averaged over time. Over-utilization signals an opportunity to add bikes and stations to allow for growth in the existing service area.

We include all days of week and times from the last year of operation (2016) for this analysis.

In [16]:
```
%%bigquery df_over_util --use_rest_api
    SELECT ranges, count(station_id) as station_count
    FROM (
      SELECT station_id,
             status_date,
             status_hour,
             AVG(utilization) as avg_utilization
      FROM (
        SELECT station_id,
                EXTRACT(DATE FROM time) as status_date,
                EXTRACT(HOUR FROM time) as status_hour,
                EXTRACT(MINUTE FROM time) as status_minute,
                EXTRACT(DAY FROM time) as status_day,
                (docks_available / (bikes_available + docks_available)) as utilization,
        FROM `bigquery-public-data.san_francisco.bikeshare_status`
        WHERE bikes_available > 0 and docks_available > 0
          and time >= '2016-01-01 00:00:00+00'
      )
      GROUP BY station_id, status_date, status_hour
      ORDER BY station_id ASC, status_date DESC, status_hour DESC
    ),
    UNNEST([0, .1, .2, .3, .4, .5, .6, .7, .8, .9]) AS ranges
    WHERE avg_utilization >= ranges
      AND avg_utilization < ranges + .1
    GROUP BY ranges
    ORDER BY ranges
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

In [17]:
```
df_over_util.head()
```

Out[17]:

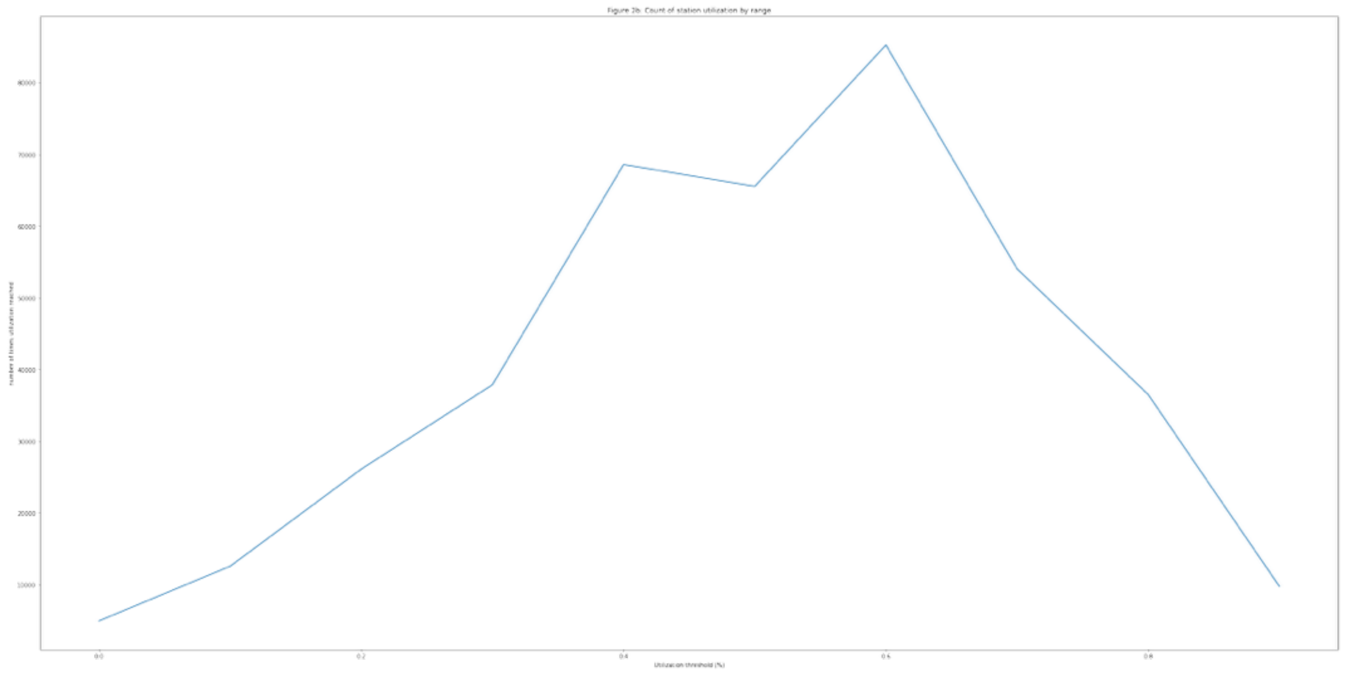
	ranges	station_count
0	0.0	4944
1	0.1	12569
2	0.2	26122
3	0.3	37884
4	0.4	68596

In [18]:
```
df_over_util.describe()
```

Out[18]:

	ranges	station_count
count	10.000000	10.000000
mean	0.450000	40125.800000
std	0.302765	27533.627072
min	0.000000	4944.000000
25%	0.225000	15957.250000
50%	0.450000	37196.500000
75%	0.675000	62681.000000
max	0.900000	85270.000000

In [19]:
```
fig_over, ax_ov = plt.subplots(figsize=(40, 20))
x_ov = df_over_util['ranges']
y_ov = df_over_util['station_count']

ax_ov.plot(x_ov, y_ov)
ax_ov.set(xlabel='Utilization threshold (%)', ylabel='number of times utilization reached',
       title='Figure 2b: Count of station utilization by range')

plt.show()
```



Analysis

The utilization data is centered on 45% utilization. This leaves room for improvement as highlighted in the Question 1 analysis section.

Next we will focus on examining stations showing utilization > 70%. This analysis will narrow in on the last year of date for the year 2016.

In [20]:
```
%%bigquery df_over_stations --use_rest_api
    SELECT status_date, ranges, station_id, count(station_id) as high_count
    FROM (
      SELECT station_id,
             status_date,
             status_hour,
             AVG(utilization) as avg_utilization
      FROM (
        SELECT station_id,
                EXTRACT(DATE FROM time) as status_date,
                EXTRACT(HOUR FROM time) as status_hour,
                EXTRACT(MINUTE FROM time) as status_minute,
                EXTRACT(DAY FROM time) as status_day,
                (docks_available / (bikes_available + docks_available)) as utilization,
        FROM `bigquery-public-data.san_francisco.bikeshare_status`
        WHERE bikes_available > 0 and docks_available > 0
          and time >= '2016-01-01 00:00:00+00'
      )
      GROUP BY station_id, status_date, status_hour
      ORDER BY station_id ASC, status_date DESC, status_hour DESC
    ),
    UNNEST([0, .1, .2, .3, .4, .5, .6, .7, .8, .9, 1]) AS ranges
    WHERE avg_utilization > ranges
      AND avg_utilization <= ranges + .1
    GROUP BY ranges, status_date, station_id
    ORDER BY status_date, ranges, station_id
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

```
In [21]: df_over_stations.head()
```

Out[21]:

	status_date	ranges	station_id	high_count
0	2016-01-01	0.0	55	1
1	2016-01-01	0.0	60	1
2	2016-01-01	0.1	32	24
3	2016-01-01	0.1	55	11
4	2016-01-01	0.1	60	8

Filtering to retain stations which have:

- More than 70% of high utilization
- 6 or more hours per day of high utilization
- High utilization during more than 50% of the year

```
In [22]: first_filter = df_over_stations[df_over_stations['ranges']>=.7]
         grouped_stations = first_filter[first_filter['high_count']>=6].groupby('station_id')
         station_data = grouped_stations.filter(lambda x: len(x['high_count']) > 365/2)
```

```
In [25]: # top high utilization stations
         stations = pd.unique(station_data['station_id'])
         stations

Out[25]: array([45, 47, 51, 62, 63, 67,  5])
```

Plot of utilization data for top utilized stations during 2016

```
In [28]: # data smoothed over 14-day rolling mean to facilitate visualization
         for station in stations:
             plt_data = df_over_stations[df_over_stations['station_id']==station]
             fig_over_stations, ax_os = plt.subplots(figsize=(40, 5))
             ax_os.plot(plt_data['status_date'], plt_data['ranges'].rolling(14).mean())

             title_label = 'Figure 2c: Station utilization by date for station #' + f'{station:.{0}f}'
             ax_os.set(xlabel='Utilization threshold (%)', ylabel='number of times utilization reached',
                    title=title_label)

         plt.show()
```

Figure 2c: Station utilization by date for station #45

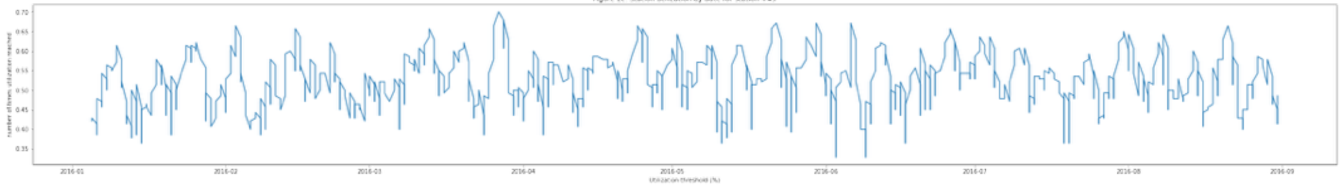


Figure 2c: Station utilization by date for station #47

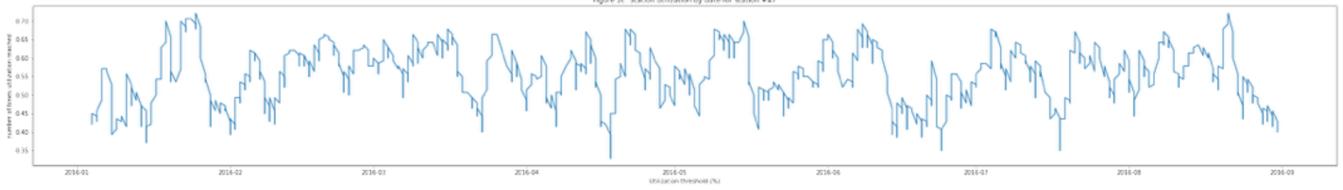


Figure 2c: Station utilization by date for station #51

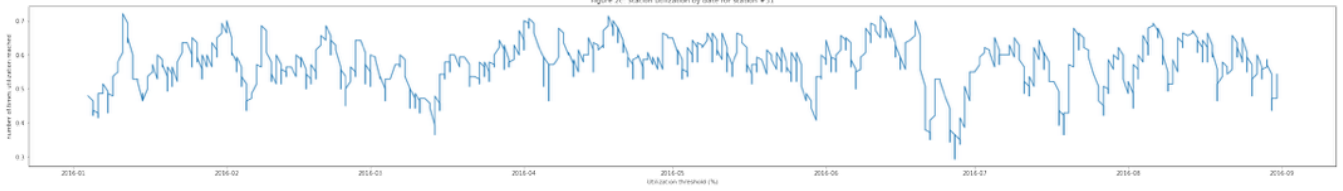


Figure 2c: Station utilization by date for station #62

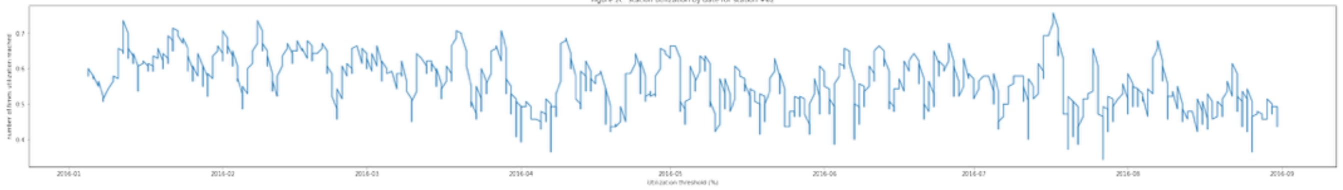


Figure 2c: Station utilization by date for station #63

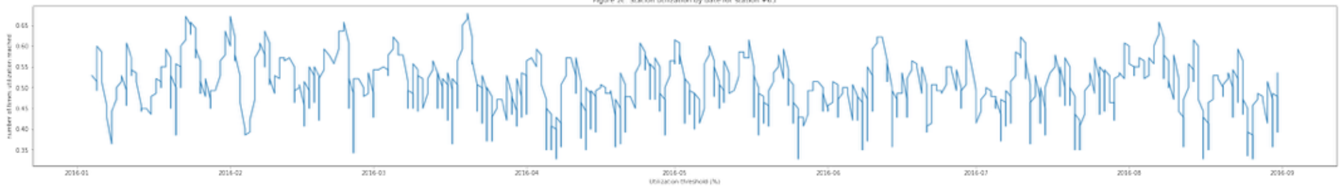


Figure 2c: Station utilization by date for station #67

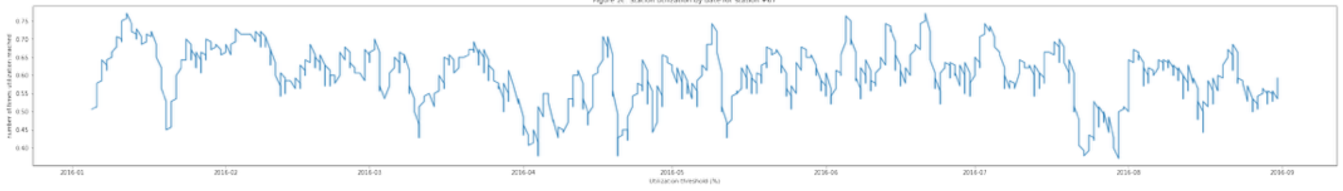
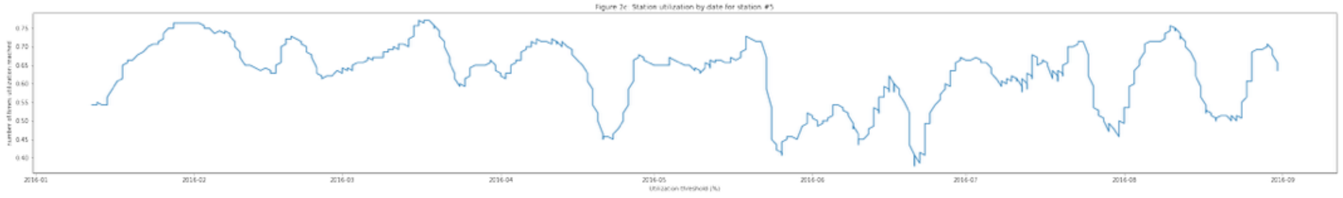


Figure 2c: Station utilization by date for station #5



Analysis

From the Figure 2c charts for each station, we can see that of the top utilized stations:

- There is no discernable consistent growth in utilization during the last year of data
- Some time periods exhibit higher utilization, which suggests that there are opportunities for optimization of bike allocations during such timeframes.

This needs to be considered with additional examination of data for local circumstances surrounding each station. For example, it should be determined if there are additional factors causing the high utilization which may influence decision to expand capacity or not. For example, a temporary work zone-related issue that may have encourage riders for a period of time may indicate that permanently expanding local station capacity is not warranted and a simple temporary solution be more appropriate.

In such cases, a solution such as surge deployment of bikes to stations that are forecast to exceed capacity may be a more financially sound alternative.

Recommendation

- Continue regular analysis of utilization patterns to monitor for sustained utilization growth.
- Develop better forecasting of surge demand situations.
- Explore additional flexible solutions to manage deployment of additional capacity to meet surge demand.

Question 3: Are there zip codes that are under-served?

Lyft serves the San Francisco Bay region with stations concentrated around several locations. Customer billing zip codes are stored for each trip, which can provide a view into where customer home locations. This enables an analysis of customer zip codes to determine if there are zip codes with high numbers of users that could benefit from an expansion of the service area. The expansion could be achieved using one of two approaches:

1. New stations in zip codes which are adjacent to existing service areas to provide longer route options for commuters
2. New stations in isolated zip codes where a new service location might be financially viable

There are known data quality issues for zip codes in the dataset bikeshare_trips due to erroneous user data entry at kiosks for non-subscribers. Non-subscriber data will be excluded for this reason.

First, we look at zip codes in scope from bikeshare_trips to see how many will be excluded from the data due to invalid length.

In [29]:
```
%%bigquery total_records --use_rest_api
    SELECT COUNT(*)
    FROM `bigquery-public-data.san_francisco.bikeshare_trips`
    WHERE subscriber_type = 'Subscriber'
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

In [30]:
```
%%bigquery invalid_zips --use_rest_api
    SELECT COUNT(*)
    FROM `bigquery-public-data.san_francisco.bikeshare_trips`
    WHERE zip_code IN (
        SELECT DISTINCT zip_code
        FROM `bigquery-public-data.san_francisco.bikeshare_trips`
        WHERE subscriber_type = 'Subscriber'
          AND LENGTH(zip_code) != 5
        ORDER BY zip_code ASC
    )
```

In [31]:
```
pct = invalid_zips.iloc[0][0] / total_records.iloc[0][0] * 100

print('The total number of records that will be exluded out of the total dataset')
print(invalid_zips.iloc[0][0], " of ", total_records.iloc[0][0], " records, or ", f'{pct:.{1}f}', "%")
```

```
The total number of records that will be exluded out of the total dataset
1970  of  846839  records, or  0.2 %
```

Next, we visualize the top 50 "unserved" zip codes.

Unserved zip codes are zip codes which do not currently contain an existing station.

The top 50 unserved zip codes are determined by the total number of trips taken by riders in the same unserved zip code. The 50 zip codes with the highest trip count are retained.

We then plot this data, along with the locations of existing stations on a map.

Extracting top 50 unserved zip codes

In [32]:
```
%%bigquery df_underserved_zips --use_rest_api
       SELECT A.zip_code, A.zip_code_geom, B.trip_count
          FROM `bigquery-public-data.geo_us_boundaries.zip_codes` AS A
          JOIN (
              SELECT zip_code, covered, count(trip_id) as trip_count
              FROM (
                SELECT C.trip_id,
                       C.zip_code,
                       C.zip_code IN (
                          SELECT B.zip_code
                          FROM `bigquery-public-data.san_francisco.bikeshare_stations` as A
                          JOIN `bigquery-public-data.geo_us_boundaries.zip_codes` AS B
                          ON ST_CONTAINS(B.zip_code_geom, ST_GeogPoint(A.longitude, A.latitude))
                       ) as covered,
                       C.subscriber_type
                FROM `bigquery-public-data.san_francisco.bikeshare_trips` as C
                ORDER BY C.zip_code ASC
              )
              WHERE covered = FALSE
              GROUP BY zip_code, covered
              ORDER BY trip_count DESC
              LIMIT 50
        ) as B
        ON A.zip_code = B.zip_code
```

In [33]:
```
df_underserved_zips.head()
```

Out[33]:

	zip_code	zip_code_geom	trip_count
0	94070	POLYGON((-122.303554 37.493717, -122.303201 37...	8397
1	94114	POLYGON((-122.454002 37.758785, -122.453799 37...	9544
2	94025	POLYGON((-122.22869 37.424297, -122.229133 37....	9452
3	94121	POLYGON((-122.514897 37.779848, -122.514948 37...	3544
4	94030	POLYGON((-122.428202 37.604967, -122.428149 37...	4484

Plotting the unserved zip codes onto a map

In [58]:
```
import gmaps
gmaps.configure(api_key='AIzaSyAuu5MAexK5-MLQ8btpc1RNbEmBb7qxWWI')
from ipywidgets.embed import embed_minimal_html
from IPython.display import Image
```

In [35]:
```
# data cleanup and transform for mapping lat/lng points in zip code boundary polygons
import re
filter_non_decimal = re.compile(r'[^-\d., ]+')
point_list = []
for zip in df_underserved_zips.index:
    points = []
    for point in df_underserved_zips.iloc[zip]['zip_code_geom'].split(','):
        latlng = filter_non_decimal.sub('', point).split()
        tmp = float(latlng[0])
        latlng[0] = float(latlng[1])
        latlng[1] = tmp
        tup = tuple(latlng)
        points.append(tup)
    point_list.append(points)
df_underserved_zips['latlng_points'] = point_list
df_underserved_zips.head()
```

Out[35]:

	zip_code	zip_code_geom	trip_count	latlng_points
0	94070	POLYGON((-122.303554 37.493717, -122.303201 37...	8397	[(37.493717, -122.303554), (37.493529, -122.30...
1	94114	POLYGON((-122.454002 37.758785, -122.453799 37...	9544	[(37.758785, -122.454002), (37.758737, -122.45...
2	94025	POLYGON((-122.22869 37.424297, -122.229133 37....	9452	[(37.424297, -122.22869), (37.424295, -122.229...
3	94121	POLYGON((-122.514897 37.779848, -122.514948 37...	3544	[(37.779848, -122.514897), (37.779719, -122.51...
4	94030	POLYGON((-122.428202 37.604967, -122.428149 37...	4484	[(37.604967, -122.428202), (37.604752, -122.42...

Setup of the base map and adding a layer for each unserved zip code.

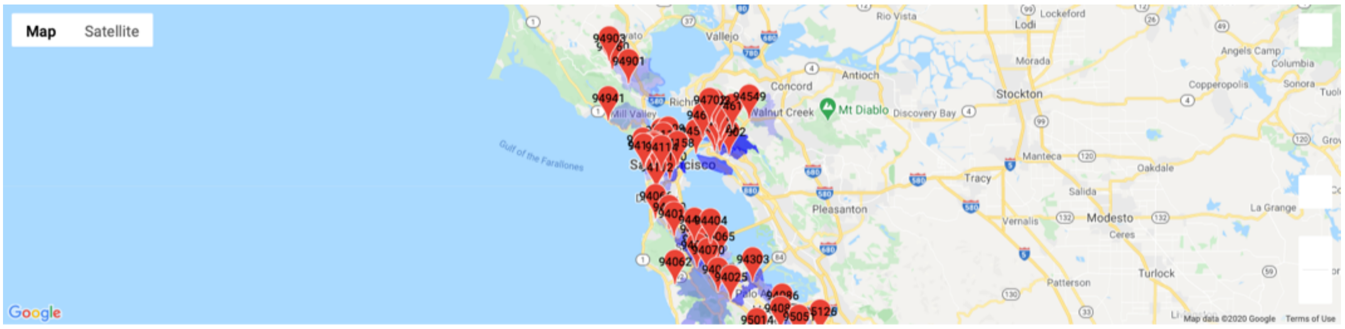
Trip count is normalized and used to highlight zip codes with higher trip counts.

Darker shading indicates higher trip count.

In [60]:
```
san_fran_coordinates = (37.7749, -122.4194)
fig = gmaps.figure(center=san_fran_coordinates, zoom_level=9)

max_vol_trips = max(df_underserved_zips['trip_count'])
min_vol_trips = min(df_underserved_zips['trip_count'])

for zip in df_underserved_zips.index:
    zip_weight = df_underserved_zips.iloc[zip]['trip_count']/max_vol_trips

    zips_polygon = gmaps.Polygon(
        df_underserved_zips.iloc[zip]['latlng_points'],
        stroke_color='blue',
        stroke_weight=zip_weight,
        stroke_opacity=zip_weight,
        fill_color='blue',
        fill_opacity=zip_weight
    )

    zip_labels = gmaps.Marker(df_underserved_zips.iloc[zip]['latlng_points'][0],
                  label=df_underserved_zips.iloc[zip]['zip_code'])

    zips_layer = gmaps.drawing_layer(
        mode='DISABLED',
        features=[zip_labels, zips_polygon],
        show_controls=False
    )
    fig.add_layer(zips_layer)

# fig # This line is used to display the interactive map.  It is then manually saved to .png because it does not render
when exported to PDF.
```

In [59]:
```
Image(filename='map_zips.png') # This line is used to display the map for exporting the notebook to PDF.
```

Out[59]:



Add display of existing bike stations on map.

Existing stations will be shown in a heatmap view to highlight the concentraion of stations around each landmark.

In [37]:
```
%%bigquery df_station_locs --use_rest_api
    SELECT
        station_id,
        ST_GeogPoint(longitude, latitude)  AS WKT,
        dockcount
    FROM `bigquery-public-data.san_francisco.bikeshare_stations`
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

In [38]:
```
df_station_locs.head()
```

Out[38]:

	station_id	WKT	dockcount
0	4	POINT(-121.894902 37.333988)	11
1	37	POINT(-122.159504 37.448598)	11
2	35	POINT(-122.163093 37.444521)	11
3	32	POINT(-122.083678 37.385956)	11
4	84	POINT(-121.895617 37.342725)	15

In [39]:
```
point_list = []
for zip in df_station_locs.index:
    for point in df_station_locs.iloc[zip]['WKT'].split(','):
        latlng = filter_non_decimal.sub('', point).split()
        tmp = float(latlng[0])
        latlng[0] = float(latlng[1])
        latlng[1] = tmp
        tup = tuple(latlng)
    point_list.append(tup)
df_station_locs['latlng_points'] = point_list
df_station_locs.head()
```

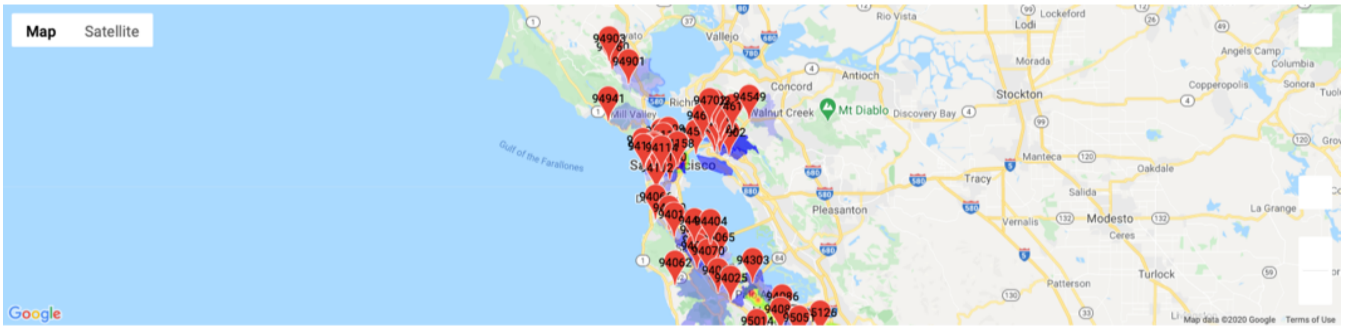
Out[39]:

	station_id	WKT	dockcount	latlng_points
0	4	POINT(-121.894902 37.333988)	11	(37.333988, -121.894902)
1	37	POINT(-122.159504 37.448598)	11	(37.448598, -122.159504)
2	35	POINT(-122.163093 37.444521)	11	(37.444521, -122.163093)
3	32	POINT(-122.083678 37.385956)	11	(37.385956, -122.083678)
4	84	POINT(-121.895617 37.342725)	15	(37.342725, -121.895617)

In [63]:
```
stations_layer = gmaps.heatmap_layer(df_station_locs['latlng_points'])
fig.add_layer(stations_layer)
#fig # This line is used to display the interactive map.  It is then manually saved to .png because it does not render
when exported to PDF.
```

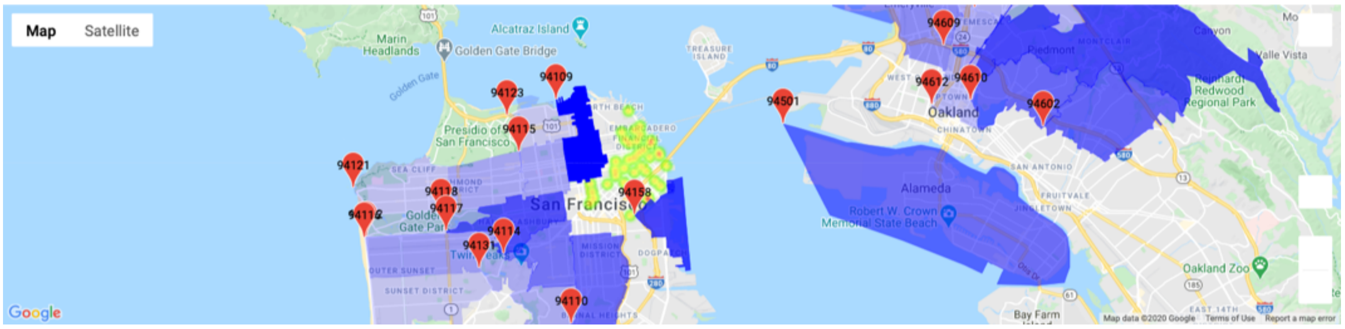
In [62]:
```
Image(filename='map_zips_stations.png') # This line is used to display the map for exporting the notebook to PDF.
```

Out[62]:



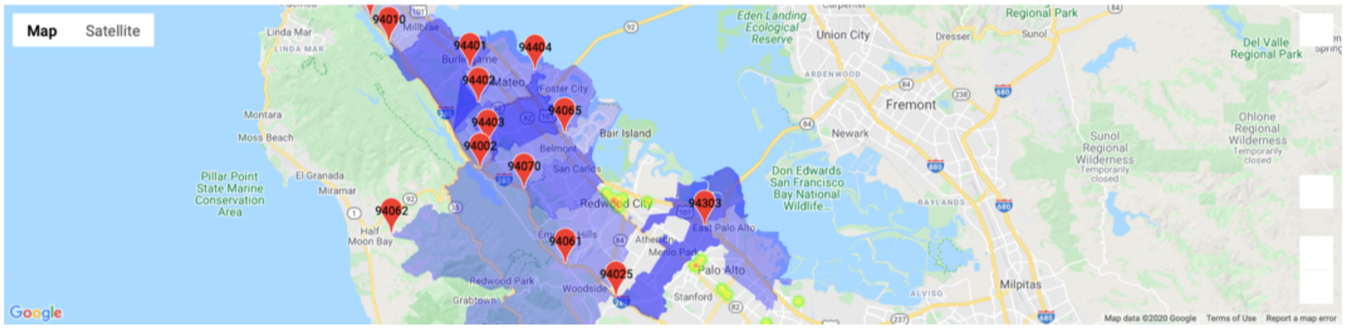
In [64]:
```
Image(filename='map_san_fran.png') # This line is used to display the map for exporting the notebook to PDF.
```

Out[64]:



In [65]:
```
Image(filename='map_redwood.png') # This line is used to display the map for exporting the notebook to PDF.
```

Out[65]:



Analysis

Visual inspection of the map highlights several unserved zip codes with high numbers of riders.

Expansion Approach 1 Candidates (adjacent zip codes):

- Adjacent to San Francisco: 94158, 94109, 94110, 94117, 94114
- Adjacent to Redwood City: 94070, 94061

Expansion Approach 2 Candidates (new service locations):

- Oakland/Alameda: 94501, 94602, 94609, 94610, 94612
- San Mateo: 94401, 94402, 94403

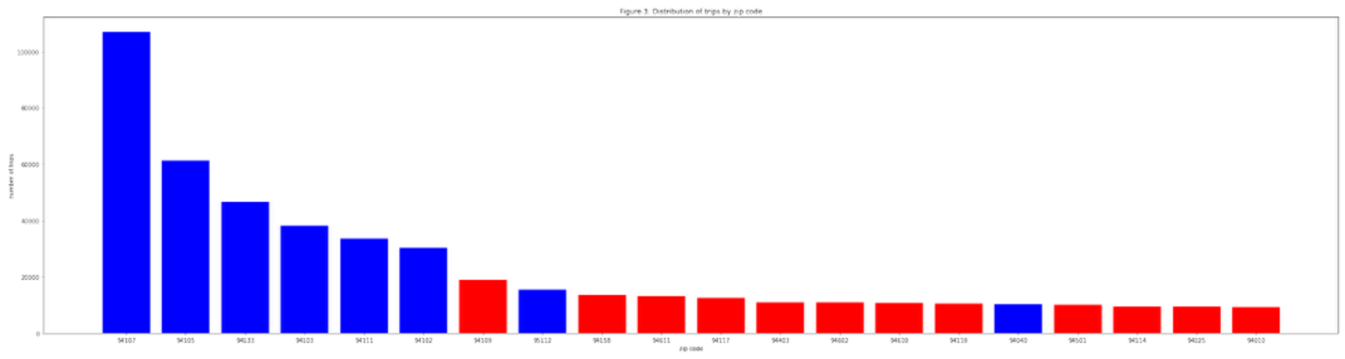
Note: San Mateo is along a transportation corridor that serves multiple existing service locations with bus and train commute. Residents of San Mateo may be in need of local bike-share services which are currently unserved, for example for local commutes.

Examine the volume of trips in unserved zip codes against the volume for served zip codes

In [41]:
```
%%bigquery total_trips_by_zip --use_rest_api
    SELECT zip_code, COUNT(trip_id) as trip_count
    FROM `bigquery-public-data.san_francisco.bikeshare_trips`
    WHERE zip_code IN (
        SELECT DISTINCT zip_code
        FROM `bigquery-public-data.san_francisco.bikeshare_trips`
        WHERE subscriber_type = 'Subscriber'
          AND LENGTH(zip_code) = 5
        ORDER BY zip_code ASC
    )
    GROUP BY zip_code
    ORDER BY trip_count DESC
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

In [43]:
```
fig_zip_vol, ax_zv = plt.subplots(figsize=(40, 10))
x_zv = total_trips_by_zip['zip_code'][:20]
y_zv = total_trips_by_zip['trip_count'][:20]

colors = []
for zip in x_zv.values:
    if str(zip) in df_underserved_zips['zip_code'].values:
        colors.append('r')
    else:
        colors.append('b')

ax_zv.bar(x_zv, y_zv, color=colors)
ax_zv.set(xlabel='zip code', ylabel='number of trips',
       title='Figure 3: Distribution of trips by zip code')

plt.show()
```



From this chart we can see that the unserved zip codes represent a meaningful portion of overall trip volume. This could represent an opportunity to expand the service area and ride volume.

Recommendation

Conduct additional analysis to develop business cases for both approaches and for each of the highlighted zones to expanding the service area, focusing on:

Expansion Approach 1 Candidates (adjacent zip codes):

- Adjacent to San Francisco: 94158, 94109, 94110, 94117, 94114
- Adjacent to San Jose: 95126
- Adjacent to Redwood City: 94070, 94061

Expansion Approach 2 Candidates (new service locations):

- Oakland/Alameda: 94501, 94602, 94609, 94610, 94612
- San Mateo: 94401, 94402, 94403

Question 4: Is there a potential customer segment for a 45-minute subscription?

For the time period covered by the dataset utilized for this analysis, Lyft offered a 30-minute duration subscription. A significant number of rides that are close to 30 minutes in duration implies the possibility that there is a market segment that could benefit from a new 45-minute subscription. This could attract more riders who are unable to ride within the 30-minute free allowance.

In [44]:
```
%%bigquery df_all_rides --use_rest_api
    SELECT A.trip_count as total_rides, B.trip_count as long_rides
    FROM (
        SELECT count(trip_id) as trip_count
        FROM `bigquery-public-data.san_francisco.bikeshare_trips`
        WHERE subscriber_type = 'Subscriber'
    ) as A,
    (
        SELECT count(trip_id) as trip_count
        FROM `bigquery-public-data.san_francisco.bikeshare_trips`
        WHERE subscriber_type = 'Subscriber'
          AND duration_sec > 30*60
          AND duration_sec < 45*60

    ) as B
```

```
In [45]: df_all_rides
```

Out[45]:

	total_rides	long_rides
0	846839	2398

Analysis

From this we can see that the number of rides taken by "Subscriber" customers in the range 30-45 minutes is very small compared to the overall volume of rides. Indeed, also referring to Figure 1 in this document, we can see that ride durations tail off quickly after 10 to 15 minutes duration. The fact that ride durations significantly decrease after 15-20 minutes suggests that there is a very small addressable market for a special ride package of greater than 30 minutes duration.

Recommendation

At this time we recommend no further action on this option.

Question 5: Is there a way to gamify specific bikes?

The premise of this question is to examine if there is a particular pattern to bike usage that suggests an opportunity to employ a promotional "game" to increase ridership.

This approach would involve visibly marking specific bikes (by color or label) and offering riders special promotions, for example:

- Discounted fares
- Revenue from marked bikes is donated to charity

A more elaborate approach could involve awarding riders who participate in the achievement of some goal, for example:

- Completing a geocache course. This could be co-sponsored by businesses along the geocache route, or with healthcare providers to encourage fitness.
- Completing a geo-design. This could be leveraged to support social-responsibility awareness campaigns.

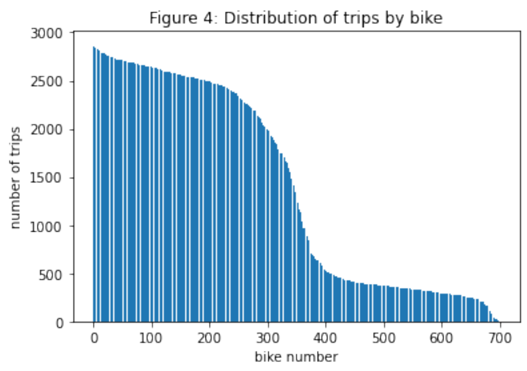
From an operational perspective, such a program could be used to redistribute load across more bikes so that maintenance costs are smoothed and reduced across the overall inventory of bikes.

Examining raw count of trips per bike

```
In [46]: %%bigquery df_bike_data --use_rest_api
             SELECT bike_number, count(trip_id) as trip_count
             FROM `bigquery-public-data.san_francisco.bikeshare_trips`
             GROUP BY bike_number
             ORDER BY trip_count DESC
```

```
/opt/conda/lib/python3.7/site-packages/google/cloud/bigquery/client.py:407: UserWarning: Cannot create BigQuery Storage
client, the dependency google-cloud-bigquery-storage is not installed.
  "Cannot create BigQuery Storage client, the dependency "
```

```
In [47]: fig4, ax4 = plt.subplots()
         x4 = df_bike_data.index
         y4 = df_bike_data['trip_count']
         ax4.bar(x4, y4)
         ax4.set(xlabel='bike number', ylabel='number of trips',
                title='Figure 4: Distribution of trips by bike')

         plt.show()
```



From Figure 4 we can see that roughly 50% of bikes are heavily used while the remaining 50% are lightly used. There is a significant transition at 50%.

How are the bikes distributed across landmarks?

```
In [48]: %%bigquery df_landmark_bikes --use_rest_api
            SELECT B.landmark, A.bike_number, count(A.trip_id) as trip_count
             FROM `bigquery-public-data.san_francisco.bikeshare_trips` as A
             INNER JOIN `bigquery-public-data.san_francisco.bikeshare_stations` as B
             ON A.start_station_id = B.station_id
             GROUP BY B.landmark, A.bike_number
             ORDER BY B.landmark, trip_count DESC
```

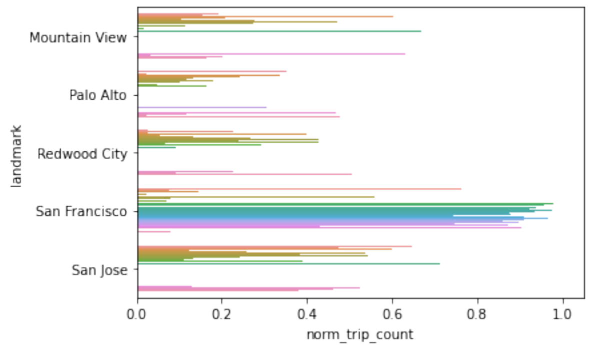
```
In [49]: df_landmark_bikes.head()
```

Out[49]:

	landmark	bike_number	trip_count
0	Mountain View	141	317
1	Mountain View	13	256
2	Mountain View	263	225
3	Mountain View	296	212
4	Mountain View	203	207

```
In [50]: df_plot5 = df_landmark_bikes
         dist =  df_plot5.groupby('landmark').transform('max')
         df_plot5['norm_trip_count'] = df_plot5['trip_count'].div(dist['trip_count'])
         df_plot5.sort_values(['landmark', 'trip_count'], ascending=[True, False], inplace=True)
```

```
In [54]: import seaborn as sns
         plt.figure()
         g = sns.barplot(x="norm_trip_count", y="landmark", hue='bike_number', data=df_plot5)
         g.legend_.remove()
```



Analysis

While Figure 5 above is somewhat difficult to read, it indicates that

- In San Francisco, the majority of bikes are utilized at relatively the same frequency
- In other locations, there are a few bikes in each location that are more heavily utilized

In San Francisco, using a gamification approach may generate increased ridership through novelty and/or good will for public benefit programs.

Outside of San Francisco, there could be some benefits from attempting to distribute the utilization load across more bikes in those locations.

Recommendation

In San Francisco:

- Test the concept of a gamification program by dedicating a small set of bikes (randomly chosen from high-usage and low-usage bikes) and possibly donating the revenue to charity.
- Measure the effectiveness of the program to determine its impact on increasing ridership.

For other locations:

- Examine maintenance cost data to determine if a business case exists for implementing an approach to redistribute load across more bikes
- Consider piloting a gamification program after concept testing in San Francisco

Conclusion

This analysis examined several questions related to how to increase ridership for Lyft.

Summary recommendations

- Bike Availability
  - Create a detailed inventory of stations with higher-than-average availability.
  - Develop targeted marketing initiatives to attract new riders.
- High Utilization
  - Continue monitoring utilization growth, improving surge demand forecasting.
  - Explore flexible surge management solutions to satisfy latent demand.
- Service Area Extension
  - There are several underserved customer zones in the current Lyft service area and surrounding region.
  - Develop business cases for the areas identified in this analysis.
- Gamification
  - Opportunities exist for increasing ridership and redistributing workload across bikes.
  - Conduct a proof-of-concept to determine feasability of the program.

Next steps

- Setup meetings with key stakeholder groups to discuss the key findings and recommendations of this analysis
- New service areas:
  - Continue analysis of extending service into new areas to support business case development.
  - Launch survey initiative to collect data on level of interest in target areas.

```
In [ ]:
```