Udacity Project 5: Communicate Data Findings

Analyzing Ford GoBike System Data

by Souvik Das

Preliminary Wrangling

> Ford GoBike(currently Bay Wheels) is a regional public bicycle sharing system in the San Francisco Bay Area, California. Beginning operation in August 2013 as Bay Area Bike Share, the Ford GoBike system currently has over 2,600 bicycles in 262 stations across San Francisco, East Bay and San Jose. On June 28, 2017, the system officially launched as Ford GoBike in a partnership with Ford Motor Company.After Motivate's acquisition by Lyft, the system was subsequently renamed to Bay Wheels in June 2019. The system is expected to expand to 7,000 bicycles around 540 stations in San Francisco, Oakland, Berkeley, Emeryville, and San Jose.

> Ford GoBike, like other bike share systems, consists of a fleet of specially designed, sturdy and durable bikes that are locked into a network of docking stations throughout the city. The bikes can be unlocked from one station and returned to any other station in the system, making them ideal for one-way trips. The bikes are available for use 24 hours/day, 7 days/week, 365 days/year and riders have access to all bikes in the network when they become a member or purchase a pass.

> On June 2019, the company was renamed as Bay Wheels.For the sake of the analysis data upto April 2019 has been taken.The data provided after that misses key columns like cyclist's age and gender.

In [1]:
```
# import all packages and set plots to be embedded inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import requests
import seaborn as sb
from io import BytesIO
from zipfile import ZipFile
import os

%matplotlib inline

plt.rcParams["figure.figsize"] = (10,8)
SMALL_SIZE = 12
MEDIUM_SIZE = 16
BIGGER_SIZE = 18

plt.rc('font', size=SMALL_SIZE)          # controls default text sizes
plt.rc('axes', titlesize=BIGGER_SIZE)     # fontsize of the axes title
plt.rc('axes', labelsize=MEDIUM_SIZE)    # fontsize of the x and y labels
plt.rc('xtick', labelsize=SMALL_SIZE)    # fontsize of the tick labels
plt.rc('ytick', labelsize=SMALL_SIZE)    # fontsize of the tick labels
plt.rc('legend', fontsize=SMALL_SIZE)    # legend fontsize
```

In [2]:
```
folder_name_csvs = 'trip_data_files'
if not os.path.exists(folder_name_csvs):
    os.makedirs(folder_name_csvs)
```

In [ ]:
```
# Donwload all the data files and unzip the them
urls = ['https://s3.amazonaws.com/fordgobike-data/201801-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201802-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201803-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201804-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201805-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201806-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201807-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201808-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201809-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201810-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201811-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201812-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201901-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201902-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201903-fordgobike-tripdata.csv.zip',
        'https://s3.amazonaws.com/fordgobike-data/201904-fordgobike-tripdata.csv.zip',]

for url in urls:
    response = requests.get(url)
    zip_file = ZipFile(BytesIO(response.content))
    zip_file.extractall(folder_name_csvs)
```

In [4]:
```python
# Read all files except one and append into a list
list_dfs = []

for file in sorted(os.listdir(folder_name_csvs)):
    print('Append file:', file)
    list_dfs.append(pd.read_csv(folder_name_csvs + '/' + file))
```

```
Append file: 201801-fordgobike-tripdata.csv
Append file: 201802-fordgobike-tripdata.csv
Append file: 201803-fordgobike-tripdata.csv
Append file: 201804-fordgobike-tripdata.csv
Append file: 201805-fordgobike-tripdata.csv
Append file: 201806-fordgobike-tripdata.csv
Append file: 201807-fordgobike-tripdata.csv
Append file: 201808-fordgobike-tripdata.csv
Append file: 201809-fordgobike-tripdata.csv
Append file: 201810-fordgobike-tripdata.csv
Append file: 201811-fordgobike-tripdata.csv
Append file: 201812-fordgobike-tripdata.csv
Append file: 201901-fordgobike-tripdata.csv
Append file: 201902-fordgobike-tripdata.csv
Append file: 201903-fordgobike-tripdata.csv
Append file: 201904-fordgobike-tripdata.csv
```

In [5]:
```python
# Concatenate all files into a single dataframe
df = pd.concat(list_dfs, sort=False)
print('Shape df:', df.shape)
```

```
Shape df: (2734625, 16)
```

In [6]:
```python
# Saving the combined dataframe into a new file to work with
df.to_csv("bikes_dataset_combined.csv", index = False)
```

Assess

In [7]:
```python
bikes_df = pd.read_csv('bikes_dataset_combined.csv')
print(bikes_df.shape)
```

```
(2734625, 16)
```

In [8]:
```python
bikes_df
```

Out[8]:

	duration_sec	start_time	end_time	start_station_id	start_station_name	start_station_latitude	start_station_longitude	end_station_id	e
0	75284	2018-01-31 22:52:35.2390	2018-02-01 19:47:19.8240	120.0	Mission Dolores Park	37.761420	-122.426435	285.0	
1	85422	2018-01-31 16:13:34.3510	2018-02-01 15:57:17.3100	15.0	San Francisco Ferry Building (Harry Bridges Pl...	37.795392	-122.394203	15.0	
2	71576	2018-01-31 14:23:55.8890	2018-02-01 10:16:52.1160	304.0	Jackson St at 5th St	37.348759	-121.894798	296.0	
3	61076	2018-01-31 14:53:23.5620	2018-02-01 07:51:20.5000	75.0	Market St at Franklin St	37.773793	-122.421239	47.0	
4	39966	2018-01-31 19:52:24.6670	2018-02-01 06:58:31.0530	74.0	Laguna St at Hayes St	37.776435	-122.426244	19.0	
...	...	...	...	...	...	...	...	...	
2734620	184	2019-04-01 00:09:17.5660	2019-04-01 00:12:22.5170	133.0	Valencia St at 22nd St	37.755213	-122.420975	132.0	
2734621	539	2019-04-01 00:03:02.5730	2019-04-01 00:12:02.0670	78.0	Folsom St at 9th St	37.773717	-122.411647	77.0	
2734622	292	2019-04-01 00:06:04.2370	2019-04-01 00:10:56.9850	243.0	Bancroft Way at College Ave	37.869360	-122.254337	269.0	
2734623	471	2019-04-01 00:01:38.4110	2019-04-01 00:09:29.9650	370.0	Jones St at Post St	37.787327	-122.413278	43.0	
2734624	356	2019-04-01 00:00:28.7290	2019-04-01 00:06:25.0650	14.0	Clay St at Battery St	37.795001	-122.399970	371.0	

2734625 rows × 16 columns

In [9]:
```python
bikes_df.sample(10)
```

Out[9]:

	duration_sec	start_time	end_time	start_station_id	start_station_name	start_station_latitude	start_station_longitude	end_station_id
234966	172	2018-03-23 18:29:25.4300	2018-03-23 18:32:17.4400	112.0	Harrison St at 17th St	37.763847	-122.413004	108.0
1998910	287	2019-01-11 17:12:42.6630	2019-01-11 17:17:29.9310	171.0	Rockridge BART Station	37.844279	-122.251900	207.0
1962431	1228	2019-01-18 17:41:45.3160	2019-01-18 18:02:14.2620	49.0	S Park St at 3rd St	37.780760	-122.394989	120.0
1278376	578	2018-09-20 08:50:46.2360	2018-09-20 09:00:24.3390	357.0	2nd St at Julian St	37.341132	-121.892844	327.0
1149955	4506	2018-08-10 10:07:03.0870	2018-08-10 11:22:09.4140	49.0	S Park St at 3rd St	37.780760	-122.394989	356.0
534982	500	2018-05-16 07:49:32.2190	2018-05-16 07:57:52.4030	277.0	Morrison Ave at Julian St	37.333658	-121.908586	313.0
2083752	423	2019-02-25 09:06:25.3660	2019-02-25 09:13:29.0250	67.0	San Francisco Caltrain Station 2 (Townsend St...	37.776639	-122.395526	363.0
1311526	911	2018-09-14 18:41:45.6370	2018-09-14 18:56:56.8300	42.0	San Francisco City Hall (Polk St at Grove St)	37.778650	-122.418230	134.0
684322	463	2018-06-21 19:55:04.4570	2018-06-21 20:02:48.4560	245.0	Downtown Berkeley BART	37.870348	-122.267764	256.0
659967	536	2018-06-25 19:30:38.1160	2018-06-25 19:39:34.2630	3.0	Powell St BART Station (Market St at 4th St)	37.786375	-122.404904	24.0

Programmatic assessment

In [10]:
```python
bikes_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2734625 entries, 0 to 2734624
Data columns (total 16 columns):
 #   Column                   Dtype  
---  ------                   -----  
 0   duration_sec             int64  
 1   start_time               object 
 2   end_time                 object 
 3   start_station_id         float64
 4   start_station_name       object 
 5   start_station_latitude   float64
 6   start_station_longitude  float64
 7   end_station_id           float64
 8   end_station_name         object 
 9   end_station_latitude     float64
 10  end_station_longitude    float64
 11  bike_id                  int64  
 12  user_type                object 
 13  member_birth_year        float64
 14  member_gender            object 
 15  bike_share_for_all_trip  object 
dtypes: float64(7), int64(2), object(7)
memory usage: 333.8+ MB
```

In [11]:
```python
bikes_df.isnull().sum()
```

Out[11]:
```
duration_sec                    0
start_time                      0
end_time                        0
start_station_id            12501
start_station_name          12501
start_station_latitude          0
start_station_longitude         0
end_station_id              12501
end_station_name            12501
end_station_latitude            0
end_station_longitude           0
bike_id                         0
user_type                       0
member_birth_year          151625
member_gender              151271
bike_share_for_all_trip         0
dtype: int64
```

In [12]:
```
bikes_df.duplicated().sum()
```
Out[12]:
```
0
```

In [13]:
```
bikes_df.member_birth_year.value_counts()
```
Out[13]:
```
1988.0    151210
1989.0    129011
1987.0    126198
1990.0    124272
1993.0    120640
           ...
1906.0         2
1930.0         2
1903.0         1
1886.0         1
1910.0         1
Name: member_birth_year, Length: 92, dtype: int64
```

In [14]:
```
bikes_df[bikes_df.member_birth_year < 1920]
```
Out[14]:

	duration_sec	start_time	end_time	start_station_id	start_station_name	start_station_latitude	start_station_longitude	end_station_id
93	465	2018-01-31 21:56:17.6330	2018-01-31 22:04:02.8280	3.0	Powell St BART Station (Market St at 4th St)	37.786375	-122.404904	60.0
1065	549	2018-01-31 18:01:24.7290	2018-01-31 18:10:34.2680	78.0	Folsom St at 9th St	37.773717	-122.411647	5.0
1254	568	2018-01-31 17:40:47.7010	2018-01-31 17:50:16.5020	223.0	16th St Mission BART Station 2	37.764765	-122.420091	60.0
1532	658	2018-01-31 17:11:03.4240	2018-01-31 17:22:01.9530	19.0	Post St at Kearny St	37.788975	-122.403452	30.0
2983	1681	2018-01-31 09:37:00.0450	2018-01-31 10:05:01.8970	6.0	The Embarcadero at Sansome St	37.804770	-122.403234	98.0
...	...	...	...	...	...	...	...	...
2724602	363	2019-04-02 06:21:33.0400	2019-04-02 06:27:36.6320	136.0	23rd St at San Bruno Ave	37.754436	-122.404364	134.0
2725996	847	2019-04-01 19:21:18.9910	2019-04-01 19:35:26.0960	141.0	Valencia St at Cesar Chavez St	37.747998	-122.420219	136.0
2731170	400	2019-04-01 10:17:47.5730	2019-04-01 10:24:28.5320	67.0	San Francisco Caltrain Station 2 (Townsend St...	37.776639	-122.395526	26.0
2731480	1083	2019-04-01 09:34:37.6300	2019-04-01 09:52:40.8930	375.0	Grove St at Masonic Ave	37.774836	-122.446546	36.0
2734440	385	2019-04-01 06:18:23.9770	2019-04-01 06:24:49.3760	136.0	23rd St at San Bruno Ave	37.754436	-122.404364	134.0

1529 rows × 16 columns

Clean

In [15]:
```
bikes_df_clean = bikes_df.copy()
```

In [16]:
```
bikes_df_clean['start_time'] = pd.to_datetime(bikes_df_clean['start_time'])
bikes_df_clean['end_time'] = pd.to_datetime(bikes_df_clean['end_time'])
```

In [17]:
```
bikes_df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2734625 entries, 0 to 2734624
Data columns (total 16 columns):
 #   Column                   Dtype
---  ------                   -----
 0   duration_sec             int64
 1   start_time               datetime64[ns]
 2   end_time                 datetime64[ns]
 3   start_station_id         float64
 4   start_station_name       object
 5   start_station_latitude   float64
 6   start_station_longitude  float64
 7   end_station_id           float64
 8   end_station_name         object
 9   end_station_latitude     float64
 10  end_station_longitude    float64
 11  bike_id                  int64
 12  user_type                object
 13  member_birth_year        float64
 14  member_gender            object
 15  bike_share_for_all_trip  object
dtypes: datetime64[ns](2), float64(7), int64(2), object(5)
memory usage: 333.8+ MB
```

Outliers should be removed

> Cyclists with age greater than 100 are probably typing mistakes,hence they are removed.(Person with birth year 1998 typed as 1899 etc.)

In [18]:
```
print('Values < 1920:', (bikes_df_clean.member_birth_year < 1920).sum())
print('Rows before:', bikes_df_clean.shape[0])
bikes_df_clean['member_birth_year'] = bikes_df_clean.member_birth_year.apply(lambda x: int('19'+str(int(x))[-2:]) if x < 1
bikes_df_clean = bikes_df_clean[bikes_df_clean.member_birth_year >= 1920]
```

```
Values < 1920: 1529
Rows before: 2734625
```

In [19]:
```
print('Rows after:', bikes_df_clean.shape[0])
```

```
Rows after: 2581545
```

In [20]:
```
bikes_df_clean['member_age'] = 2020 - bikes_df_clean['member_birth_year']
```

In [21]:
```
bikes_df_clean['member_age'].sample(5)
```

Out[21]:
```
2293169    36.0
1766259    24.0
267048     45.0
2351224    43.0
816291     48.0
Name: member_age, dtype: float64
```

In [22]:
```
age_bins = [0, 19, 29, 39, 49, 59,
            69, 79, 89, 99]
age_labels = ['10 - 19', '20 - 29', '30 - 39', '40 - 49', '50 - 59',
              '60 - 69', '70 - 79', '80 - 89', '90 - 99']

bikes_df_clean['age_group'] = pd.cut(bikes_df_clean['member_age'], bins = age_bins, labels = age_labels, right = False)
```

In [23]:
```
bikes_df_clean[['member_age', 'age_group']].sample(10)
```

Out[23]:

	member_age	age_group
390116	54.0	50 - 59
105208	29.0	30 - 39
1947085	33.0	30 - 39
1771094	22.0	20 - 29
456430	26.0	20 - 29
753009	49.0	50 - 59
2311171	30.0	30 - 39
947896	44.0	40 - 49
134121	27.0	20 - 29
1737812	42.0	40 - 49

In [24]:
```
bikes_df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2581545 entries, 0 to 2734624
Data columns (total 18 columns):
 #   Column                   Dtype
---  ------                   -----
 0   duration_sec             int64
 1   start_time               datetime64[ns]
 2   end_time                 datetime64[ns]
 3   start_station_id         float64
 4   start_station_name       object
 5   start_station_latitude   float64
 6   start_station_longitude  float64
 7   end_station_id           float64
 8   end_station_name         object
 9   end_station_latitude     float64
 10  end_station_longitude    float64
 11  bike_id                  int64
 12  user_type                object
 13  member_birth_year        float64
 14  member_gender            object
 15  bike_share_for_all_trip  object
 16  member_age               float64
 17  age_group                category
dtypes: category(1), datetime64[ns](2), float64(8), int64(2), object(5)
memory usage: 357.0+ MB
```

In [25]:
```
bikes_df_clean.age_group.value_counts()
```

Out[25]:
```
30 - 39    1156569
20 - 29     585197
40 - 49     494733
50 - 59     244822
60 - 69      84264
70 - 79      13875
80 - 89       1243
90 - 99        726
10 - 19          0
Name: age_group, dtype: int64
```

Create new column

Define

In [26]:
```
bikes_df_clean['month_year'] = bikes_df_clean['start_time'].dt.to_period('M')
```

Test

In [27]:
```
bikes_df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2581545 entries, 0 to 2734624
Data columns (total 19 columns):
 #   Column                   Dtype
---  ------                   -----
 0   duration_sec             int64
 1   start_time               datetime64[ns]
 2   end_time                 datetime64[ns]
 3   start_station_id         float64
 4   start_station_name       object
 5   start_station_latitude   float64
 6   start_station_longitude  float64
 7   end_station_id           float64
 8   end_station_name         object
 9   end_station_latitude     float64
 10  end_station_longitude    float64
 11  bike_id                  int64
 12  user_type                object
 13  member_birth_year        float64
 14  member_gender            object
 15  bike_share_for_all_trip  object
 16  member_age               float64
 17  age_group                category
 18  month_year               period[M]
dtypes: category(1), datetime64[ns](2), float64(8), int64(2), object(5), period[M](1)
memory usage: 376.7+ MB
```

In [28]:
```
bikes_df_clean.month_year.value_counts()
```

Out[28]:
```
2019-03    244471
2019-04    227849
2018-10    192736
2018-07    186721
2018-06    183262
2019-01    182275
2018-08    181150
2018-09    176189
2019-02    175076
2018-05    167270
2018-11    128991
2018-12    126287
2018-04    121677
2018-03    102246
2018-02     98498
2018-01     86847
Freq: M, Name: month_year, dtype: int64
```

Remove unnecessary columns

Define

In [29]:
```
bikes_df_clean = bikes_df_clean.drop(columns=['start_station_id', 'end_station_id', 'start_station_latitude', 'start_stati
```

Test

In [30]:
```
bikes_df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2581545 entries, 0 to 2734624
Data columns (total 13 columns):
 #   Column                   Dtype
---  ------                   -----
 0   duration_sec             int64
 1   start_time               datetime64[ns]
 2   end_time                 datetime64[ns]
 3   start_station_name       object
 4   end_station_name         object
 5   bike_id                  int64
 6   user_type                object
 7   member_birth_year        float64
 8   member_gender            object
 9   bike_share_for_all_trip  object
 10  member_age               float64
 11  age_group                category
 12  month_year               period[M]
dtypes: category(1), datetime64[ns](2), float64(2), int64(2), object(5), period[M](1)
memory usage: 258.5+ MB
```

Saving the clean dataset

In [31]:
```
bikes_df_clean.to_csv('ford_gobike_cleaned.csv', index=False)
```

In [32]:
```
master_df = pd.read_csv('ford_gobike_cleaned.csv')
master_df.tail(10)
```

Out[32]:

	duration_sec	start_time	end_time	start_station_name	end_station_name	bike_id	user_type	member_birth_year	member_gender	bike
2581535	197	2019-04-01 00:23:21.039	2019-04-01 00:26:38.384	Parker St at Fulton St	Channing Way at Shattuck Ave	6425	Subscriber	1998.0	Male	
2581536	914	2019-04-01 00:11:07.612	2019-04-01 00:26:21.707	Guerrero Park	Bryant St at 6th St	5487	Subscriber	1991.0	Male	
2581537	869	2019-04-01 00:08:19.001	2019-04-01 00:22:48.864	Berry St at 4th St	Berry St at 4th St	5910	Customer	1997.0	Male	
2581538	396	2019-04-01 00:14:37.960	2019-04-01 00:21:14.402	San Francisco Public Library (Grove St at Hyde...	Turk St at Fillmore St	6448	Subscriber	1986.0	Male	
2581539	421	2019-04-01 00:11:05.276	2019-04-01 00:18:06.822	San Fernando St at 7th St	Ryland Park	6274	Subscriber	1992.0	Male	
2581540	184	2019-04-01 00:09:17.566	2019-04-01 00:12:22.517	Valencia St at 22nd St	24th St at Chattanooga St	6430	Subscriber	1976.0	Male	
2581541	539	2019-04-01 00:03:02.573	2019-04-01 00:12:02.067	Folsom St at 9th St	11th St at Natoma St	4972	Subscriber	1981.0	Male	
2581542	292	2019-04-01 00:06:04.237	2019-04-01 00:10:56.985	Bancroft Way at College Ave	Telegraph Ave at Carleton St	3415	Subscriber	1997.0	Male	
2581543	471	2019-04-01 00:01:38.411	2019-04-01 00:09:29.965	Jones St at Post St	San Francisco Public Library (Grove St at Hyde...	5018	Subscriber	1996.0	Female	
2581544	356	2019-04-01 00:00:28.729	2019-04-01 00:06:25.065	Clay St at Battery St	Lombard St at Columbus Ave	5956	Subscriber	1970.0	Male	

In [33]:
```
master_df.isnull().sum()
```

Out[33]:
```
duration_sec                0
start_time                  0
end_time                    0
start_station_name      12167
end_station_name        12167
bike_id                     0
user_type                   0
member_birth_year           0
member_gender               0
bike_share_for_all_trip     0
member_age                  0
age_group                 116
month_year                  0
dtype: int64
```

In [34]:
```
#Adding additional columns to work with weekdays
master_df['start_time'] = pd.to_datetime(master_df.start_time)
master_df['end_time'] = pd.to_datetime(master_df.end_time)
master_df['minutes'] = master_df.duration_sec / 60.

daymap = {0:'Monday',1:'Tuesday',2:'Wednesday',3:'Thursday',4:'Friday',5:'Saturday',6:'Sunday'}
master_df['day'] = master_df.start_time.apply(lambda time: time.dayofweek).map(daymap)

monmap = {1: 'January', 2: 'February',3: 'March',4: 'April',5: 'May',6: 'June',7: 'July',8: 'August',9: 'September',10: 'O
master_df['month'] = master_df.start_time.apply(lambda time: time.month).map(monmap)

master_df['month_num'] = master_df.start_time.apply(lambda time: time.month)
master_df['day_num'] = master_df.start_time.apply(lambda time: time.dayofweek)


master_df['hour'] = master_df.start_time.dt.hour

master_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2581545 entries, 0 to 2581544
Data columns (total 19 columns):
 #   Column                   Dtype
---  ------                   -----
 0   duration_sec             int64
 1   start_time               datetime64[ns]
 2   end_time                 datetime64[ns]
 3   start_station_name       object
 4   end_station_name         object
 5   bike_id                  int64
 6   user_type                object
 7   member_birth_year        float64
 8   member_gender            object
 9   bike_share_for_all_trip  object
 10  member_age               float64
 11  age_group                object
 12  month_year               object
 13  minutes                  float64
 14  day                      object
 15  month                    object
 16  month_num                int64
 17  day_num                  int64
 18  hour                     int64
dtypes: datetime64[ns](2), float64(3), int64(5), object(9)
memory usage: 374.2+ MB
```

In [35]:
```
master_df.sample(10)
```

Out[35]:

	duration_sec	start_time	end_time	start_station_name	end_station_name	bike_id	user_type	member_birth_year	member_gender	bike
394510	1046	2018-04-04 16:09:35.230	2018-04-04 16:27:02.062	Commercial St at Montgomery St	3rd St at Townsend St	190	Customer	1983.0	Male	
900349	911	2018-07-09 19:57:41.546	2018-07-09 20:12:52.742	Steuart St at Market St	Laguna St at Hayes St	713	Subscriber	1991.0	Male	
2530679	413	2019-04-06 14:45:18.634	2019-04-06 14:52:12.256	Broadway at Kearny	San Francisco Ferry Building (Harry Bridges Pl...	5257	Subscriber	1990.0	Male	
1043923	1298	2018-08-15 09:02:52.110	2018-08-15 09:24:30.851	Central Ave at Fell St	Montgomery St BART Station (Market St at 2nd St)	3240	Subscriber	1968.0	Female	
807105	930	2018-07-24 12:19:15.677	2018-07-24 12:34:46.257	MacArthur BART Station	MacArthur BART Station	211	Customer	1994.0	Male	
1175245	454	2018-09-23 06:12:22.274	2018-09-23 06:19:56.892	14th St at Filbert St	West Oakland BART Station	13	Subscriber	1982.0	Female	
1171394	163	2018-09-24 07:38:50.465	2018-09-24 07:41:33.728	Telegraph Ave at 27th St	Telegraph Ave at 19th St	711	Subscriber	1988.0	Male	
706561	997	2018-06-10 10:53:01.841	2018-06-10 11:09:39.665	Folsom St at 3rd St	Washington St at Kearny St	3900	Subscriber	1983.0	Male	
1684444	295	2018-12-15 11:26:22.307	2018-12-15 11:31:18.083	Lombard St at Columbus Ave	Broadway at Kearny	5441	Subscriber	1976.0	Other	
731626	649	2018-06-05 19:29:51.547	2018-06-05 19:40:41.288	San Francisco Caltrain (Townsend St at 4th St)	Broadway at Battery St	533	Subscriber	1989.0	Male	

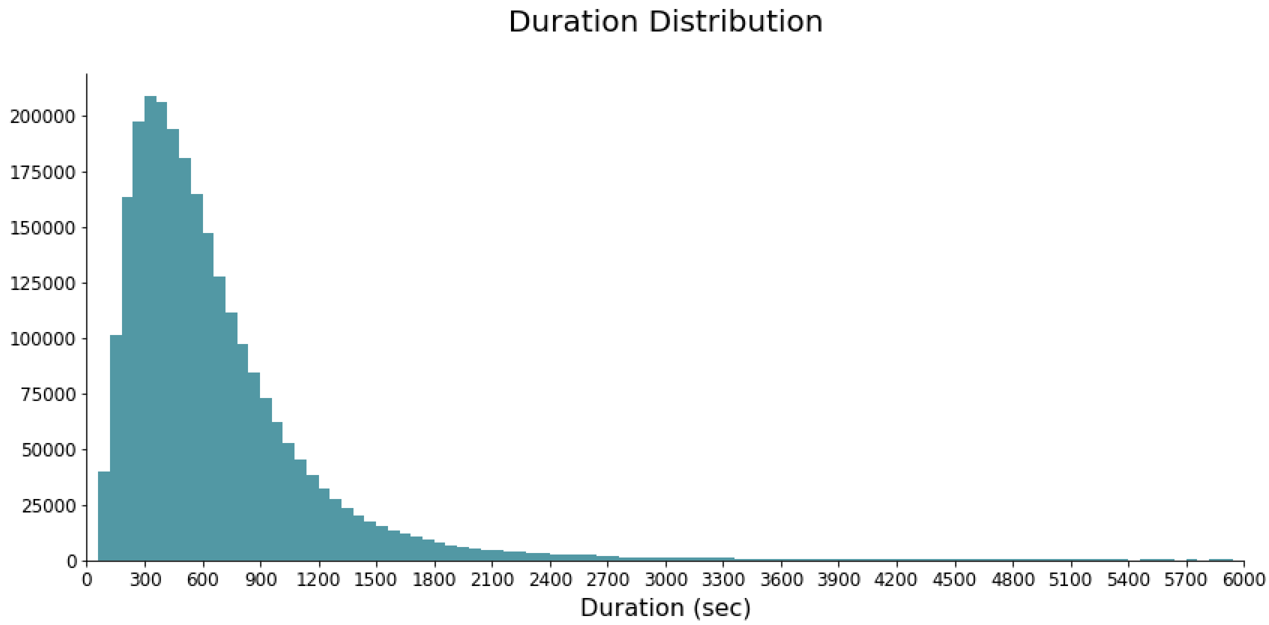
Univariate exploration

- **duration_sec**

In [36]:
```
# Set bin size and color
bin_size = 60
bins = np.arange(0, master_df.duration_sec.max()+bin_size, bin_size)
color = sb.color_palette('viridis')[2]

# Plotting
fig, axes = plt.subplots(figsize = (12,6))
plt.hist(master_df.duration_sec, bins = bins, color= color, alpha=0.8);

# Aesthetic wrangling
plt.xticks(ticks = [x for x in range(0,6001,300)])
plt.title('Duration Distribution\n', size=20)
plt.xlabel('Duration (sec)')
plt.xlim(0,6000)
sb.despine(fig)
plt.tight_layout();
```

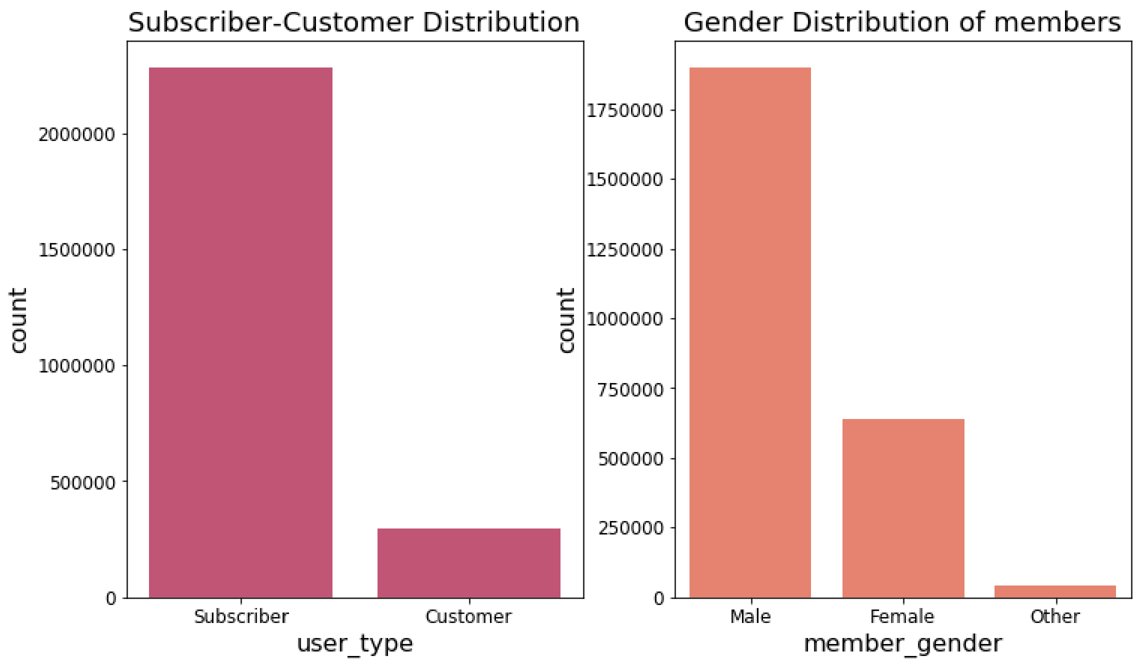


- We can observe that this distribution is highly skewed to right. Most of the trips have the duration between 300 to 600 seconds, that is, 5 to 10 minutes.

- **user_type**
- **member_gender**

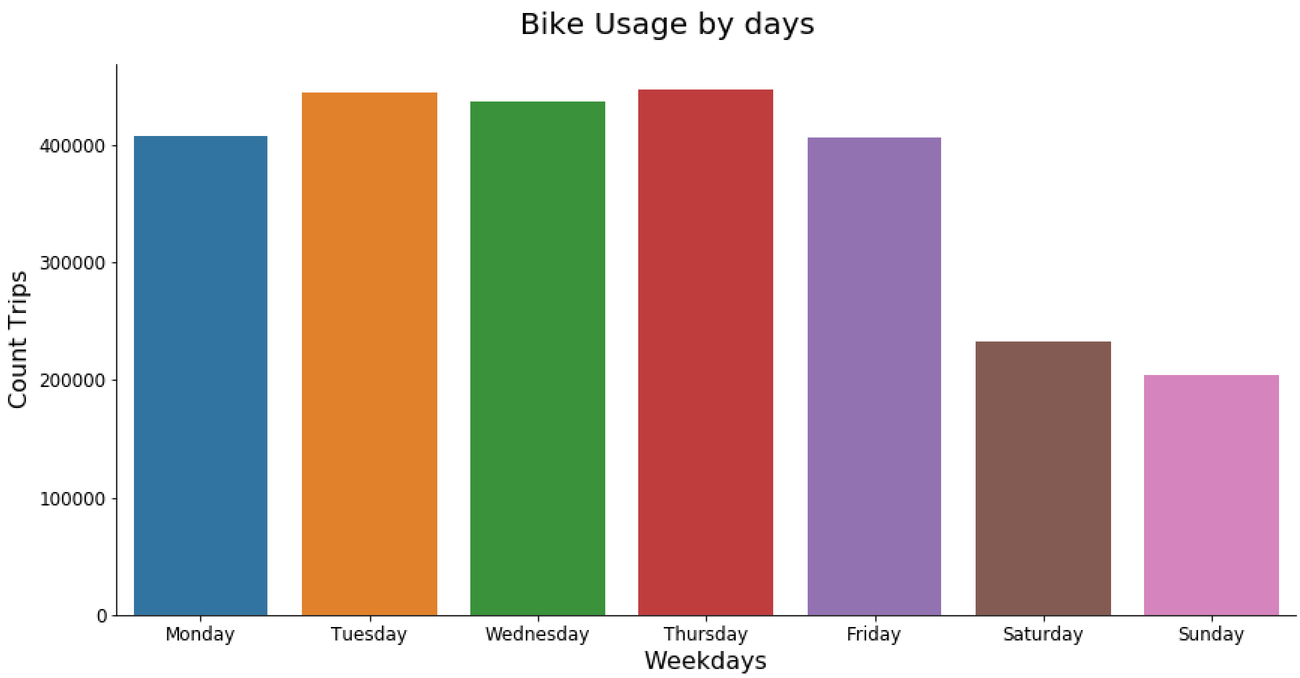
In [37]:
```
fig = plt.figure(figsize=(12,15))
ax1 = fig.add_subplot(221)
ax1.title.set_text('Subscriber-Customer Distribution')
sb.countplot(master_df['user_type'], color=sb.color_palette('magma')[3])


ax2 = fig.add_subplot(222)
ax2.title.set_text('Gender Distribution of members')
sb.countplot(master_df['member_gender'], color=sb.color_palette('magma')[4]);
```



- There are much more subscriber than customer.
- There are much more male than the others genders.

In [38]:
```
# creating order for days of the week
dow_order = ['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']
# actual plot
ax = sb.catplot(data=master_df, x='day', kind='count', order = dow_order,height=6,aspect=2)
ax.fig.suptitle('Bike Usage by days', y=1.05, fontsize=20, fontweight='normal');
ax.set_axis_labels('Weekdays', 'Count Trips')
ax.set_xticklabels(rotation=0);
```

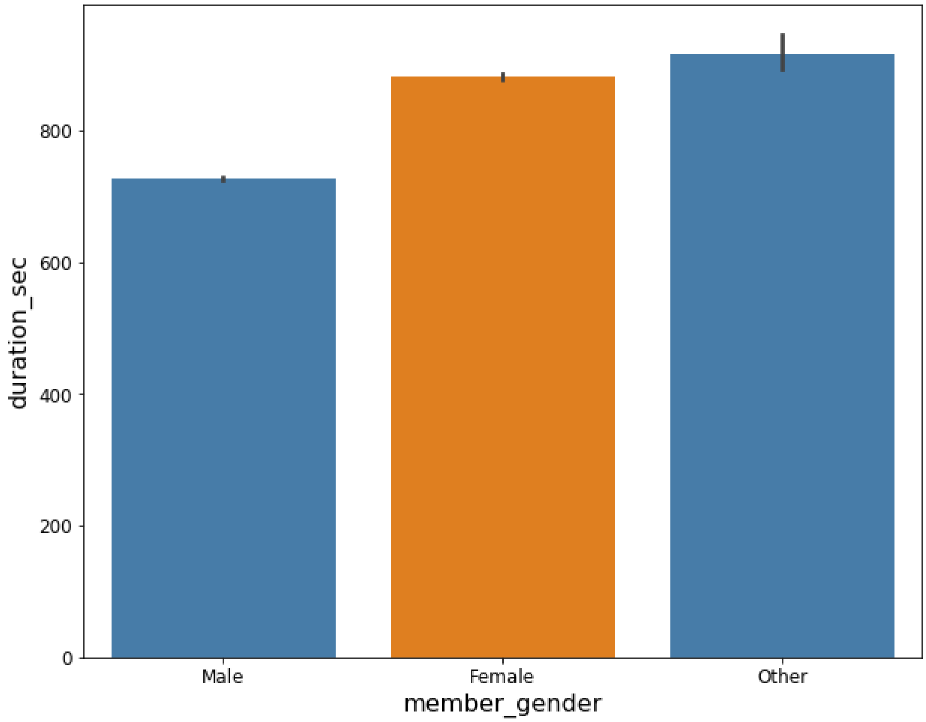


- Number of rides are less on weekend compared to weekdays

Bivariate Exploration

- **member_gender x duration_sec**

In [39]:
```
plt.figure(figsize=(10, 8))
sb.barplot(data=master_df, x='member_gender', y='duration_sec', palette=['#377eb8', '#ff7f00']);
```

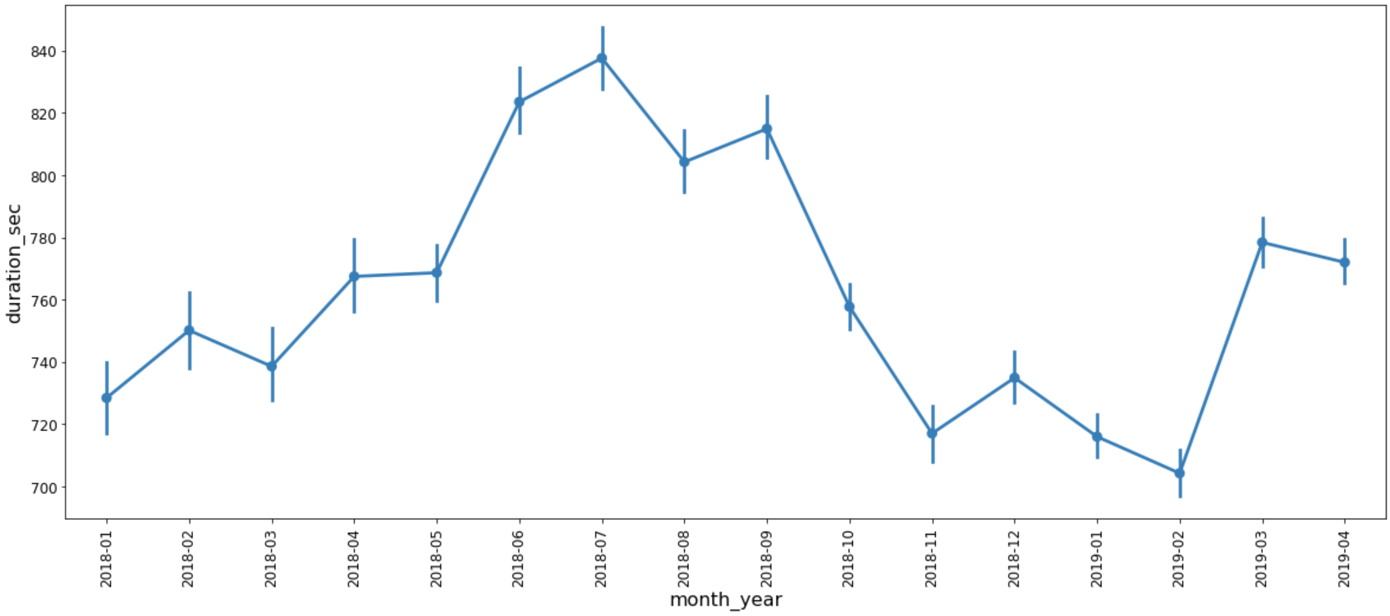


- We can observe that duration of the trips in each gender are very different. In average, the duration of the trips in the male gender are lower in terms of seconds (about 730 sec, that is, 14 minutes aproximattely). In contrast, female gender use the trips for more time (about 875 sec, that is, 15 minutes aproximattely). Finally, others genders are about (875-925 sec, that is, 15-16 minutes).

- **month_year x duration_sec**

In [40]:
```
plt.figure(figsize=(20, 8))

sb.pointplot(data=master_df, x='month_year', y='duration_sec', color='#377eb8')
plt.xticks(rotation=90);
```



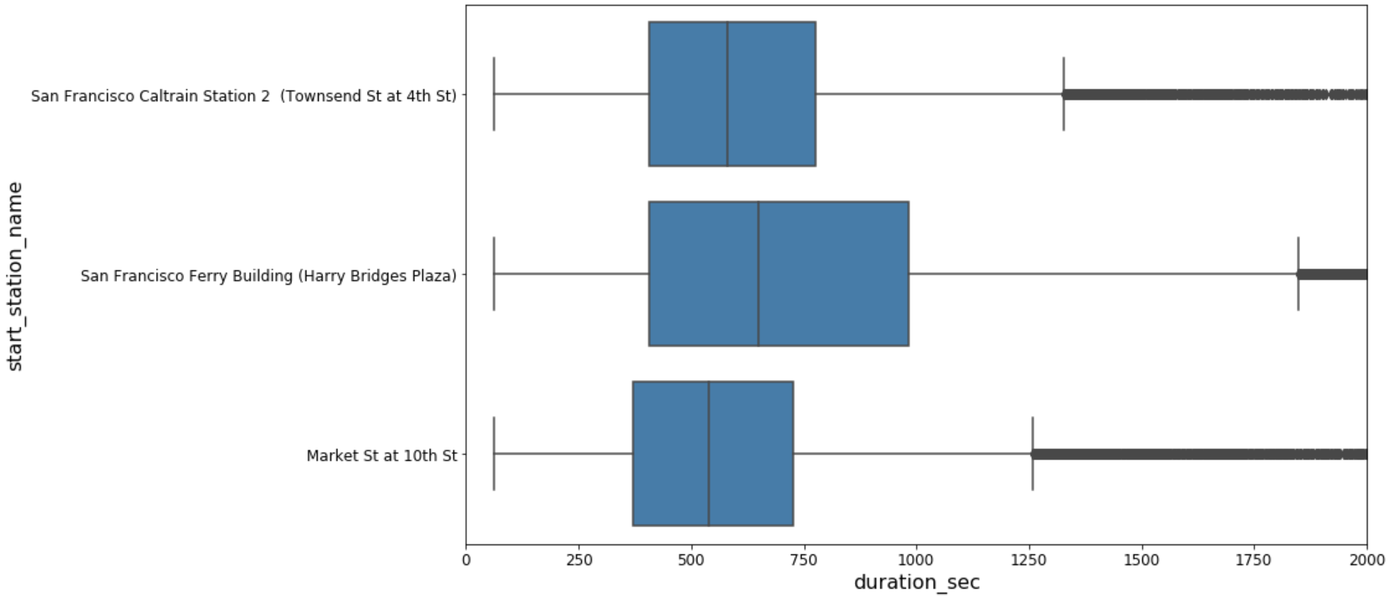
- The months who have more duration in the trips are June, July (Highest), August and September.
- The months who have lower duration is the trips is November, December, January and Frebruary (Lowest).

- **duration_sec x start_station_name (Top 3)**

In [41]:
```
# Top 3 stations
top_3_stations = master_df['start_station_name'].value_counts().index[:3]
master_df_top_3_stations = master_df.loc[master_df['start_station_name'].isin(top_3_stations)]

plt.figure(figsize = [13, 8])

sb.boxplot(data = master_df_top_3_stations, y = 'start_station_name', x = 'duration_sec', color = '#377eb8', orient="h")
plt.xlim(0, 2000);
```



- Here we can observe the top 5 stations that users use to start a trip. We can observe the users who start in the station San Francisco Ferry Building (Harry Bridges Plaza) have more duration in the trip, the median of duration is about 650 seconds, that is, about 11 minutes and the Q3 quartile is pretty higher. The other other stations have basically the same results, the median is about 520 seconds, that is, about 9 minutes.

Multivariate Exploration

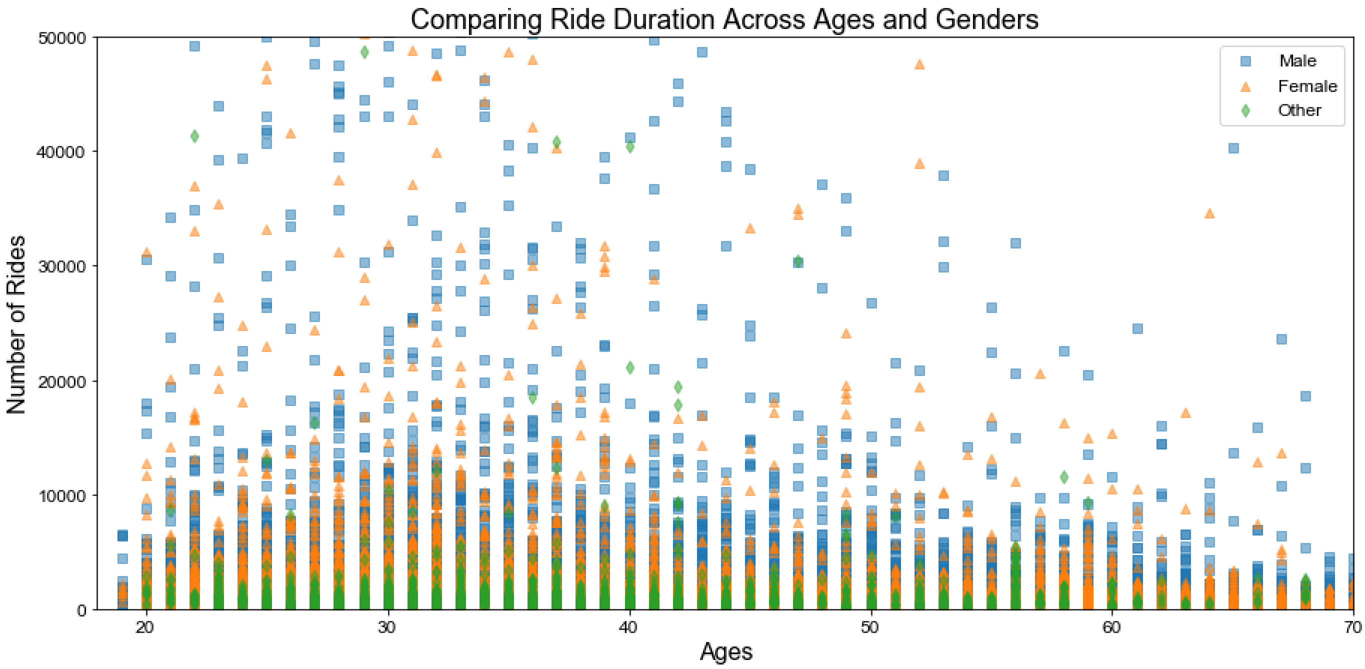
- **member_gender x age x duration_sec**

In [42]:
```
#Visualizing the data in an adapted scatterplot with a sample size of 250000 records
gobike_df_samp = master_df.sample(250000)

plt.figure(figsize = (15,7))

cat_markers = [['Male', 's'],
               ['Female', '^'],
               ['Other', 'd']]

for cat, marker in cat_markers:
    df_gender = gobike_df_samp[gobike_df_samp['member_gender'] == cat]
    plt.scatter(data = df_gender, x = 'member_age', y = 'duration_sec', marker = marker, alpha = .5);
plt.legend(['Male', 'Female', 'Other']);
plt.xlim(18, 70);
plt.ylim(0, 50000);
plt.title('Comparing Ride Duration Across Ages and Genders', fontsize = 18);
plt.xlabel('Ages');
plt.ylabel('Number of Rides');
plt.style.use('seaborn')
```



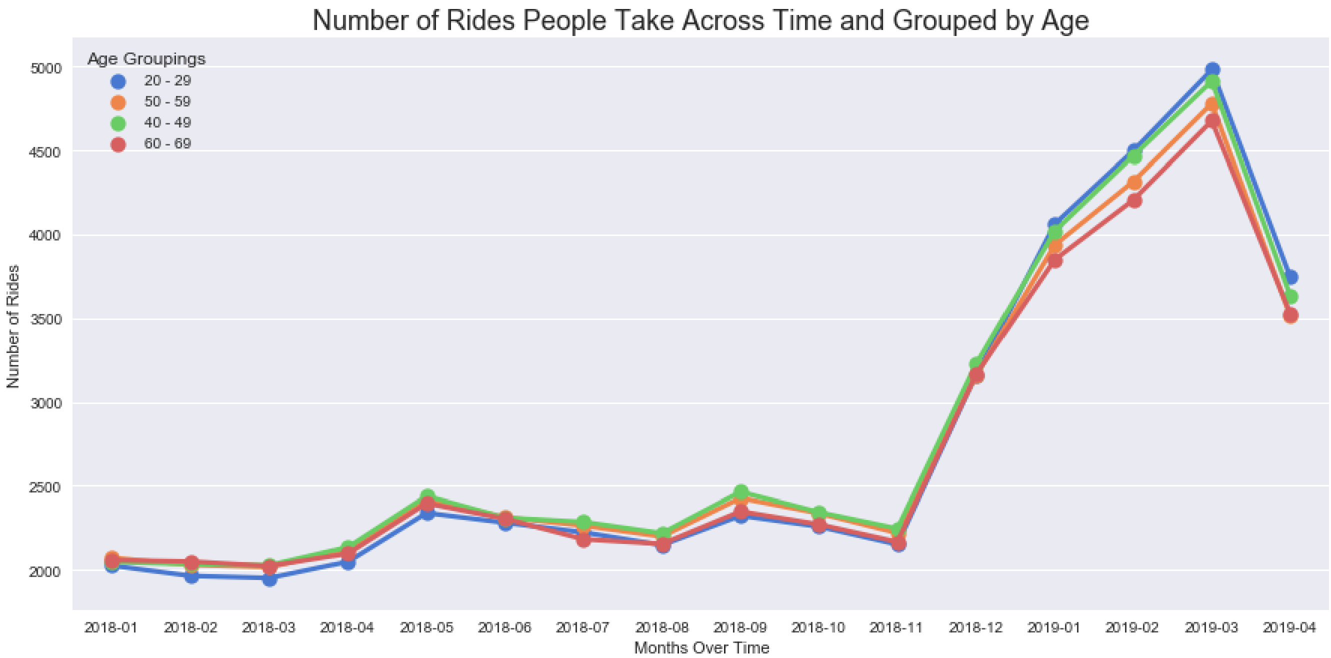
- We used some gender distinctive markers to determine how long rides tend to be across the genders and as a person gets older. Not surprisingly, the durations taper off as a person gets older, but the data doesn't at all seem to suggest that any single gender has a "stronger" longevity as they get older. In other words, we see just as many women and unnamed gender people riding just as long (if not longer) than men.

- **number_of_rides x age_group x month_year**

In [43]:

```
#Visualizing the data in an adapted pointplot with the errorbar removed
gobike_df_sub = master_df.loc[master_df['age_group'].isin(['10 - 19', '20 - 29', '30-39', '40 - 49',
                                                           '50 - 59', '60 - 69'])]

plt.figure(figsize = (15, 7))
plt.style.use('seaborn')
sb.pointplot(data = gobike_df_sub.sort_values(by='month_year'), x = 'month_year', y = 'bike_id',
             hue = 'age_group', palette = 'muted', ci = None);
plt.title('Number of Rides Per Age Group Over Time', fontsize = 18);
plt.legend(title = 'Age Groupings');
plt.title('Number of Rides People Take Across Time and Grouped by Age', fontsize = 18);
plt.xlabel('Months Over Time');
plt.ylabel('Number of Rides');
```



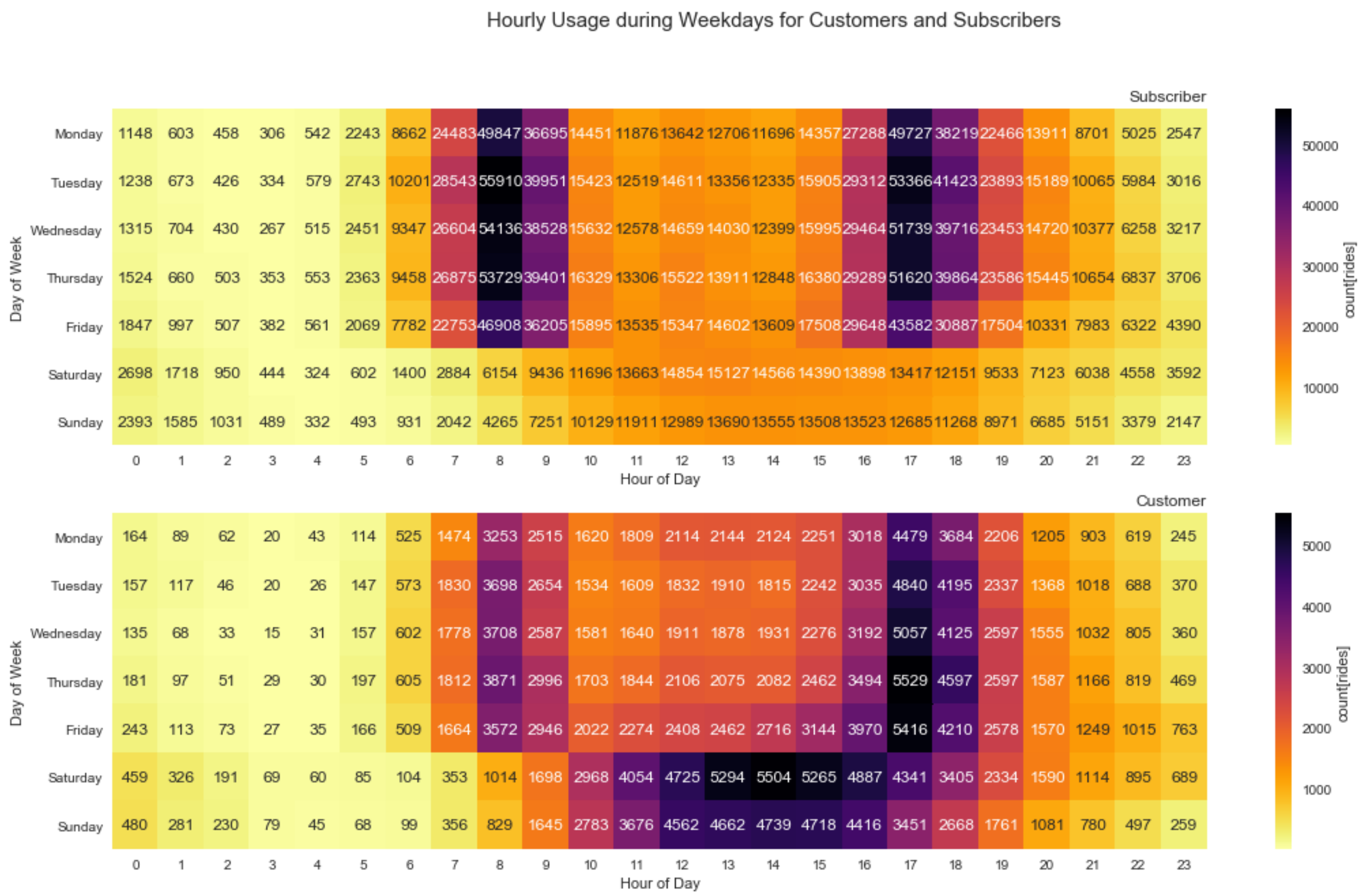
- In this visualization we try to analyse the relation between number of rides taken by people of different age groups across time. We see there is a spike in the beginning of 2019, my guess is tourists travelling in the Bay Area for the new year might have found the bikes very useful to commute in the area.

- **number_of_rides x hour_of_day x day_of_week**

```
In [44]: dow_order = ['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']
plt.figure(figsize=(18,10))
plt.suptitle('Hourly Usage during Weekdays for Customers and Subscribers' , fontsize=15)

plt.subplot(2, 1, 1)
subscribers = master_df.query('user_type == "Subscriber"')
st_counts = subscribers.groupby(['day', 'hour']).size()
st_counts = st_counts.reset_index(name='count')
st_counts = st_counts.pivot(index='day', columns='hour', values='count')
st_counts = st_counts.loc[dow_order,:]
sb.heatmap(st_counts, cmap='inferno_r' , annot = True, fmt = 'd' , cbar_kws = {'label' : 'count[rides]'});
plt.title('Subscriber', loc='right');
plt.xlabel('Hour of Day');
plt.ylabel('Day of Week');

plt.subplot(2, 1, 2)
customers = master_df.query('user_type == "Customer"')
ct_counts = customers.groupby(['day', 'hour']).size()
ct_counts = ct_counts.reset_index(name='count')
ct_counts = ct_counts.pivot(index='day', columns='hour', values='count')
ct_counts = ct_counts.loc[dow_order,:]
sb.heatmap(ct_counts, cmap='inferno_r' , annot = True, fmt = 'd', cbar_kws = {'label' : 'count[rides]'});
plt.title('Customer', loc='right');
plt.xlabel('Hour of Day');
plt.ylabel('Day of Week');
```



- In the final visualization we try to analyse the relation between bike usage during days of the week for customers and subscribers. Subscribers might be office goers as number of rides are increased between 07:00-08:00 and 17:00-18:00. In case of the normal customers the usage is high during weekends which makes sense.