Forecasting Bike Rental Demand in SF

data science capstone project 2

Rui Cao

2020-01-30

Introduction

Bike Sharing System is a modern way to rent bikes and its commonly used in metropolitan cities like San Francisco, New York City, and Washington DC. The process of obtaining membership, renting and returning bikes is automated through networks throughout the city. Using these systems, people can rent a bike from one place and return it to another as needed. Currently, there are more than 500 bike-sharing programs worldwide.

These systems generate a lot of data related to the duration of travel, departure location, arrival location, and time elapsed, which is very useful for study city mobility. Also, understand how the factors (like weather, traffic, hourly/daily trend) affect the number of bike rentals will help the company make a better business strategy.

Objective

In this project, I will use the SF Bay Area Bike Share dataset
https://www.kaggle.com/benhamner/sf-bay-area-bike-share

1. Data Wrangling - explore and merge all datasets together, make a new data frame ready to analyze.
2. Data Exploratory Analysis based on a new data frame and see what is the relationship between each factor and the total daily trips.
3. Feature engineering to get the best features which affect the daily trips most.
4. Find the best model to predict daily trips in SF on a selected station.

Data Wrangling

In this project, I will use 3 datasets: trip, station, and weather. they include a variety of bike-share information between Aug 29,2013, to Aug 30, 2015.

- ### Trip

  the trip dataset has 11 columns and 648717 entries, which describes information about each trip including start/end station id, duration of travel, date of travel, etc.

  1. Start by checking the missing values, I found there are 6619 nan values in the zip_code column. because we don't need this feature for future analysis, I will just leave it like this.

```
id                      0
duration                0
start_date              0
start_station_name      0
start_station_id        0
end_date                0
end_station_name        0
end_station_id          0
bike_id                 0
subscription_type       0
zip_code             6619
dtype: int64
```

figure1. missing values

2. Check the duration values and remove the outliers by only choose the trip duration of less than 1 hour.

```
count    669959.000000
mean         18.465831
std         370.923950
min           1.000000
25%           5.733333
50%           8.616667
75%          12.583333
max      287840.000000
Name: duration, dtype: float64
```

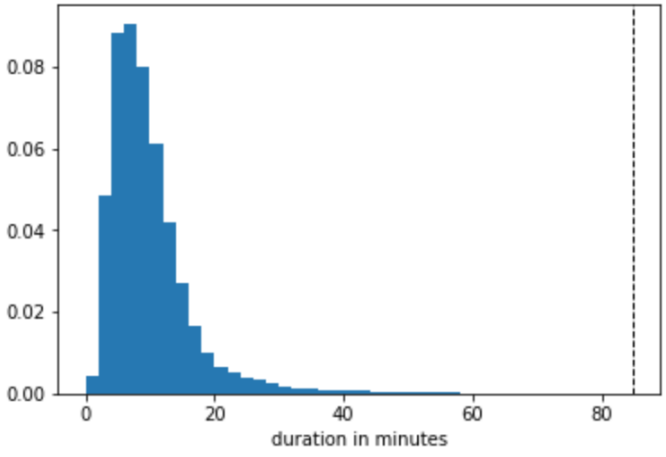


figure 2. duration info

3. Change the date column to DateTime format and use it to add new columns: date, day, hour, month, week. Renamed month and day columns, add holiday and working day columns.

4. Drop unuseful columns and change some columns datatype from object to category.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 648717 entries, 0 to 669958
Data columns (total 15 columns):
duration              648717 non-null float64
start_date            648717 non-null datetime64[ns]
start_station_name    648717 non-null category
start_station_id      648717 non-null category
end_date              648717 non-null datetime64[ns]
end_station_name      648717 non-null category
end_station_id        648717 non-null category
subscription_type     648717 non-null category
date                  648717 non-null category
day                   648717 non-null category
hour                  648717 non-null category
month                 648717 non-null category
week                  648717 non-null category
holiday               648717 non-null category
workingday            648717 non-null category
dtypes: category(12), datetime64[ns](2), float64(1)
memory usage: 27.9 MB
```

figure3. trip dataset info after wrangling

- ## Station

the station dataset contains information about 82 stations including name, location, dock_count, city, and installation_date of each station. there are no missing values in this dataset, I just change the columns with ‘object’ datatype to ‘category’.

	id	name	lat	long	dock_count	city	installation_date
0	2	San Jose Diridon Caltrain Station	37.329732	-121.901782	27	San Jose	8/6/2013
1	3	San Jose Civic Center	37.330698	-121.888979	15	San Jose	8/5/2013
2	4	Santa Clara at Almaden	37.333988	-121.894902	11	San Jose	8/6/2013
3	5	Adobe on Almaden	37.331415	-121.893200	19	San Jose	8/5/2013
4	6	San Pedro Square	37.336721	-121.894074	15	San Jose	8/7/2013

figure 4. station data frame

- ## Weather

weather dataset includes weather information for 4 major areas: San Francisco, Redwood City, Palo Alto, Mountain View, San Jose. There are almost missing values in every column.

```
date                                0
max_temperature_f                   4
mean_temperature_f                  4
min_temperature_f                   4
max_dew_point_f                    54
mean_dew_point_f                   54
min_dew_point_f                    54
max_humidity                       54
mean_humidity                      54
min_humidity                       54
max_sea_level_pressure_inches       1
mean_sea_level_pressure_inches      1
min_sea_level_pressure_inches       1
max_visibility_miles               13
mean_visibility_miles              13
min_visibility_miles               13
max_wind_Speed_mph                  1
mean_wind_speed_mph                 1
max_gust_speed_mph                899
precipitation_inches                1
cloud_cover                         1
events                           3143
wind_dir_degrees                    1
zip_code                            0
dtype: int64
```

figure 5. missing value in the weather data frame

the events column has the most missing values. It categorizes the weather type in fog, fog-rain, rain, rain- thunderstorms. I fill the nan value with ‘normal’ first and then level them up based on wind speed and visibility miles using numeric values. After exploring the dataset make sure there are no unacceptable outliers, I convert objects columns to numeric values and fill nan values with its own average except events column. And finally, map the 4 zip codes with the corresponding city name, delete unnecessary columns.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3665 entries, 0 to 3664
Data columns (total 25 columns):
date                             3665 non-null category
max_temperature_f                3665 non-null float64
mean_temperature_f               3665 non-null float64
min_temperature_f                3665 non-null float64
max_dew_point_f                  3665 non-null float64
mean_dew_point_f                 3665 non-null float64
min_dew_point_f                  3665 non-null float64
max_humidity                     3665 non-null float64
mean_humidity                    3665 non-null float64
min_humidity                     3665 non-null float64
max_sea_level_pressure_inches    3665 non-null float64
mean_sea_level_pressure_inches   3665 non-null float64
min_sea_level_pressure_inches    3665 non-null float64
max_visibility_miles             3665 non-null float64
mean_visibility_miles            3665 non-null float64
min_visibility_miles             3665 non-null float64
max_wind_Speed_mph               3665 non-null float64
mean_wind_speed_mph              3665 non-null float64
max_gust_speed_mph               3665 non-null float64
precipitation_inches             3665 non-null float64
cloud_cover                      3665 non-null float64
events                           3665 non-null int64
wind_dir_degrees                 3665 non-null float64
city                             3665 non-null category
week                             3665 non-null int64
dtypes: category(2), float64(21), int64(2)
memory usage: 695.3 KB
```

figure 6. weather dataset info after wrangling

- ## Merging Dataset

I now need to create a data frame for machine learning. from the trip data frame, group by date,station_name, holiday, weekday to get the trip_count on each day by each station, left merge with station data frame on station_name column.

	date	station_name	holiday	workingday	trip_count	dock_count	city
0	2013-08-29	2nd at Folsom	False	True	12	19.0	San Francisco
1	2013-08-29	2nd at South Park	False	True	11	15.0	San Francisco
2	2013-08-29	2nd at Townsend	False	True	8	27.0	San Francisco
3	2013-08-29	5th at Howard	False	True	12	15.0	San Francisco
4	2013-08-29	Adobe on Almaden	False	True	3	19.0	San Jose

figure6. merging station and trip data frame

The new data frame has 1150 missing values in the city and dock_count columns, it is because there are 4 stations in the trip dataset not included in the station dataset. Because there is no significant difference in dock_count between each city, I will fill the city column based on the location of station_name and filling the dock_count column with the overall dock mean value.

after fixing the missing value, I will keep merging the data frame with the weather on date and city columns. There are no missing values after merging, just need to change some columns from object to category datatype.

Until now, I've got 3 well-cleaned data frames and a combined data frame for EDA and Machine learning purposes.

Exploratory Data Analysis

- ## Most Commonly used Stations

There is a total of 82 stations listed in this dataset. My main target in this project is to predict the daily trips for an individual station. So the first thing I need to do is to check out which station is the busiest station in SF. I check the top 10 commonly used stations based on the start-station and end-station and find the station named: San Fransico Caltrain(Townsend at 4th) is the most popular station. I will choose this station for my machine learning.

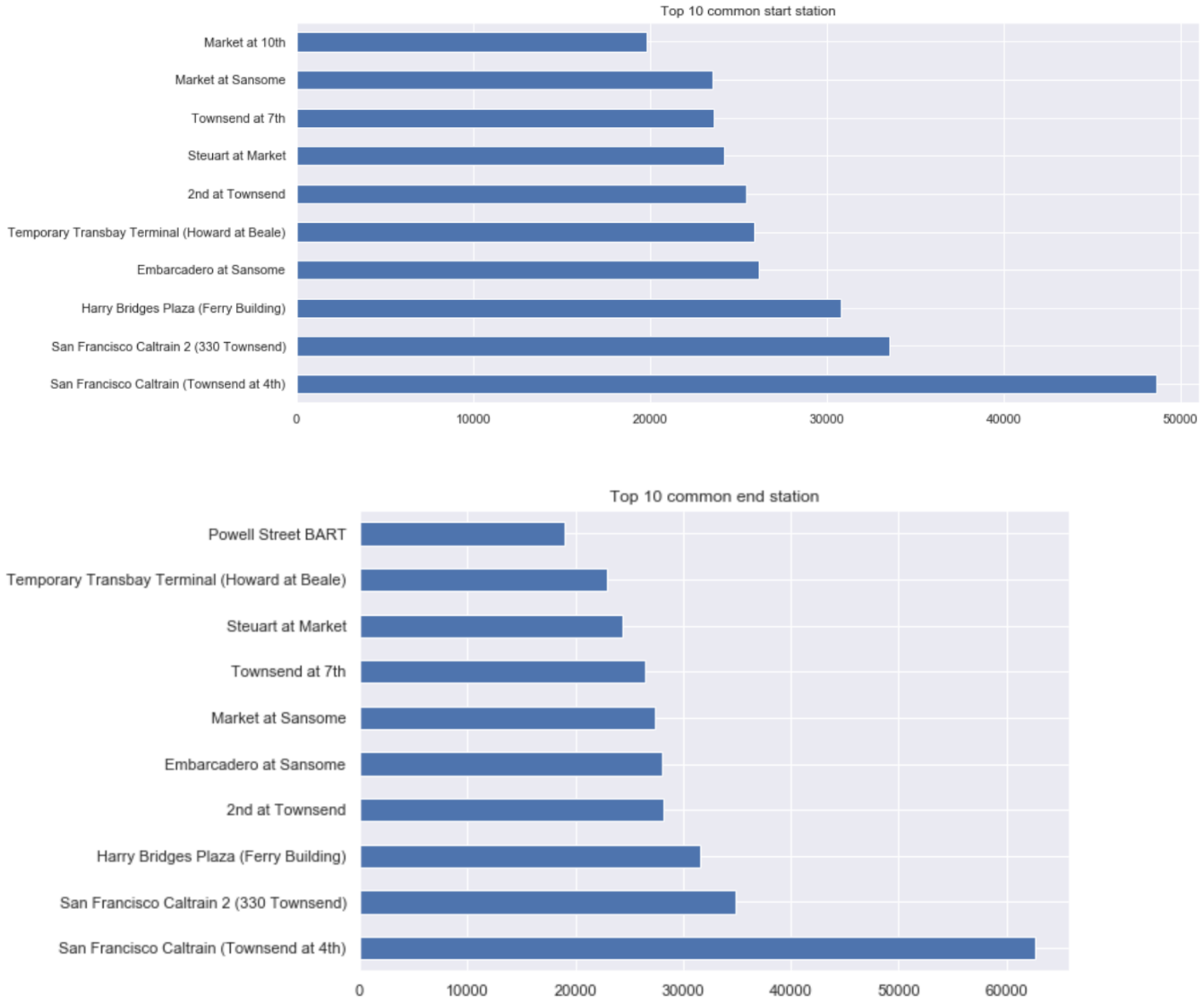


figure 7. most commonly unused station in SF

- ## Total Trip counts distribution by subscription type

  To have a deep understanding of who uses this service most, I start by looking into trips by different subscription types: subscriber, the customer(non-subscriber). First, I Check the total trip counts distribution on timestamp by different subscriptions. the people using bike share are most subscribers. The number of daily trips is pretty stable only has a big drop every year around December - January. I assume that most people using this service are for daily transportation, the less usage at the end of the year is due to the holiday season.

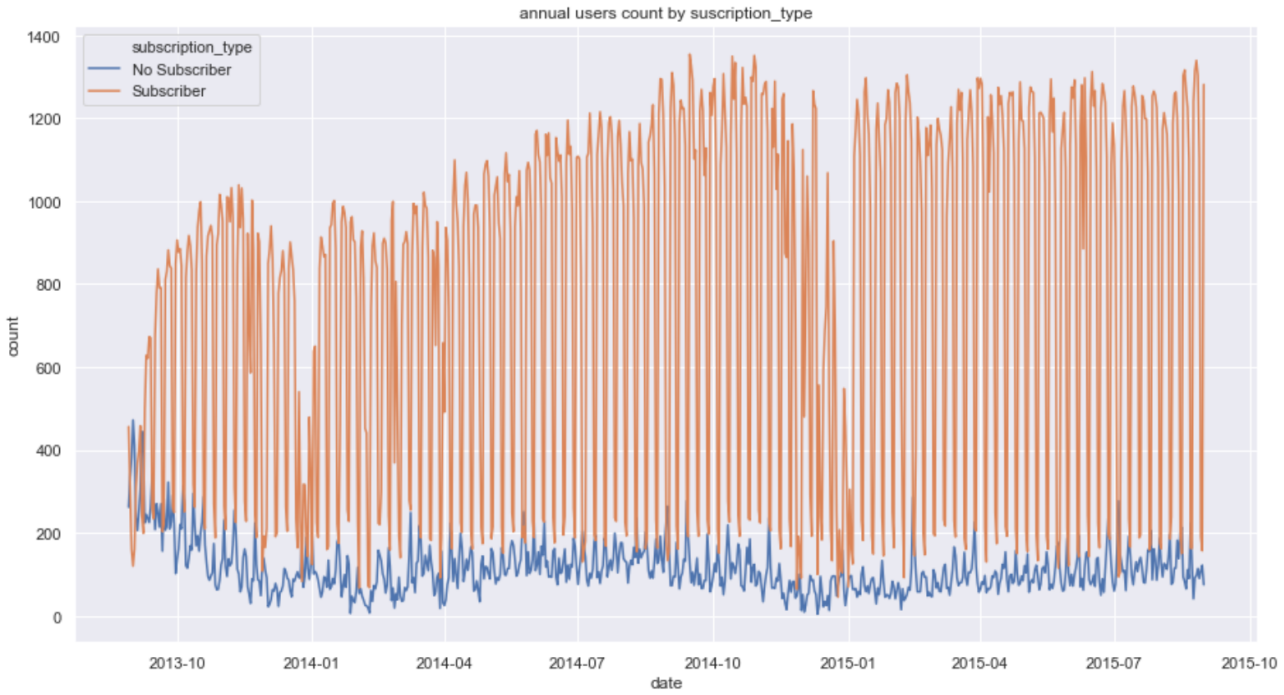


figure 8.overall trips count by subscripthin_type

To confirm my assumption of the main target and how they use the service, I also check the trip counts on weekdays/weekends, holidays by different suscrption_type. As shown in the pictures below, subscribers using the service most times in weekdays and non_hoidays, which approved my assumption.

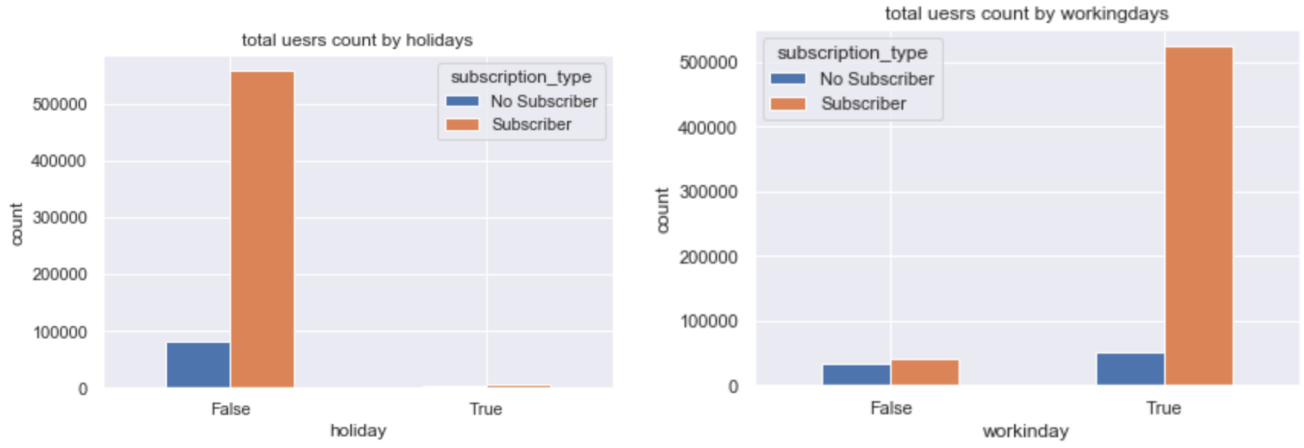


figure 9 trip counts on weekday/holiday by suscription_type

- ## Trip Counts distribution by Month/Day/Hour

Because the weather in SF is very warm, the trip counts by month has no big drop in wintertime. I believe the slight drop in December is due to holiday reasons.

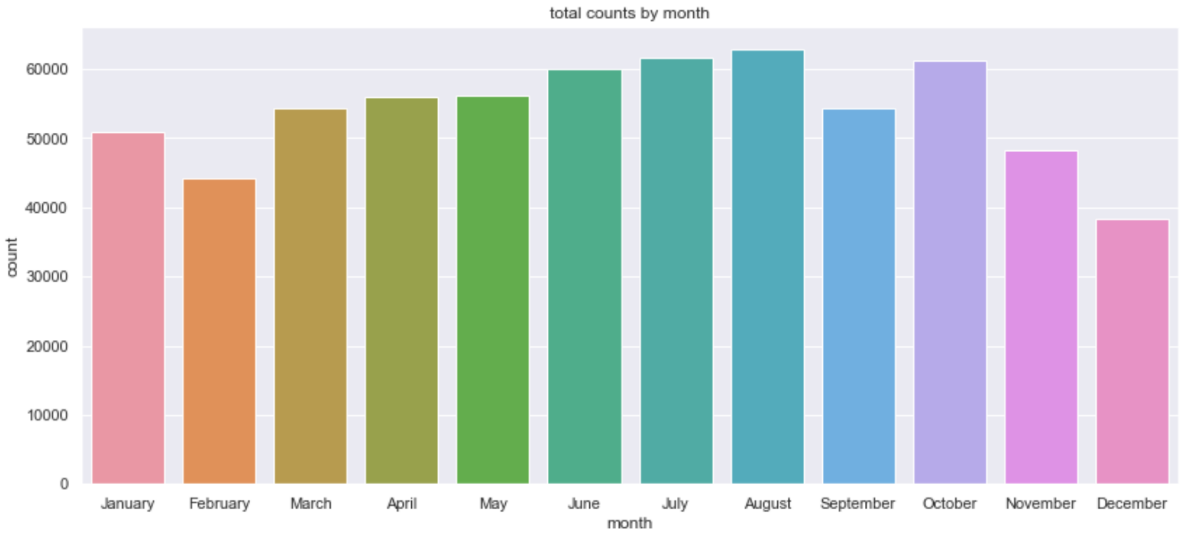


figure 10. trip counts by month

As we already know, customers use the bike-share system most on working days as personal transportation. which hour or which day they use the most? As I deep into the data, I find out that there are no significant different trip counts by non-subscribers. But for subscribers, they usually use the bike on weekdays, especially at peak hours (8 am in the morning and 5 pm at night). I assume that the slight drop on Friday's peak hours due to some personal choice to work at home on that day.

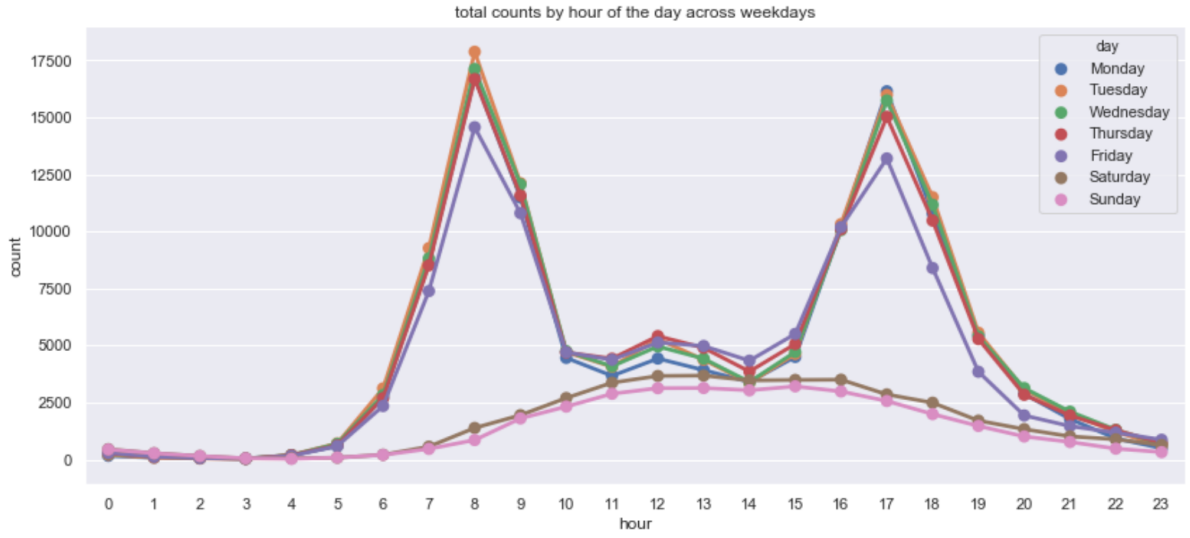


figure 11. trip counts of the day across weekdays

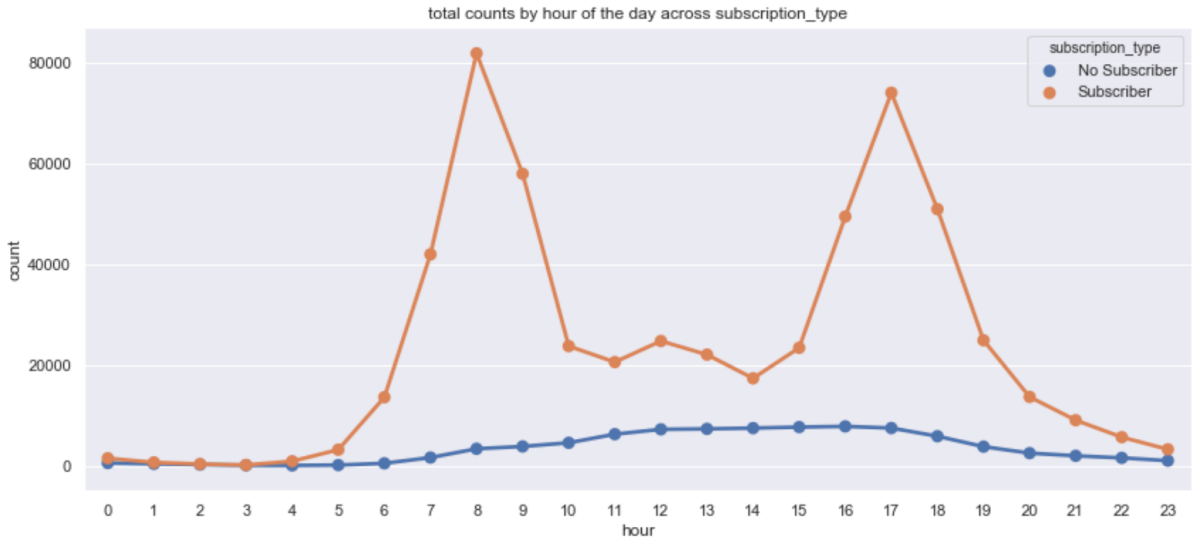


figure 12. trip counts of the day across subscription_type

- ## Trip counts distribution by weather features

  There is no clear relationship between weather features and daily trips. poeple are willing to use bike rental service on a normal day rather than fog_raining day. And there is more fog weather in wintertime them summer.

  compared to temperature, wind speed, and visibility vs trip counts, which do have slightly effect on trips, but would not key feature.

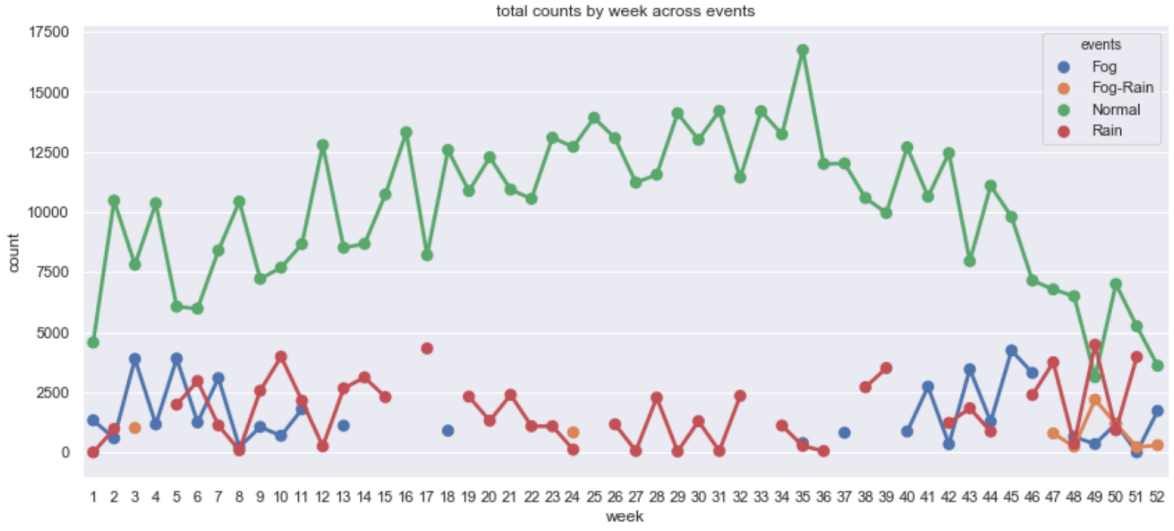
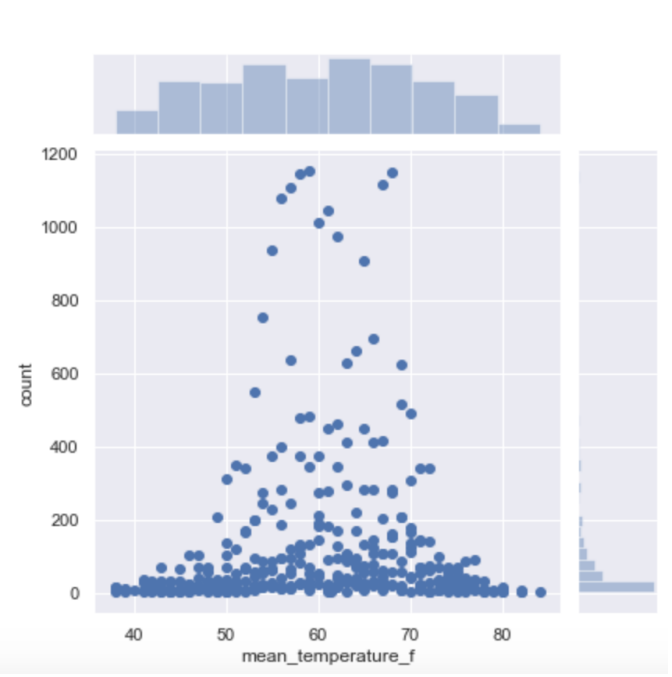
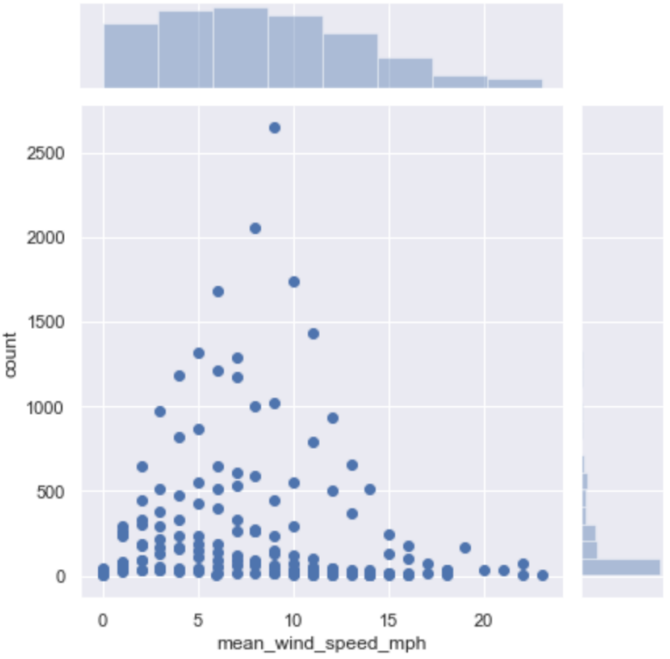


figure 12. trip counts by week across events





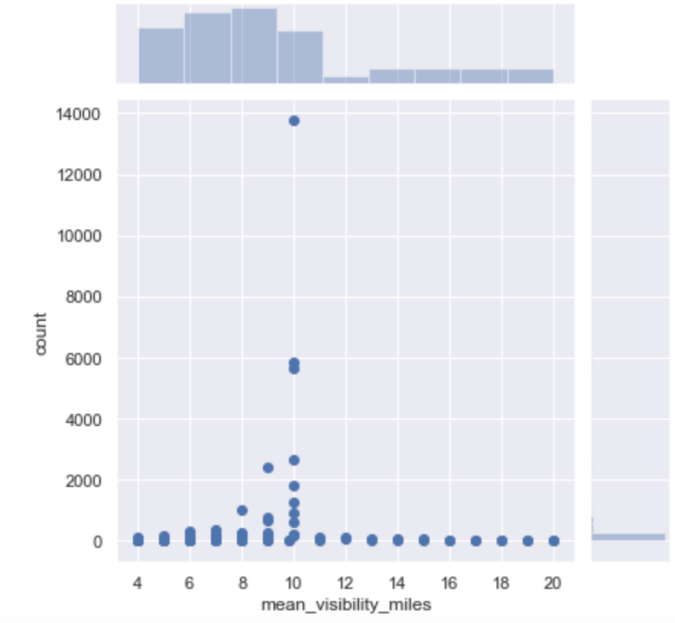


figure 13. trip counts vs. temperature,wind_speed, and visibility

Machine Learning

- ## trip counts predictive model

  In this project, I aim to build a regression model to predict daily trips on a target station. based on the data, I decided to choose the most popular station: San Francisco Caltrain (Townsend at 4th) as my predict station. I first change categorical columns into numerical data, then drop the columns like the city, station name, date and set the train, test split ready for modeling.

  I try to find the best performance model by checking the mean absolute error on 5 different algorithms: Random Forest, Decision Tree, Linear Regression, Gradient Boosting, and Adaboost. After finding the best parameter using Gridsearch, I decide to use a random forest regressor which has the smallest mean absolute error of 11.69.

figure 14. predicted value vs. actual value of daily trips

- feature engineering

It does not surprise me that the working day has a significant impact on daily trips based on my Exploratory data analysis. Holidays, week of the yeat also are some key features to predict daily trips then weather.

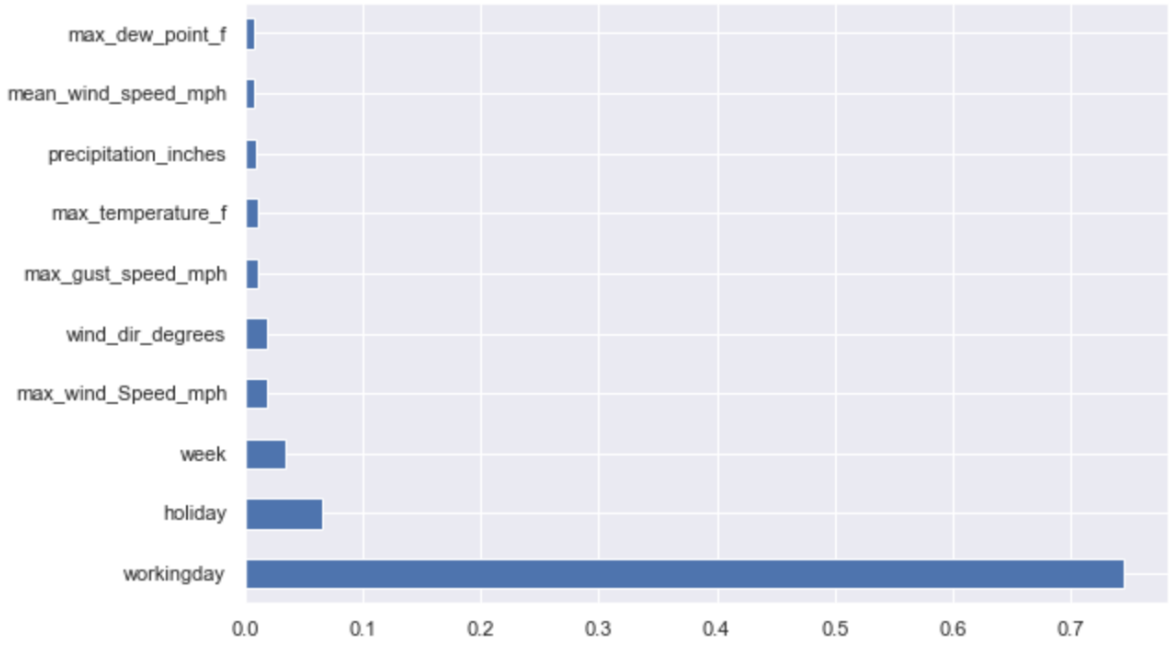


figure 15. Importance Features

Conclusion

I believe I have made a good model to predict daily trips that will take with the bike-sharing service at a particular station. The mean absolute error of my model is about 11 times a day. The company could use my model to predict any station usage they are looking for and get a good rough estimate of the dock account they should keep for each day.

To make the model more accurate and usable, The next step I will focus on the larger numb er of trips, try to figure out what features affect the numbers and how to optimize the model to make better predictions on those particular days.