In [442]:

```
%matplotlib inline
import pandas as pd
import numpy as np
from scipy.stats.stats import pearsonr
from pandas.tseries.holiday import USFederalHolidayCalendar
from pandas.tseries.offsets import CustomBusinessDay
from datetime import datetime
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
```

In [235]:

```
trips = pd.read_csv('trip_data.csv')
stations = pd.read_csv('station_data.csv')
weather = pd.read_csv('weather_data.csv')
```

Data Exploration

Trip df & Station df

I'm going to summarize the data in trips df and station df and combine them into one df. To that end, I'll create some helper functions

In [124]:

```
def create_station_mapping(station_data):
    """
    Create a mapping from station IDs to cities, returning the
    result as a dictionary.
    """
    station_map = {}
    for data_file in station_data:
        with open(data_file, 'r') as f_in:

            weather_reader = csv.DictReader(f_in)

            for row in weather_reader:
                station_map[row['Id']] = row['Name']
    return station_map
```

In [372]:

```
def summarise_data(trip_in, station_data, trip_out):

    station_map = create_station_mapping(station_data)

    with open(trip_out, 'w') as f_out:
        out_colnames = ['start_date', 'start_year',
                        'start_month', 'start_hour', 'end_date', 'end_hour', 'we
ekday',
                        'start_station', 'end_station', 'subscription_type']

        trip_writer = csv.DictWriter(f_out, fieldnames = out_colnames)
        trip_writer.writeheader()

        for data_file in trip_in:
            with open(data_file, 'r') as f_in:
                trip_reader = csv.DictReader(f_in)

                # collect data from and process each row
                for row in trip_reader:
                    new_point = {}

                    trip_date = datetime.strptime(row['Start Date'], '%d/%m/%Y %
H:%M')
                    new_point['start_date']  = trip_date.strftime('%Y-%m-%d')
                    new_point['start_year']  = trip_date.strftime('%Y')
                    new_point['start_month'] = trip_date.strftime('%m')
                    new_point['start_hour']  = trip_date.strftime('%H')
                    new_point['weekday']     = trip_date.strftime('%a')

                    trip_enddate = datetime.strptime(row['End Date'], '%d/%m/%Y
 %H:%M')
                    new_point['end_date']  = trip_enddate.strftime('%Y-%m-%d')
                    new_point['end_hour']  = trip_enddate.strftime('%H')

                    new_point['start_station'] = station_map[row['Start
Station']]
                    new_point['end_station'] = station_map[row['End Station']]
                    if 'Subscription Type' in row:
                        new_point['subscription_type'] = row['Subscription
Type']
                    else:
                        new_point['subscription_type'] = row['Subscriber Type']


                    trip_writer.writerow(new_point)
```

In [184]:

```
station_data = ['station_data.csv']
trip_in = ['trip_data.csv']
trip_out = 'trip_summary.csv'
summarise_data(trip_in, station_data, trip_out)
```

In [185]:

```
trips = pd.read_csv(trip_out)
```

In [186]:

```
# We check to see if there are any NA values and then see a sample of the first
 few rows
trips.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 354152 entries, 0 to 354151
Data columns (total 10 columns):
start_date           354152 non-null object
start_year           354152 non-null int64
start_month          354152 non-null int64
start_hour           354152 non-null int64
end_date             354152 non-null object
end_hour             354152 non-null int64
weekday              354152 non-null object
start_station        354152 non-null object
end_station          354152 non-null object
subscription_type    354152 non-null object
dtypes: int64(4), object(6)
memory usage: 27.0+ MB
```

In [187]:

```
trips.head(5)
```

Out[187]:

	start_date	start_year	start_month	start_hour	end_date	end_hour	weekday	start
0	2015-08-31	2015	8	23	2015-08-31	23	Mon	Harr Bridg Plaz Build
1	2015-08-31	2015	8	23	2015-08-31	23	Mon	San Shop Cent
2	2015-08-31	2015	8	23	2015-08-31	23	Mon	Post Kear
3	2015-08-31	2015	8	23	2015-08-31	23	Mon	San City
4	2015-08-31	2015	8	23	2015-08-31	23	Mon	Emb at Fc

This is the number of trips started at each station for a given date and hour

In [188]:

```
start_trips = trips.groupby(['start_station','start_date','start_hour'], as_inde
x=False).count()
start_trips = start_trips.iloc[:,0:4]
start_trips.rename(columns = {'start_year':'start_trips'}, inplace = True)
start_trips.head()
```

Out[188]:

	start_station	start_date	start_hour	start_trips
0	2nd at Folsom	2014-09-01	14	1
1	2nd at Folsom	2014-09-01	16	1
2	2nd at Folsom	2014-09-02	6	1
3	2nd at Folsom	2014-09-02	7	1
4	2nd at Folsom	2014-09-02	8	2

This is the number of trips ended at each station for a given date and hour

In [197]:

```
end_trips = trips.groupby(['end_station','end_date','end_hour'],
as_index=False).count()
end_trips = end_trips.iloc[:,0:4]
end_trips.rename(columns = {'start_date':'end_trips'}, inplace = True)
end_trips.head()
```

Out[197]:

	end_station	end_date	end_hour	end_trips
0	2nd at Folsom	2014-09-01	17	1
1	2nd at Folsom	2014-09-01	20	1
2	2nd at Folsom	2014-09-02	8	3
3	2nd at Folsom	2014-09-02	9	3
4	2nd at Folsom	2014-09-02	11	1

From these we can calculate the net rate of bike renting

In [405]:

```
test = pd.merge(start_trips, end_trips, how='outer', left_on=['start_station','s
tart_date','start_hour'], right_on = ['end_station','end_date','end_hour'])
```

In [406]:

```
test.end_trips = test.end_trips.fillna(0)
```

In [407]:

```
test['net_trips'] = test.end_trips - test.start_trips
```

In [408]:

```
test = test.drop(['end_station','end_date','end_hour'],1)
```

In [409]:

```
test = test.dropna()
```

In [410]:

```
test.head()
```

Out[410]:

	start_station	start_date	start_hour	start_trips	end_trips	net_trips
0	2nd at Folsom	2014-09-01	14.0	1.0	0.0	-1.0
1	2nd at Folsom	2014-09-01	16.0	1.0	0.0	-1.0
2	2nd at Folsom	2014-09-02	6.0	1.0	0.0	-1.0
3	2nd at Folsom	2014-09-02	7.0	1.0	0.0	-1.0
4	2nd at Folsom	2014-09-02	8.0	2.0	3.0	1.0

In [411]:

```
stns = pd.get_dummies(test.start_station)
```

In [412]:

```
test = test.merge(stns, left_index = True, right_index = True)
```

In [414]:

```
test = test.drop('start_station',1)
```

Let's find out what are the top 10 most popular stations to start a ride

In [146]:

```
net_trips = start_trips.copy()
net_trips['end_trips'] = end_trips.end_trips
net_trips['net_trips'] = end_trips.end_trips - start_trips.start_trips
net_trips.rename(columns = {'start_station':'station'}, inplace = True)
net_trips.rename(columns = {'start_hour':'hour'}, inplace = True)
net_trips.head()
```

Out[146]:

	station	hour	start_trips	end_trips	net_trips
0	2nd at Folsom	0	17	15	-2.0
1	2nd at Folsom	1	9	1	-8.0
2	2nd at Folsom	2	5	1	-4.0
3	2nd at Folsom	4	70	3	-67.0
4	2nd at Folsom	5	125	212	87.0

In [381]:

```
total_trips = net_trips.groupby('station', as_index = False).sum()
total_trips.head()
top10_stations_start = list(total_trips.sort_values('start_trips',
ascending=0).station[:10])
top10_stations_start
```

Out[381]:

```
['San Francisco Caltrain (Townsend at 4th)',
 'San Francisco Caltrain 2 (330 Townsend)',
 'Harry Bridges Plaza (Ferry Building)',
 'Temporary Transbay Terminal (Howard at Beale)',
 'Embarcadero at Sansome',
 '2nd at Townsend',
 'Townsend at 7th',
 'Steuart at Market',
 'Market at 10th',
 'Market at Sansome']
```

In [ ]:

```
And top 10 stations to end a ride
```

In [382]:

```
top10_stations_end = list(total_trips.sort_values('end_trips', ascending=0).stat
ion[:10])
top10_stations_end
```

Out[382]:

```
['San Francisco Caltrain (Townsend at 4th)',
 'San Francisco Caltrain 2 (330 Townsend)',
 'Harry Bridges Plaza (Ferry Building)',
 '2nd at Townsend',
 'Embarcadero at Sansome',
 'Townsend at 7th',
 'Market at Sansome',
 'Temporary Transbay Terminal (Howard at Beale)',
 'Steuart at Market',
 'Powell Street BART']
```

Weather df

In [49]:

```
weather_df.head()
```

Out[49]:

	Date	Max TemperatureF	Mean TemperatureF	Min TemperatureF	Max Dew PointF	MeanDew PointF	Dewpo
0	01/09/2014	83.0	70.0	57.0	58.0	56.0	52.0
1	02/09/2014	72.0	66.0	60.0	58.0	57.0	55.0
2	03/09/2014	76.0	69.0	61.0	57.0	56.0	55.0
3	04/09/2014	74.0	68.0	61.0	57.0	57.0	56.0
4	05/09/2014	72.0	66.0	60.0	57.0	56.0	54.0

5 rows × 24 columns

In [271]:

```
weather.Date = pd.to_datetime(weather.Date, format='%d/%m/%Y')
```

```
In [273]:
```

```
weather.head()
```

```
Out[273]:
```

	Date	Max TemperatureF	Mean TemperatureF	Min TemperatureF	Max Dew PointF	MeanDew PointF	Min DewpointF
0	2014-09-01	83.0	70.0	57.0	58.0	56.0	52.0
1	2014-09-02	72.0	66.0	60.0	58.0	57.0	55.0
2	2014-09-03	76.0	69.0	61.0	57.0	56.0	55.0
3	2014-09-04	74.0	68.0	61.0	57.0	57.0	56.0
4	2014-09-05	72.0	66.0	60.0	57.0	56.0	54.0

5 rows × 24 columns

```
In [274]:
```

```
# Looks like we have multiple zip codes per date
weather.Zip.unique()
```

```
Out[274]:
```

```
array([94107, 94063, 94301, 94041, 95113])
```

In [275]:

```
# See which zip code is "cleanest"
for z in weather.Zip.unique():
    print("Zip code:", z)
    print(weather[weather.Zip == z].isnull().sum())
    print()
```

```
Zip code: 94107
Date                           0
Max TemperatureF               0
Mean TemperatureF              0
Min TemperatureF               0
Max Dew PointF                 0
MeanDew PointF                 0
Min DewpointF                  0
Max Humidity                   0
Mean Humidity                  0
Min Humidity                   0
Max Sea Level PressureIn       0
Mean Sea Level PressureIn      0
Min Sea Level PressureIn       0
Max VisibilityMiles            0
Mean VisibilityMiles           0
Min VisibilityMiles            0
Max Wind SpeedMPH              0
Mean Wind SpeedMPH             0
Max Gust SpeedMPH              6
PrecipitationIn                0
CloudCover                     0
Events                       273
WindDirDegrees                 0
Zip                            0
dtype: int64

Zip code: 94063
Date                           0
Max TemperatureF               0
Mean TemperatureF              0
Min TemperatureF               0
Max Dew PointF                 0
MeanDew PointF                 0
Min DewpointF                  0
Max Humidity                   0
Mean Humidity                  0
Min Humidity                   0
Max Sea Level PressureIn       0
Mean Sea Level PressureIn      0
Min Sea Level PressureIn       0
Max VisibilityMiles            0
Mean VisibilityMiles           0
Min VisibilityMiles            0
Max Wind SpeedMPH              0
Mean Wind SpeedMPH             0
Max Gust SpeedMPH            209
PrecipitationIn                0
CloudCover                     0
Events                       317
WindDirDegrees                 0
Zip                            0
dtype: int64

Zip code: 94301
Date                           0
Max TemperatureF               1
Mean TemperatureF              1
Min TemperatureF               1
Max Dew PointF                47
MeanDew PointF                47
```

```
Min DewpointF                 47
Max Humidity                  47
Mean Humidity                 47
Min Humidity                  47
Max Sea Level PressureIn       1
Mean Sea Level PressureIn      1
Min Sea Level PressureIn       1
Max VisibilityMiles            5
Mean VisibilityMiles           5
Min VisibilityMiles            5
Max Wind SpeedMPH              1
Mean Wind SpeedMPH             1
Max Gust SpeedMPH            313
PrecipitationIn                1
CloudCover                     1
Events                       324
WindDirDegrees                 1
Zip                            0
dtype: int64

Zip code: 94041
Date                           0
Max TemperatureF               3
Mean TemperatureF              3
Min TemperatureF               3
Max Dew PointF                 3
MeanDew PointF                 3
Min DewpointF                  3
Max Humidity                   3
Mean Humidity                  3
Min Humidity                   3
Max Sea Level PressureIn       0
Mean Sea Level PressureIn      0
Min Sea Level PressureIn       0
Max VisibilityMiles            0
Mean VisibilityMiles           0
Min VisibilityMiles            0
Max Wind SpeedMPH              0
Mean Wind SpeedMPH             0
Max Gust SpeedMPH              7
PrecipitationIn                0
CloudCover                     0
Events                       311
WindDirDegrees                 0
Zip                            0
dtype: int64

Zip code: 95113
Date                           0
Max TemperatureF               0
Mean TemperatureF              0
Min TemperatureF               0
Max Dew PointF                 0
MeanDew PointF                 0
Min DewpointF                  0
Max Humidity                   0
Mean Humidity                  0
Min Humidity                   0
Max Sea Level PressureIn       0
Mean Sea Level PressureIn      0
Min Sea Level PressureIn       0
```

```
Max VisibilityMiles          0
Mean VisibilityMiles         0
Min VisibilityMiles          0
Max Wind SpeedMPH            0
Mean Wind SpeedMPH           0
Max Gust SpeedMPH            6
PrecipitationIn              0
CloudCover                   0
Events                     313
WindDirDegrees               0
Zip                          0
dtype: int64
```

In [276]:

```
# 94107 seems to be the best so I'll stick to this zip code from now on
weather = weather[weather.Zip == 94107]
```

In [277]:

```
weather.Events.unique()
```

Out[277]:

```
array([nan, 'Rain', 'Fog', 'Fog-Rain', 'Rain-Thunderstorm'], dtype=o
bject)
```

In [278]:

```
# nan most likely means normal weather condition
weather.loc[weather.Events == 'Rain', 'Events'] = "Rain"
weather.loc[weather.Events.isnull(), 'Events'] = "Normal"
```

```
/Users/VAN/anaconda/lib/python3.6/site-packages/pandas/core/indexin
g.py:517: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/panda
s-docs/stable/indexing.html#indexing-view-versus-copy
  self.obj[item] = s
```

In [279]:

```
weather.Events
```

```
Out[279]:

0                  Normal
1                  Normal
2                  Normal
3                  Normal
4                  Normal
5                  Normal
6                  Normal
7                  Normal
8                  Normal
9                  Normal
10                 Normal
11                 Normal
12                 Normal
13                 Normal
14                 Normal
15                 Normal
16                   Rain
17                   Rain
18                 Normal
19                 Normal
20                 Normal
21                 Normal
22                   Rain
23                   Rain
24                   Rain
25                 Normal
26                 Normal
27                 Normal
28                 Normal
29                 Normal
             ...
335                Normal
336                Normal
337                Normal
338                Normal
339    Rain-Thunderstorm
340                Normal
341                Normal
342                Normal
343                Normal
344                Normal
345                Normal
346                Normal
347                Normal
348                Normal
349                Normal
350                Normal
351                Normal
352                Normal
353                Normal
354                Normal
355                Normal
356                Normal
357                Normal
358                Normal
359                Normal
360                Normal
361                Normal
362                  Rain
363                Normal
```

```
363                Normal

364                Normal
Name: Events, Length: 365, dtype: object
```

In [280]:

```
# Convert values to new features
events = pd.get_dummies(weather.Events)
weather = weather.merge(events, left_index = True, right_index = True)
```

In [283]:

```
# Remove features we no longer need
weather = weather.drop(['Events','Zip'],1)
```

In [284]:

```
weather.head()
```

Out[284]:

	Date	Max TemperatureF	Mean TemperatureF	Min TemperatureF	Max Dew PointF	MeanDew PointF	Min DewpointF
0	2014-09-01	83.0	70.0	57.0	58.0	56.0	52.0
1	2014-09-02	72.0	66.0	60.0	58.0	57.0	55.0
2	2014-09-03	76.0	69.0	61.0	57.0	56.0	55.0
3	2014-09-04	74.0	68.0	61.0	57.0	57.0	56.0
4	2014-09-05	72.0	66.0	60.0	57.0	56.0	54.0

5 rows × 27 columns

In [285]:

```
weather.isnull().sum()
```

Out[285]:

```
Date                          0
Max TemperatureF              0
Mean TemperatureF             0
Min TemperatureF              0
Max Dew PointF                0
MeanDew PointF                0
Min DewpointF                 0
Max Humidity                  0
Mean Humidity                 0
Min Humidity                  0
Max Sea Level PressureIn      0
Mean Sea Level PressureIn     0
Min Sea Level PressureIn      0
Max VisibilityMiles           0
Mean VisibilityMiles          0
Min VisibilityMiles           0
Max Wind SpeedMPH             0
Mean Wind SpeedMPH            0
Max Gust SpeedMPH             6
PrecipitationIn               0
CloudCover                    0
WindDirDegrees                0
Fog                           0
Fog-Rain                      0
Normal                        0
Rain                          0
Rain-Thunderstorm             0
dtype: int64
```

In [286]:

```
# Still have NAs in max gust speed. However, max wind speed and max gust speed i
s likely to be
# correlated => can use the former to fill the latter. Verify correlation with P
earson's corr

print(pearsonr(weather['Max Wind SpeedMPH'][weather['Max Gust SpeedMPH'] >= 0],
               weather['Max Gust SpeedMPH'][weather['Max Gust SpeedMPH'] >= 0]))
```

```
(0.75997233822788968, 8.7711694198354035e-69)
```

In [287]:

```
#For each value of max_wind, find the median max_gust
weather.loc[weather['Max Gust SpeedMPH'].isnull(), 'Max Gust SpeedMPH'] = weathe
r.groupby('Max Wind SpeedMPH')['Max Gust SpeedMPH'].apply(lambda x: x.fillna(x.m
edian()))
```

In [288]:

```
weather.dtypes
```

Out[288]:

```
Date                         datetime64[ns]
Max TemperatureF                    float64
Mean TemperatureF                   float64
Min TemperatureF                    float64
Max Dew PointF                      float64
MeanDew PointF                      float64
Min DewpointF                       float64
Max Humidity                        float64
Mean Humidity                       float64
Min Humidity                        float64
Max Sea Level PressureIn            float64
Mean Sea Level PressureIn           float64
Min Sea Level PressureIn            float64
Max VisibilityMiles                 float64
Mean VisibilityMiles                float64
Min VisibilityMiles                 float64
Max Wind SpeedMPH                   float64
Mean Wind SpeedMPH                  float64
Max Gust SpeedMPH                   float64
PrecipitationIn                     float64
CloudCover                          float64
WindDirDegrees                      float64
Fog                                   uint8
Fog-Rain                              uint8
Normal                                uint8
Rain                                  uint8
Rain-Thunderstorm                     uint8
dtype: object
```

In [289]:

```
weather.head()
```

Out[289]:

	Date	Max TemperatureF	Mean TemperatureF	Min TemperatureF	Max Dew PointF	MeanDew PointF	Min DewpointF
0	2014-09-01	83.0	70.0	57.0	58.0	56.0	52.0
1	2014-09-02	72.0	66.0	60.0	58.0	57.0	55.0
2	2014-09-03	76.0	69.0	61.0	57.0	56.0	55.0
3	2014-09-04	74.0	68.0	61.0	57.0	57.0	56.0
4	2014-09-05	72.0	66.0	60.0	57.0	56.0	54.0

5 rows × 27 columns

In [290]:

```
num_trips.head()
```

Out[290]:

	Date	num_trips
0	2014-09-01	368
1	2014-09-02	1319
2	2014-09-03	1404
3	2014-09-04	1389
4	2014-09-05	1265

In [291]:

```
weather.head()
```

Out[291]:

	Date	Max TemperatureF	Mean TemperatureF	Min TemperatureF	Max Dew PointF	MeanDew PointF	Min DewpointF
0	2014-09-01	83.0	70.0	57.0	58.0	56.0	52.0
1	2014-09-02	72.0	66.0	60.0	58.0	57.0	55.0
2	2014-09-03	76.0	69.0	61.0	57.0	56.0	55.0
3	2014-09-04	74.0	68.0	61.0	57.0	57.0	56.0
4	2014-09-05	72.0	66.0	60.0	57.0	56.0	54.0

5 rows × 27 columns

In [415]:

```
test['Date'] = test['start_date']
test['num_trips'] = test['net_trips']
```

In [416]:

```
test = test.groupby('Date', as_index=False).sum()
```

In [417]:

```
test = test.drop(['start_hour','start_trips','net_trips'],1)
```

In [418]:

```
test = test.drop(['end_trips'],1)
```

In [419]:

```
test.head()
```

Out[419]:

	Date	2nd at Folsom	2nd at South Park	2nd at Townsend	5th at Howard	Adobe on Almaden	Arena Green / SAP Center	Beale at Market	Broadway St at Battery St	Br
0	2014-09-01	2	4	4	2	0	1	3	4	0
1	2014-09-02	15	12	14	11	1	2	15	12	0
2	2014-09-03	16	13	12	9	1	3	12	9	1
3	2014-09-04	10	13	16	7	2	2	14	9	0
4	2014-09-05	14	11	13	10	2	0	14	10	0

5 rows × 73 columns

In [420]:

```
train = test.merge(weather, on = test.Date)
```

In [421]:

```
train.head()
```

Out[421]:

	Date_x	2nd at Folsom	2nd at South Park	2nd at Townsend	5th at Howard	Adobe on Almaden	Arena Green / SAP Center	Beale at Market	Broadway St at Battery St	E
0	2014-09-01	2	4	4	2	0	1	3	4	0
1	2014-09-02	15	12	14	11	1	2	15	12	0
2	2014-09-03	16	13	12	9	1	3	12	9	1
3	2014-09-04	10	13	16	7	2	2	14	9	0
4	2014-09-05	14	11	13	10	2	0	14	10	0

5 rows × 100 columns

In [422]:

```
train['date'] = train['Date_x']
train.drop(['Date_y','Date_x'],1, inplace= True)
```

```
In [423]:
```

```
train.head()
```

```
Out[423]:
```

	2nd at Folsom	2nd at South Park	2nd at Townsend	5th at Howard	Adobe on Almaden	Arena Green / SAP Center	Beale at Market	Broadway St at Battery St	Broadwa at Mai
0	2	4	4	2	0	1	3	4	0
1	15	12	14	11	1	2	15	12	0
2	16	13	12	9	1	3	12	9	1
3	10	13	16	7	2	2	14	9	0
4	14	11	13	10	2	0	14	10	0

5 rows × 99 columns

Add Special Date Features

```
In [424]:
```

```
#Find all of the holidays during our time span
calendar = USFederalHolidayCalendar()
holidays = calendar.holidays(start=train.date.min(), end=train.date.max())
```

```
In [425]:
```

```
holidays
```

```
Out[425]:
```

```
DatetimeIndex(['2014-09-01', '2014-10-13', '2014-11-11', '2014-11-2
7',
               '2014-12-25', '2015-01-01', '2015-01-19', '2015-02-1
6',
               '2015-05-25', '2015-07-03'],
              dtype='datetime64[ns]', freq=None)
```

```
In [426]:
```

```
#Find all of the business days in our time span
us_bd = CustomBusinessDay(calendar=USFederalHolidayCalendar())
business_days = pd.DatetimeIndex(start=train.date.min(), end=train.date.max(), f
req=us_bd)
```

In [427]:

```
business_days
```

Out[427]:

```
DatetimeIndex(['2014-09-02', '2014-09-03', '2014-09-04', '2014-09-0
5',
               '2014-09-08', '2014-09-09', '2014-09-10', '2014-09-1
1',
               '2014-09-12', '2014-09-15',
               ...
               '2015-08-18', '2015-08-19', '2015-08-20', '2015-08-2
1',
               '2015-08-24', '2015-08-25', '2015-08-26', '2015-08-2
7',
               '2015-08-28', '2015-08-31'],
              dtype='datetime64[ns]', length=251, freq='C')
```

In [428]:

```
business_days = pd.to_datetime(business_days, format='%Y/%m/%d').date
holidays = pd.to_datetime(holidays, format='%Y/%m/%d').date
```

In [429]:

```
train['business_day'] = train.date.isin(business_days)
train['holiday'] = train.date.isin(holidays)
```

In [430]:

```
train.head()
```

Out[430]:

	2nd at Folsom	2nd at South Park	2nd at Townsend	5th at Howard	Adobe on Almaden	Arena Green / SAP Center	Beale at Market	Broadway St at Battery St	Broadwa at Mai
0	2	4	4	2	0	1	3	4	0
1	15	12	14	11	1	2	15	12	0
2	16	13	12	9	1	3	12	9	1
3	10	13	16	7	2	2	14	9	0
4	14	11	13	10	2	0	14	10	0

5 rows × 101 columns

In [431]:

```
#Convert True to 1 and False to 0
train.business_day = train.business_day.map(lambda x: 1 if x == True else 0)
train.holiday = train.holiday.map(lambda x: 1 if x == True else 0)
```

In [432]:

```
train['year'] = pd.to_datetime(train['date']).dt.year
train['month'] = pd.to_datetime(train['date']).dt.month
train['weekday'] = pd.to_datetime(train['date']).dt.weekday
```

In [433]:

```
labels = train.num_trips
train = train.drop(['num_trips', 'date'], 1)
```

Modeling approach

First split the data into training and test set

In [435]:

```
X_train, X_test, y_train, y_test = train_test_split(train, labels,
test_size=0.2, random_state = 2)
```

Linear Regression

In [444]:

```
from sklearn.linear_model import LinearRegression,Ridge,Lasso
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error
import warnings
pd.options.mode.chained_assignment = None
warnings.filterwarnings("ignore", category=DeprecationWarning)

# Initialize logistic regression model
lModel = LinearRegression()

# Train the model
lModel.fit(X = X_train,y = y_train)

# Make predictions
preds = lModel.predict(X= X_train)

print("RMSE Value For Linear Regression: ", mean_squared_error(y_train,
preds)**0.5)
```

```
RMSE Value For Linear Regression:  16.5663679122
```

Regularization Model - Ridge

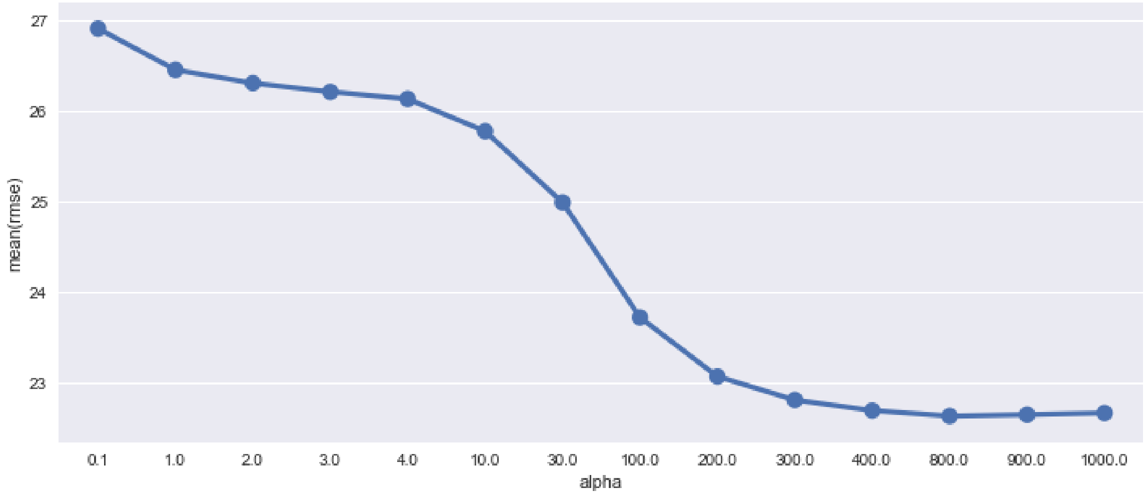
In [447]:

```
def rmse(y, y_):
    return mean_squared_error(y,y_)**0.5

ridge_m_ = Ridge()
ridge_params_ = { 'max_iter':[3000],'alpha':[0.1, 1, 2, 3, 4, 10,
30,100,200,300,400,800,900,1000]}
rmse_scorer = metrics.make_scorer(rmse, greater_is_better=False)
grid_ridge_m = GridSearchCV( ridge_m_,
                          ridge_params_,
                          scoring = rmse_scorer,
                          cv=5)
grid_ridge_m.fit( X_train, y_train )
preds = grid_ridge_m.predict(X= X_train)
print (grid_ridge_m.best_params_)
print ("RMSE Value For Ridge Regression: ",rmse(y_train,preds))

fig,ax= plt.subplots()
fig.set_size_inches(12,5)
df = pd.DataFrame(grid_ridge_m.grid_scores_)
df["alpha"] = df["parameters"].apply(lambda x:x["alpha"])
df["rmse"] = df["mean_validation_score"].apply(lambda x:-x)
sns.pointplot(data=df,x="alpha",y="rmse",ax=ax)
```

```
{'alpha': 800, 'max_iter': 3000}
RMSE Value For Ridge Regression:  19.3441269662
```

Out[447]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x121d1ad68>
```



Regularization Model - Lasso

In [450]:

```
lasso_m_ = Lasso()

alpha  = 1/np.array([0.1, 1, 2, 3, 4, 10, 30,100,200,300,400,800,900,1000])
lasso_params_ = { 'max_iter':[3000],'alpha':alpha}

grid_lasso_m = GridSearchCV( lasso_m_,lasso_params_,scoring = rmse_scorer,cv=5)
grid_lasso_m.fit( X_train, y_train )
preds = grid_lasso_m.predict(X= X_train)
print (grid_lasso_m.best_params_)
print ("RMSE Value For Lasso Regression: ",rmse(y_train,preds))

fig,ax= plt.subplots()
fig.set_size_inches(12,5)
df = pd.DataFrame(grid_lasso_m.grid_scores_)
df["alpha"] = df["parameters"].apply(lambda x:x["alpha"])
df["rmse"] = df["mean_validation_score"].apply(lambda x:-x)
sns.pointplot(data=df,x="alpha",y="rmse",ax=ax)
```

```
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
```

```
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
```

```
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
g data with very small alpha may cause precision problems.
  ConvergenceWarning)
/Users/VAN/anaconda/lib/python3.6/site-packages/sklearn/linear_mode
l/coordinate_descent.py:484: ConvergenceWarning: Objective did not c
onverge. You might want to increase the number of iterations. Fittin
```