Analysis of Days of Week based on Fremont Bicycle Data

Treating crossings each day as features to learn about the relationships between various days

In [33]:

```
%matplotlib inline
import pandas as pd
import matplotlib.pyplot as plt
import os
import sklearn
from sklearn.decomposition import PCA
from sklearn.mixture import GaussianMixture
import urllib.request
```

Get Data

Use local data or download it via DOI link from zendoo repository. Adapt the headers and calculate a total column.

In [34]:

```
FILENAME = '../data/Fremont_Bridge_Hourly_Bicycle_Counts_by_Month_October_201
URL = 'https://zenodo.org/record/2648564/files/Fremont_Bridge_Hourly_Bicycle_

def get_fremont_data(filename=FILENAME, url=URL, force_download=False):
    if force_download or not os.path.exists(filename):
        with urllib.request.urlopen(url) as response, open(filename, 'wb') as
            data = response.read() # a `bytes` object
            out_file.write(data)
    data = pd.read_csv(filename, index_col='Date', parse_dates=True)
    data.columns = ['West', 'East']
    data['Total'] = data['West'] + data['East']
    return data
```

In [35]:

```
data = get_fremont_data()
data.head()
```

Out[35]:

	West	East	Total
Date			
2019-03-31 23:00:00	6.0	10.0	16.0
2019-03-31 22:00:00	7.0	14.0	21.0
2019-03-31 21:00:00	18.0	15.0	33.0
2019-03-31 20:00:00	26.0	31.0	57.0
2019-03-31 19:00:00	30.0	58.0	88.0

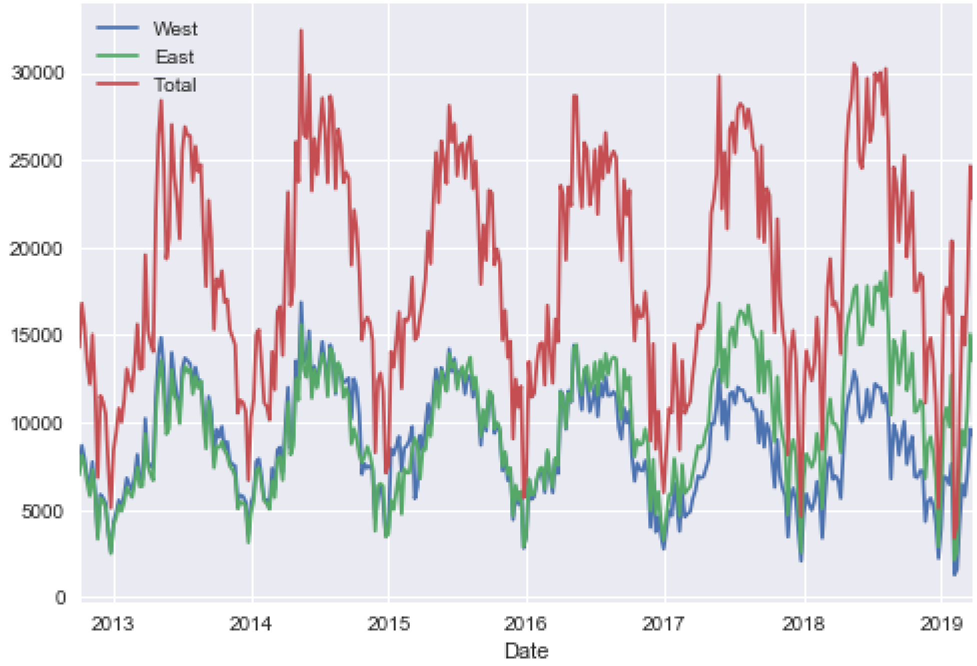
Plot weekly line graph to give a quick overview of the data

In [36]:

```
plt.style.use('seaborn')
data.resample('W').sum().plot()
```

Out[36]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x26781df90f0>
```



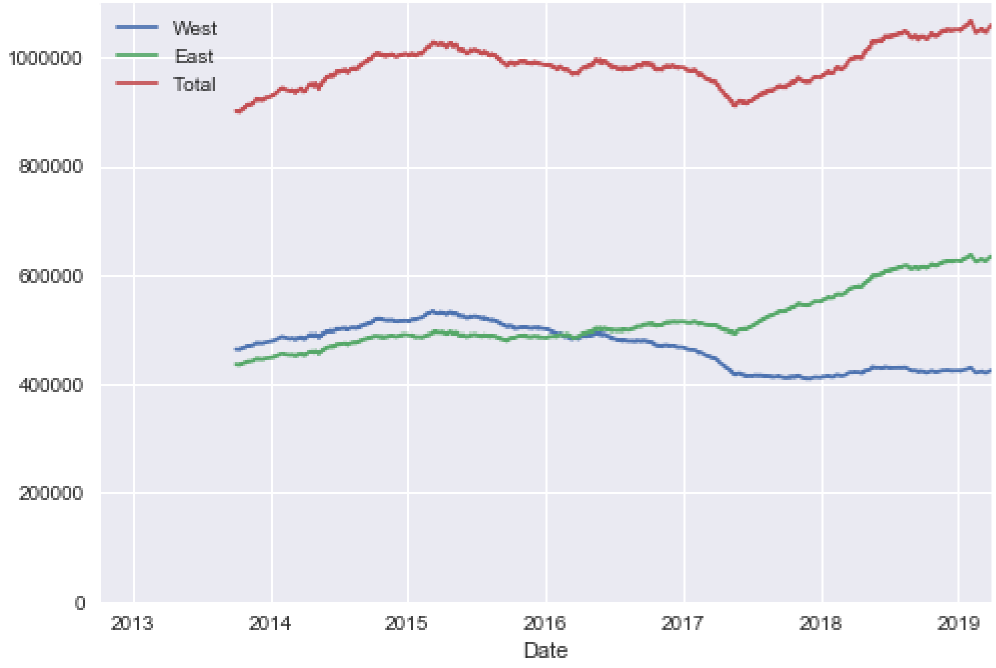
Plot daily line graph to see yearly usage

In [37]:

```
ax = data.resample('D').sum().rolling(365).sum().plot()
ax.set_ylim(0, None)
```

Out[37]:

```
(0, 1100992.6)
```



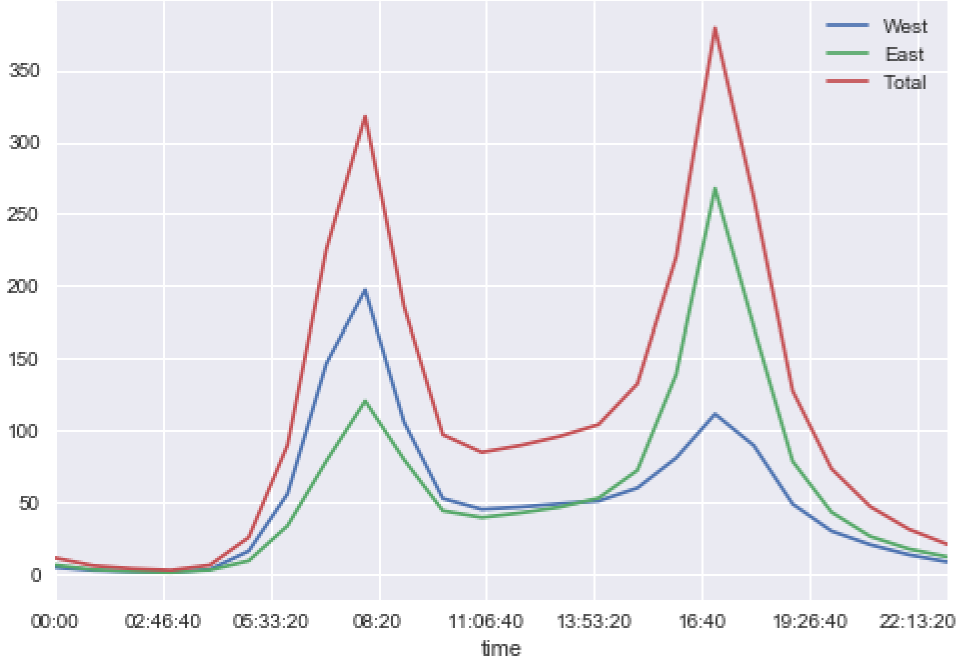
Group data by time, calc mean and plot to inspect the bridge usage per time of day

In [38]:

```
data.groupby(data.index.time).mean().plot()
```

Out[38]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x26781a1d630>
```



Pivot data and split data into date and time

In [39]:

```
pivoted = data.pivot_table('Total', index=data.index.time, columns=data.inde
pivoted.iloc[:5, :5]
```

Out[39]:

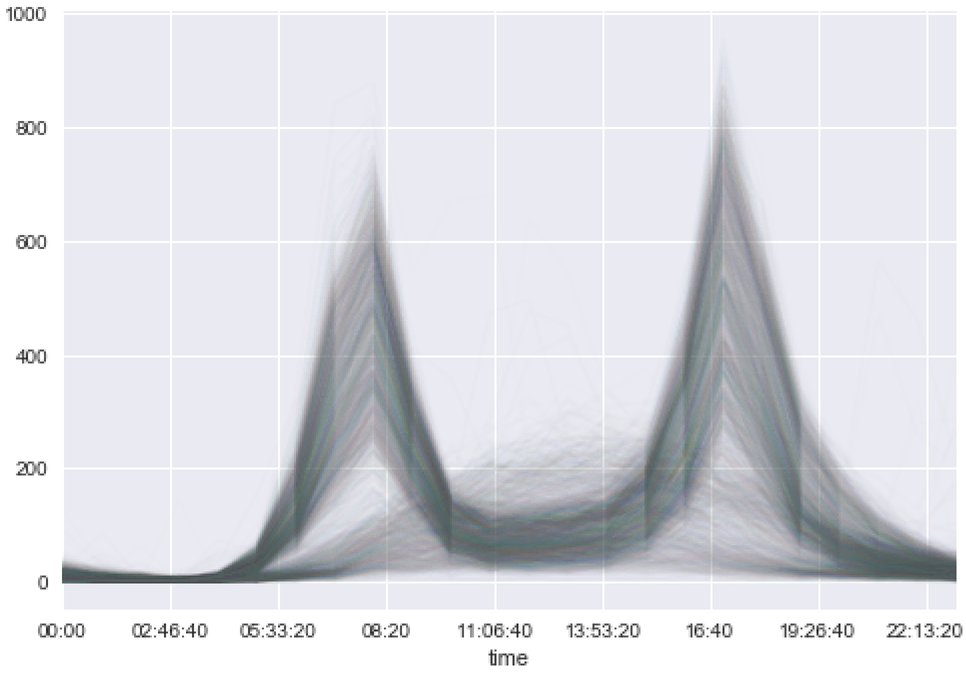
	2012-10-03	2012-10-04	2012-10-05	2012-10-06	2012-10-07
00:00:00	13.0	18.0	11.0	15.0	11.0
01:00:00	10.0	3.0	8.0	15.0	17.0
02:00:00	2.0	9.0	7.0	9.0	3.0
03:00:00	5.0	3.0	4.0	3.0	6.0
04:00:00	7.0	8.0	9.0	5.0	3.0

In [40]:

```
pivoted.plot(legend=False, alpha=0.01)
```

Out[40]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x26781a53080>
```



Principle Component Analysis

Use PCA to find patterns based on the usage per weekday

In [41]:

```
X = pivoted.fillna(0).T.values
X.shape
```

Out[41]:

```
(2371, 24)
```

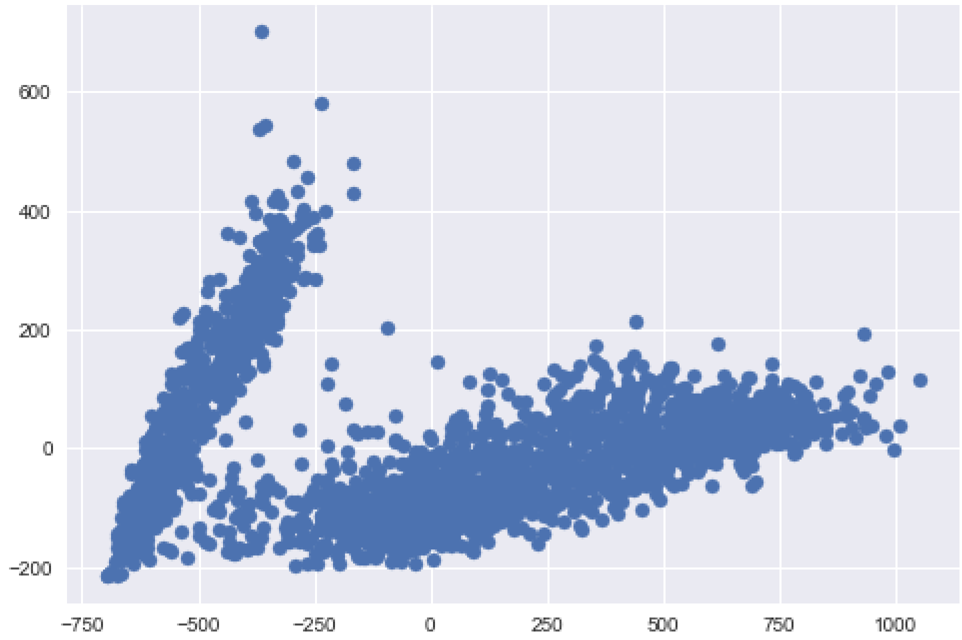
In [42]:
```
X2 = PCA(2, svd_solver='full').fit_transform(X)
```

In [43]:
```
X2.shape
```

Out[43]:
```
(2371, 2)
```

In [44]:
```
plt.scatter(X2[:, 0], X2[:, 1])
```

Out[44]:
```
<matplotlib.collections.PathCollection at 0x26784abae80>
```



Unsupervised Clustering

Further split the data and assign labels

In [45]:
```
qmm = GaussianMixture(2)
qmm.fit(X)
labels = qmm.predict(X)
labels
```

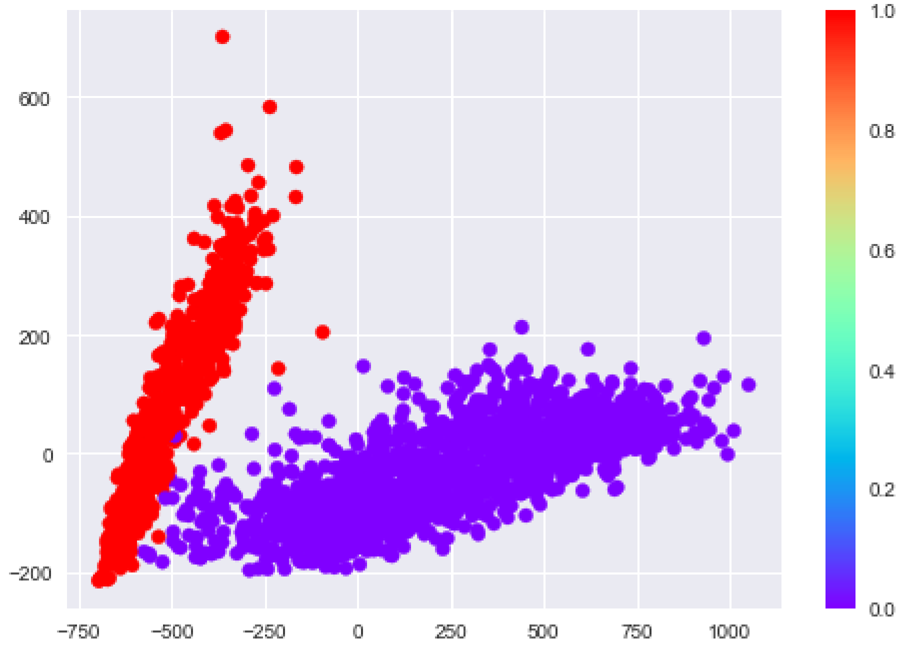
Out[45]:
```
array([0, 0, 0, ..., 0, 1, 1], dtype=int64)
```

In [46]:

```
plt.scatter(X2[:, 0], X2[:, 1], c=labels, cmap='rainbow')
plt.colorbar()
```

Out[46]:

```
<matplotlib.colorbar.Colorbar at 0x2678494bcf8>
```



Show usage patterns of each cluster

In [47]:

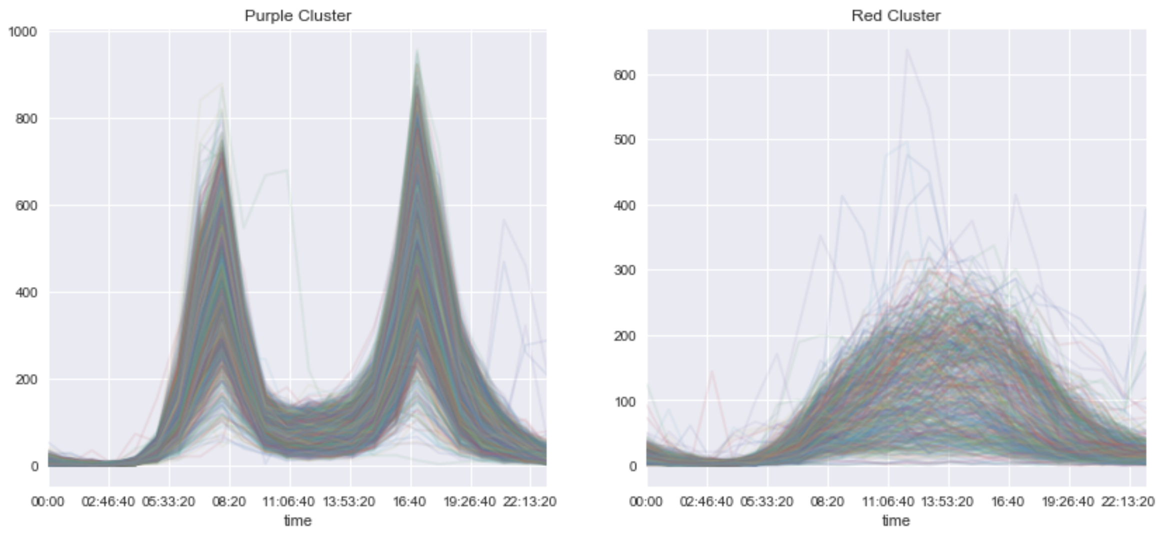
```
fig, ax = plt.subplots(1, 2, figsize=(14, 6))

pivoted.T[labels == 0].T.plot(legend=False, alpha=0.1, ax=ax[0])
pivoted.T[labels == 1].T.plot(legend=False, alpha=0.1, ax=ax[1])

ax[0].set_title('Purple Cluster')
ax[1].set_title('Red Cluster')
```

Out[47]:

```
Text(0.5, 1.0, 'Red Cluster')
```



Comparing with Day of Week

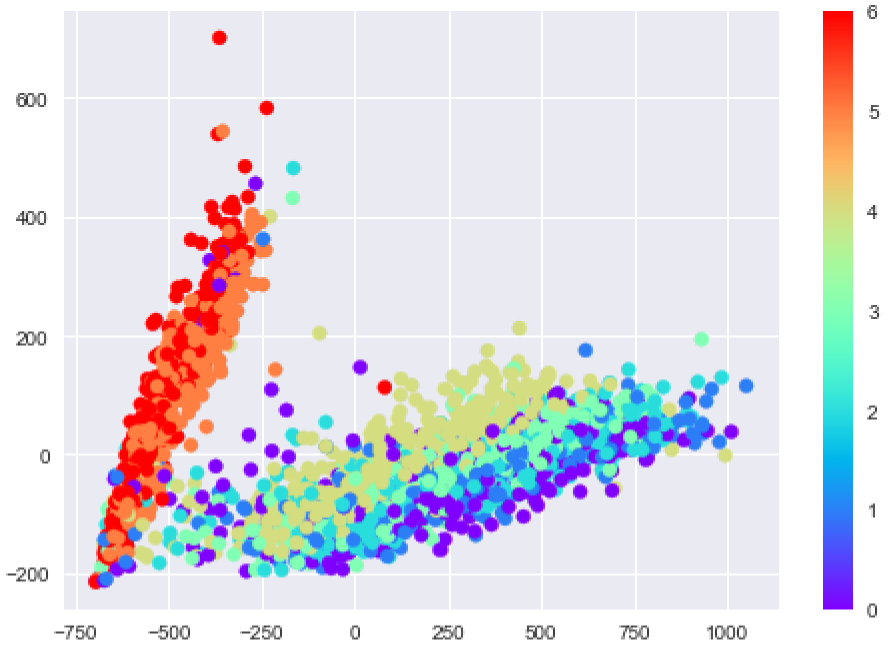
Assign colors according to weekdays to see if there is a clear separation of weekdays and weekends

In [48]:

```
dayofweek = pd.DatetimeIndex(pivoted.columns).dayofweek
plt.scatter(X2[:, 0], X2[:, 1], c=dayofweek, cmap='rainbow')
plt.colorbar()
```

Out[48]:

```
<matplotlib.colorbar.Colorbar at 0x267870b7080>
```



Analyzing Outliers

There is a separation in usage patterns between weekdays and weekends but with exceptions. The following points are weekdays with holiday-like pattern. One weekday is analyzed:

In [49]:

```
dates = pd.DatetimeIndex(pivoted.columns)
dates[(labels == 0) & (dayofweek < 5)]
```

Out[49]:

```
DatetimeIndex(['2012-10-03', '2012-10-04', '2012-10-05', '2012-10-08',
               '2012-10-09', '2012-10-10', '2012-10-11', '2012-10-12',
               '2012-10-15', '2012-10-16',
               ...
               '2019-03-18', '2019-03-19', '2019-03-20', '2019-03-21',
               '2019-03-22', '2019-03-25', '2019-03-26', '2019-03-27',
               '2019-03-28', '2019-03-29'],
              dtype='datetime64[ns]', length=1630, freq=None)
```

2017-01-02: New Year

2017-01-16: Martin Luther King day: national holiday but not all employers implemented it; demonstration with thousands of people in Seattle Thousands march (https://www.seattletimes.com/seattle-news/puget-sound/thousands-peacefully-march-rally-in-seattle-to-remember-civil-rights-leader-mlk-jr/)

2017-02-06 Thursday? Snow Storm (https://www.seattletimes.com/seattle-news/weather/weather-service-predicts-3-to-6-inches-of-snow-in-seattle-area/)

2017-05-29: Memorial day

2017-07-04: Independence day

2017-09-04: Labor day

2017-11-23: Thanksgiving

2017-11-24: Black Friday (not a holiday, but shopping event)

2017-12-25: Christmas

2017-12-26: no holiday