```
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
import numpy as np
```

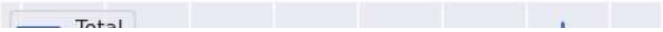
```
import pandas as pd
data = pd.read_csv('/content/Fremont_Bridge_Bicycle_Counter.csv', index_col='Date', parse_dates=True)
data.head()
```

	Fremont Bridge Sidewalks, south of N 34th St	Fremont Bridge Sidewalks, south of N 34th St Cyclist East Sidewalk	Fremont Bridge Sidewalks, south of N 34th St Cyclist West Sidewalk
Date			
2012-10-03 00:00:00	13.0	4.0	9.0
2012-10-03 01:00:00	10.0	4.0	6.0
2012-10-03 02:00:00	2.0	1.0	1.0
2012-10-03 03:00:00	5.0	2.0	3.0
2012-10-03 04:00:00	7.0	6.0	1.0

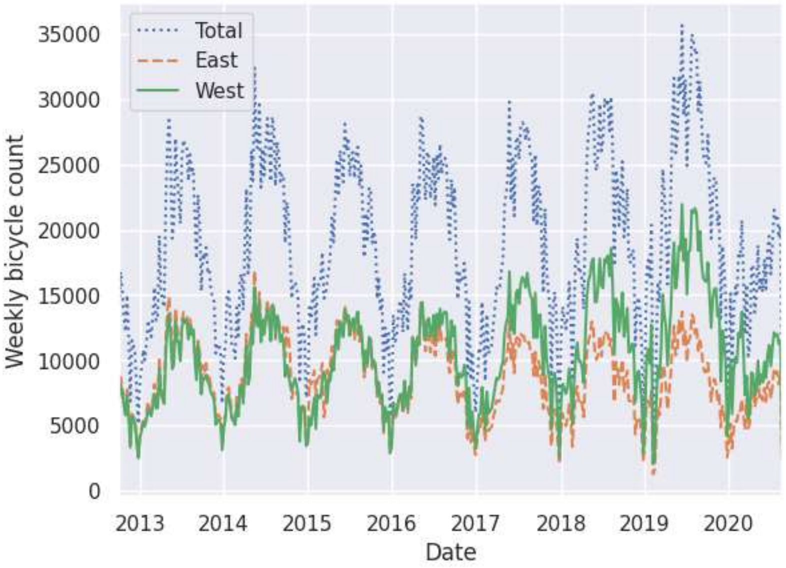
```
data.columns = ["Total","East", "West"]
data["Total"] = data["West"] + data["East"]
data.head()
```

	Total	East	West
Date			
2012-10-03 00:00:00	13.0	4.0	9.0
2012-10-03 01:00:00	10.0	4.0	6.0
2012-10-03 02:00:00	2.0	1.0	1.0
2012-10-03 03:00:00	5.0	2.0	3.0
2012-10-03 04:00:00	7.0	6.0	1.0

```
import matplotlib.pyplot as plt
import seaborn
seaborn.set()
data.plot()
plt.ylabel("Hourly Bicycle count")
plt.show()
```



```
weekly = data.resample("W").sum()
weekly.plot(style=[':', '--', '-'])
plt.ylabel('Weekly bicycle count')
plt.show()
```



```
counts = data
weather = pd.read_csv('/content/BicycleWeather.csv', index_col='DATE', parse_dates=True)
```

```
counts.head()
```

	Total	East	West
Date			
2012-10-03 00:00:00	13.0	4.0	9.0
2012-10-03 01:00:00	10.0	4.0	6.0
2012-10-03 02:00:00	2.0	1.0	1.0
2012-10-03 03:00:00	5.0	2.0	3.0
2012-10-03 04:00:00	7.0	6.0	1.0

```
weather.head()
```

	STATION	STATION_NAME	PRCP	SNWD	SNOW	TMAX	TMIN	AWND	WDF2	WDF5	...	WT17
DATE												
2012-01-01	GHCND:USW00024233	SEATTLE TACOMA INTERNATIONAL AIRPORT WA US	0	0	0	128	50	47	100	90	...	-9999
2012-01-02	GHCND:USW00024233	SEATTLE TACOMA INTERNATIONAL AIRPORT WA US	109	0	0	106	28	45	180	200	...	-9999
2012-01-03	GHCND:USW00024233	SEATTLE TACOMA INTERNATIONAL AIRPORT WA US	8	0	0	117	72	23	180	170	...	-9999
2012		SEATTLE TACOMA										

```
daily = counts.resample('d').sum()
daily['Total'] = daily.sum(axis=1)
daily = daily[['Total']] # remove other columns
```

```
daily.head()
```

	Total
Date	
2012-10-03	3521.0
2012-10-04	3475.0
2012-10-05	3148.0
2012-10-06	2006.0
2012-10-07	2142.0

```
days = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
for i in range(7):
    daily[days[i]] = (daily.index.dayofweek == i).astype(float)
```

```
daily.head()
```

	Total	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Date								
2012-10-03	3521.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
2012-10-04	3475.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
2012-10-05	3148.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
2012-10-06	2006.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
2012-10-07	2142.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

```
from pandas.tseries.holiday import USFederalHolidayCalendar
```

```
cal = USFederalHolidayCalendar()
holidays = cal.holidays('2012', '2016')
```

```
daily = daily.join(pd.Series(1, index=holidays, name='holiday'))
daily['holiday'].fillna(0, inplace=True)
```

```
pd.datetime(2000, 12, 21)
```

```
<ipython-input-20-017fdcc47849>:1: FutureWarning: The pandas.datetime class is deprecated and will be removed from pandas in a future ve
  pd.datetime(2000, 12, 21)
datetime.datetime(2000, 12, 21, 0, 0)
```

```
# temperatures are in 1/10 deg C; convert to C
weather['TMIN'] /= 10
weather['TMAX'] /= 10
weather['Temp (C)'] = 0.5 * (weather['TMIN'] + weather['TMAX'])

# precip is in 1/10 mm; convert to inches
weather['PRCP'] /= 254
weather['dry day'] = (weather['PRCP'] == 0).astype(int)
daily = daily.join(weather[['PRCP', 'Temp (C)', 'dry day']])
```

```
daily['annual'] = (daily.index - daily.index[0]).days / 365.
```

```
daily.head()
```

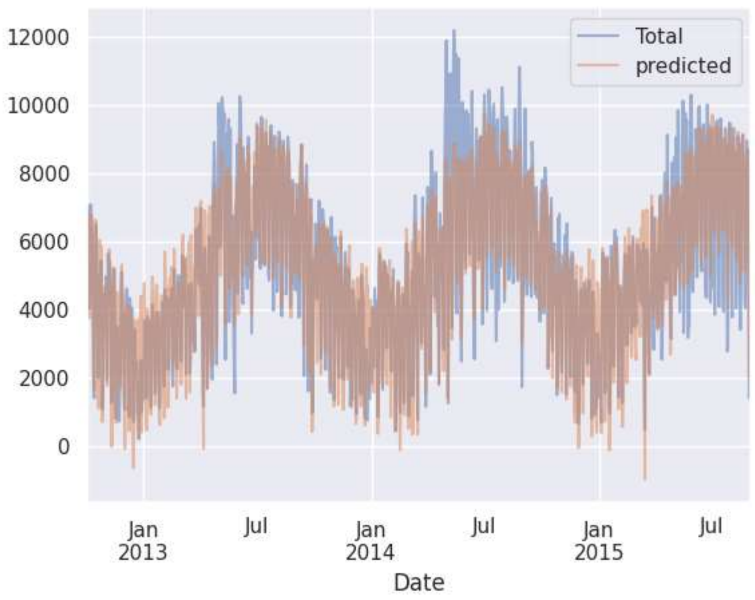
```
                                                                             Temp   dry
    Total  Mon  Tue  Wed  Thu  Fri  Sat  Sun  holiday  daylight_hrs  PRCP    (C)    day    annual
```

```
# Drop any rows with null values
daily.dropna(axis=0, how='any', inplace=True)

column_names = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun', 'holiday',
                'daylight_hrs', 'PRCP', 'dry day', 'Temp (C)', 'annual']
X = daily[column_names]
y = daily['Total']

from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=False)
model.fit(X, y)
daily['predicted'] = model.predict(X)
```

```
daily[['Total', 'predicted']].plot(alpha=0.5);
```



```
r2_score = model.score(X, y)
print("R-squared:", r2_score)
```

```
    R-squared: 0.8675358719950574
```