Data-X Spring 2019: Homework 04

Name:

Anish Saha

SID:

26071616

In this homework, you will do some exercises with plotting.

REMEMBER TO DISPLAY ALL OUTPUTS. If the question asks you to do something, make sure to print your results.

1.

Data:

Data Source: Data file is uploaded to bCourses and is named: **Energy.csv**

The dataset was created by Angeliki Xifara (Civil/Structural Engineer) and was processed by Athanasios Tsanas, Oxford Centre for Industrial and Applied Mathematics, University of Oxford, UK).

Data Description:

The dataset contains eight attributes of a building (or features, denoted by X1...X8) and response being the heating load on the building, y1.

- X1 Relative Compactness
- X2 Surface Area
- X3 Wall Area
- X4 Roof Area
- X5 Overall Height
- X6 Orientation
- X7 Glazing Area
- X8 Glazing Area Distribution
- y1 Heating Load

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Q1.1

Read the data file in python. Check if there are any NaN values, and print the results.

In [2]:
```
df = pd.read_csv("Energy.csv")
df.loc[df.isnull().any(axis=1)]
```

Out[2]:

	X1	X2	X3	X4	X5	X6	X7	X8	Y1

Q 1.2

Describe (using python function) data features in terms of type, distribution range (max and min), and mean values.

In [3]:
```
df.describe()
```

Out[3]:

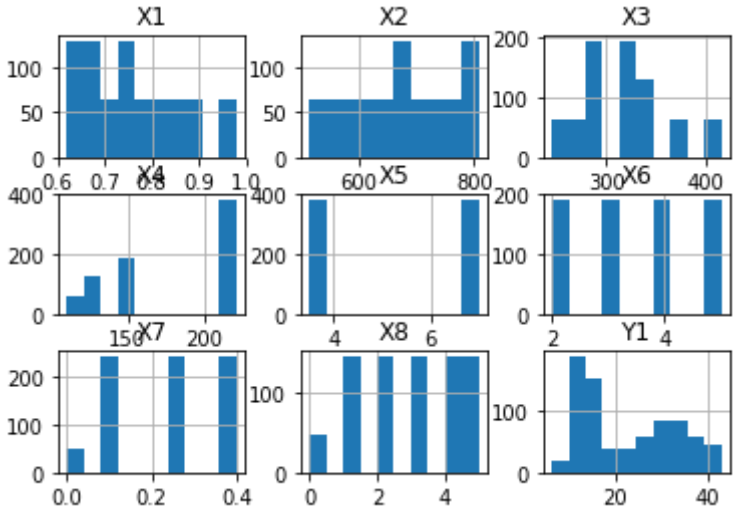
	X1	X2	X3	X4	X5	X6	X
count	768.000000	768.000000	768.000000	768.000000	768.00000	768.000000	768.00000
mean	0.764167	671.708333	318.500000	176.604167	5.25000	3.500000	0.234375
std	0.105777	88.086116	43.626481	45.165950	1.75114	1.118763	0.133221
min	0.620000	514.500000	245.000000	110.250000	3.50000	2.000000	0.000000
25%	0.682500	606.375000	294.000000	140.875000	3.50000	2.750000	0.100000
50%	0.750000	673.750000	318.500000	183.750000	5.25000	3.500000	0.250000
75%	0.830000	741.125000	343.000000	220.500000	7.00000	4.250000	0.400000
max	0.980000	808.500000	416.500000	220.500000	7.00000	5.000000	0.400000

Q 1.3

Plot feature distributions for all the attributes in the dataset (Hint - Histograms are one way to plot data distributions). This step should give you clues about data sufficiency.

In [4]:
```
df.hist()
```

Out[4]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x1155c6eb8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x11561a160>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x10ee7c470>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x10eea6780>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x10eecda90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x10eecdac8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x1156950b8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x11573e3c8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x1157676d8
>]],
      dtype=object)
```



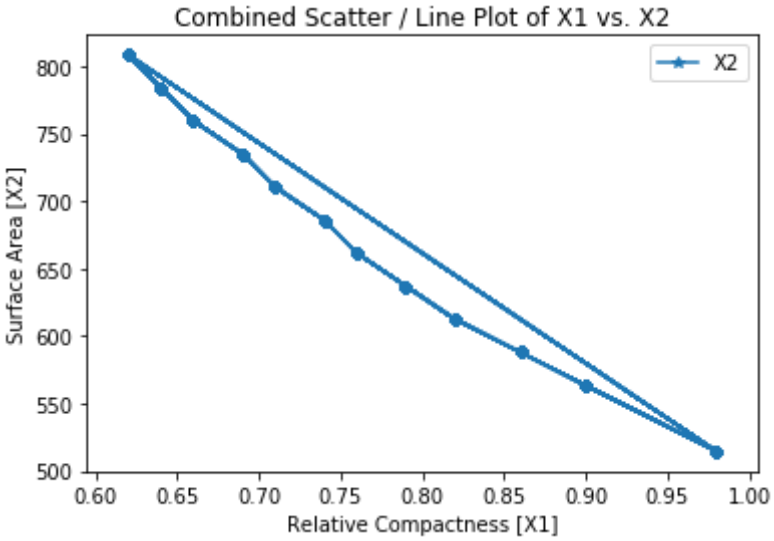
Q1.4

Create a combined line and scatter plot for attributes 'X1' and 'X2' with a marker (*). You can choose either of the attributes as x & y. Label your axes and give a title to your plot.

In [5]:
```
_, ax = plt.subplots()

df.plot.scatter(x='X1', y='X2', title="Combined Scatter / Line Plot of X
1 vs. X2", ax=ax)
df.plot.line(x='X1', y='X2', marker='*', ax=ax)
ax.set(xlabel='Relative Compactness [X1]', ylabel='Surface Area [X2]')
```

Out[5]:
```
[Text(0,0.5,'Surface Area [X2]'), Text(0.5,0,'Relative Compactness [X
1]')]
```



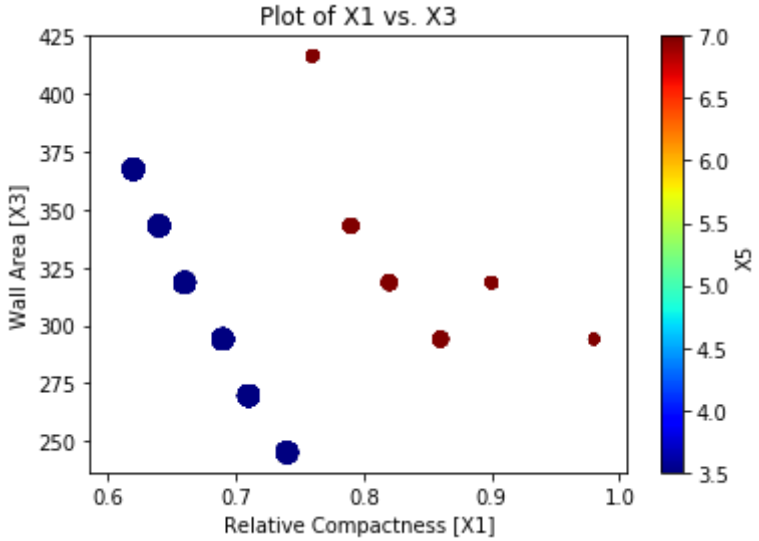
Q1.5

Create a scatter plot for how 'Wall Area' changes with 'Relative Compactness'. Give different colors for different 'Overall Height' and different bubble sizes by 'Roof Area'. Label the axes and give a title. Add a legend to your plot.

In [6]:
```
# source for function: https://github.com/pandas-dev/pandas/issues/8244
def convert_to_points(vals, size_range=(20, 100)):
    min_size, max_size = size_range
    val_range = vals.max() - vals.min()
    normalized_vals = (vals - vals.min()) / val_range
    point_sizes = (min_size + (normalized_vals * (max_size - min_size)))
    return point_sizes

_, ax = plt.subplots()
df.plot.scatter(x='X1', y='X3', c='X5', colormap='jet',
                s=convert_to_points(df['X4']), legend=True, ax=ax,
                title='Plot of X1 vs. X3')
ax.set(xlabel='Relative Compactness [X1]', ylabel='Wall Area [X3]')
```

Out[6]: [Text(0,0.5,'Wall Area [X3]'), Text(0.5,0,'Relative Compactness [X1]')]



2.

Q 2.1a.

Create a dataframe called `icecream` that has column `Flavor` with entries `Strawberry`, `Vanilla`, and `Chocolate` and another column with `Price` with entries `3.50`, `3.00`, and `4.25`. Print the dataframe.

In [7]:
```
icecream = pd.DataFrame(data={'Flavor': ['Strawberry', 'Vanilla',  'Choc
olate'],'Price': [3.50, 3.00, 4.25]})
icecream
```

Out[7]:

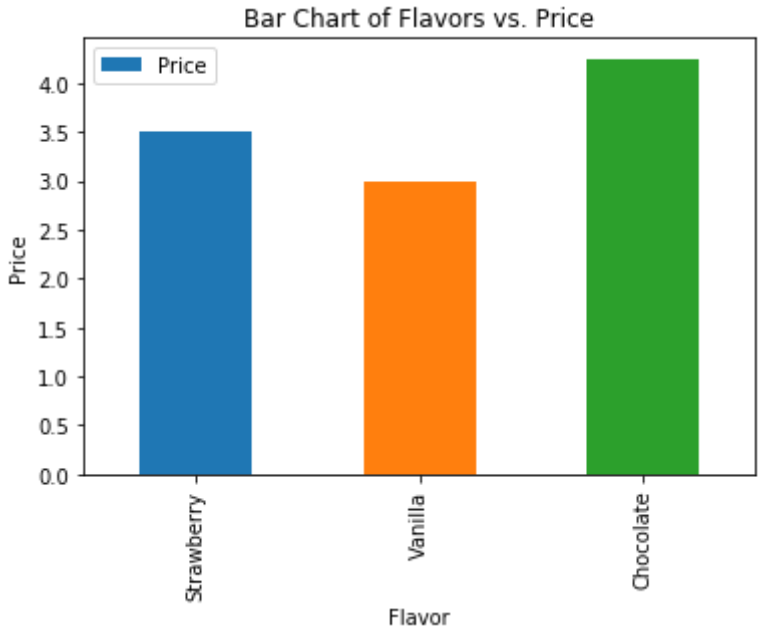
	Flavor	Price
0	Strawberry	3.50
1	Vanilla	3.00
2	Chocolate	4.25

Q 2.1b

Create a bar chart representing the three flavors and their associated prices. Label the axes and give a title.

In [8]:
```
_, ax = plt.subplots()
icecream.plot.bar(x='Flavor', y= 'Price', ax=ax, title='Bar Chart of Fla
vors vs. Price')
ax.set(xlabel='Flavor', ylabel='Price')
```

Out[8]:
```
[Text(0,0.5,'Price'), Text(0.5,0,'Flavor')]
```



Q 2.2

Create 9 random plots in a figure (Hint: There is a numpy function for generating random data).

The top three should be scatter plots (one with green dots, one with purple crosses, and one with blue triangles. The middle three graphs should be a line graph, a horizontal bar chart, and a histogram. The bottom three graphs should be trignometric functions (one sin, one cosine, one tangent). Keep in mind the range and conditions for the trignometric functions.

All these plots should be on the same figure and not 9 independent figures.

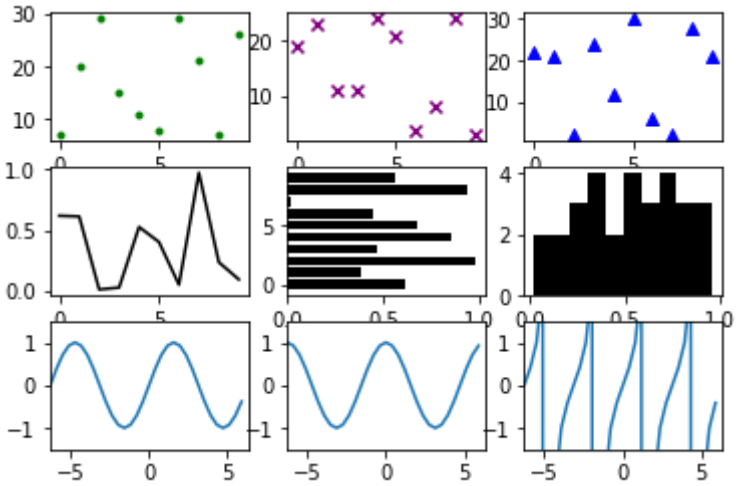
In [9]:
```
np.random.seed(42)
arr1 = np.random.random_integers(1, 30, 10)
arr2 = np.random.random_integers(1, 30, 10)
arr3 = np.random.random_integers(1, 30, 10)
arr4 = np.random.random_sample(10)
arr5 = np.random.random_sample(10)
arr6 = np.random.random_sample(30)
arr7 = np.sin(np.arange(-2*np.pi, 2*np.pi, np.pi/8))
arr8 = np.cos(np.arange(-2*np.pi, 2*np.pi, np.pi/8))
arr9 = np.tan(np.arange(-2*np.pi, 2*np.pi, np.pi/8))

plt.figure(1)
plt.subplot(331)
plt.scatter(np.arange(10), arr1, c='green', marker='.')
plt.subplot(332)
plt.scatter(np.arange(10), arr2, c='purple', marker='x')
plt.subplot(333)
plt.scatter(np.arange(10), arr3, c='blue', marker='^')
plt.subplot(334)
plt.plot(arr4, c='black')
plt.subplot(335)
plt.barh(np.arange(10), arr5, color='black')
plt.subplot(336)
plt.hist(arr6, color='black')
plt.subplot(337)
plt.xlim(-2*np.pi, 2*np.pi)
plt.ylim(-1.5, 1.5)
plt.plot(np.arange(-2*np.pi, 2*np.pi, np.pi/8), arr7)
plt.subplot(338)
plt.xlim(-2*np.pi, 2*np.pi)
plt.ylim(-1.5, 1.5)
plt.plot(np.arange(-2*np.pi, 2*np.pi, np.pi/8), arr8)
plt.subplot(339)
plt.xlim(-2*np.pi, 2*np.pi)
plt.ylim(-1.5, 1.5)
plt.plot(np.arange(-2*np.pi, 2*np.pi, np.pi/8), arr9)
```

```
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-pa
ckages/ipykernel_launcher.py:2: DeprecationWarning: This function is de
precated. Please call randint(1, 30 + 1) instead

/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-pa
ckages/ipykernel_launcher.py:3: DeprecationWarning: This function is de
precated. Please call randint(1, 30 + 1) instead
  This is separate from the ipykernel package so we can avoid doing imp
orts until
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-pa
ckages/ipykernel_launcher.py:4: DeprecationWarning: This function is de
precated. Please call randint(1, 30 + 1) instead
  after removing the cwd from sys.path.
```

Out[9]: `[<matplotlib.lines.Line2D at 0x115fbcc88>]`



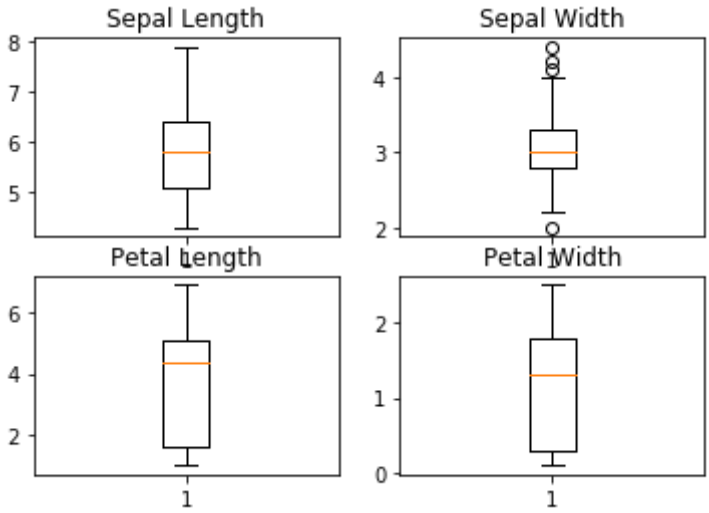
3.

Q 3.1

Load the 'Iris' dataset using seaborn. Create a box plot for the attributes 'sepal_length', sepal_width', 'petal_length' and 'petal_width' in the Iris dataset.

In [10]:
```
iris = sns.load_dataset('iris')
plt.figure(1)
plt.subplot(221)
plt.boxplot(iris['sepal_length'])
plt.title('Sepal Length')
plt.subplot(222)
plt.boxplot(iris['sepal_width'])
plt.title('Sepal Width')
plt.subplot(223)
plt.boxplot(iris['petal_length'])
plt.title('Petal Length')
plt.subplot(224)
plt.boxplot(iris['petal_width'])
plt.title('Petal Width')
```

Out[10]: Text(0.5,1,'Petal Width')



Q 3.2

In a few sentences explain what can you interpret from the above box plot.

The Sepal Length seems to have a median around 6, with most values falling in the interquartile range [5, 7].

The Sepal Width seems to have a median around 3, with several outliers that are labeled as empty circles.

The Petal Length seems to have a median around 4.5, with a skewed right distribution.

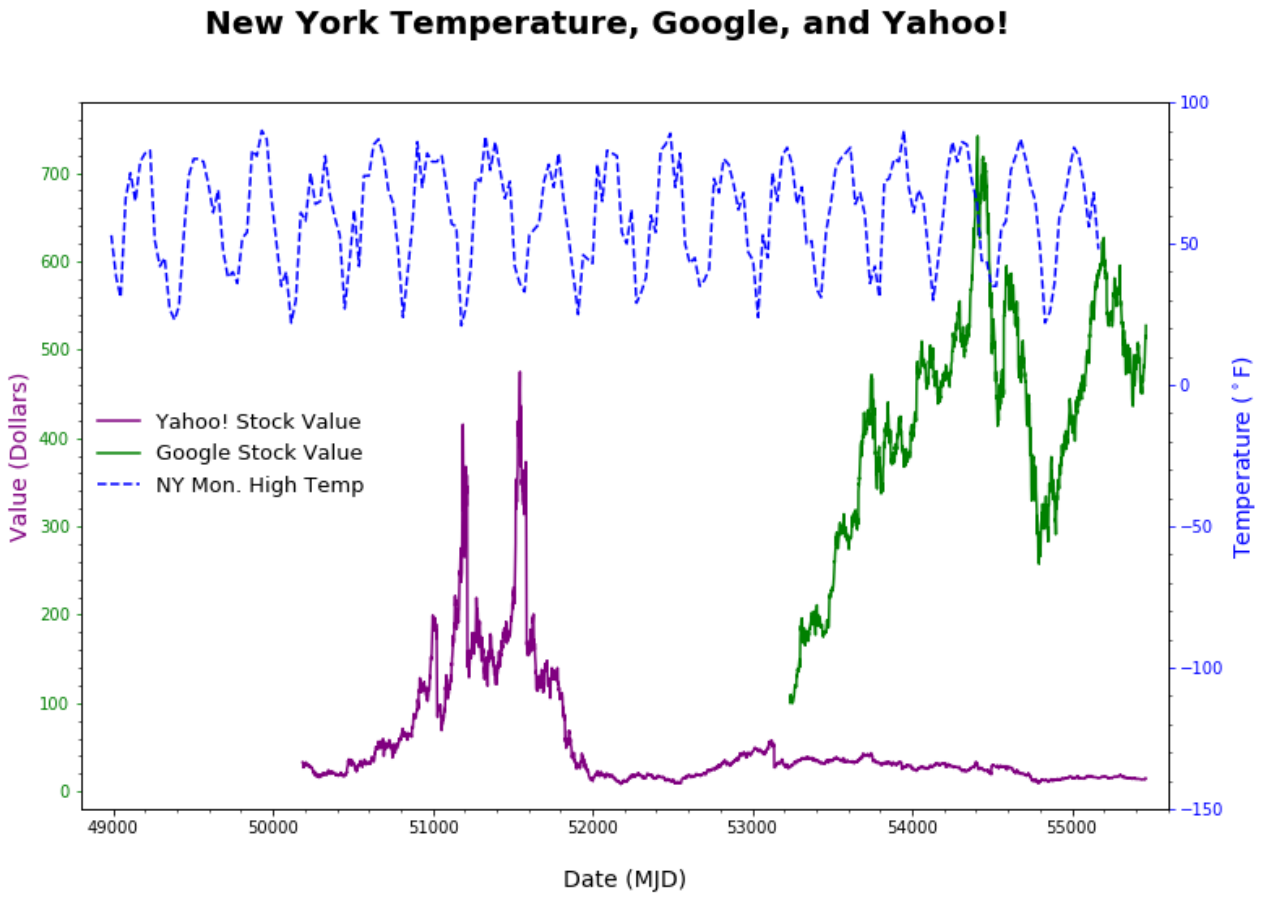
The Petal Width seems to have a median around 1.5, with most values falling in the interquartile range [0.25, 1.75].

Q 4.

The data files needed:

`google_data.txt, ny_temps.txt & yahoo_data.txt`

Use your knowledge with `Python, NumPy, pandas and matplotlib` to reproduce the plot below:



In [11]:
```
google = pd.read_table('google_data.txt')
ny = pd.read_table('ny_temps.txt')
yahoo = pd.read_table('yahoo_data.txt')

fig, ax1 = plt.subplots()
ax1.set_title('New York Temperature, Google, and Yahoo!', fontdict={'fon
tsize':'large','fontweight':600}, pad=25)
ax1.set_ylabel('Value (Dollars)', color='Purple')
ax1.tick_params(axis='y', colors='green')

ax2 = ax1.twinx()
ax2.set_ylim([-150, 100])
ax2.set_ylabel('Temperature (°F)', color='blue')
ax2.tick_params(axis='y', colors='blue')

google.plot('Modified Julian Date', 'Stock Value', color = 'green', ax=a
x1)
yahoo.plot('Modified Julian Date', 'Stock Value', color='purple', ax=ax1
)
ny.plot('Modified Julian Date', 'Max Temperature', color='blue', ax=ax2,
 linestyle='dashed')

ax1.set_xlabel('Date (MJD)')
ax2.set_xlabel('Date (MJD)')

ax1.legend(labels = ['Yahoo! Stock Value', 'Google Stock Value'], loc=6,
 fontsize='x-small', frameon=False)
ax2.legend(labels = ['NY Mon. High Temp'], loc=6, bbox_to_anchor=(0,0.42
5), fontsize='x-small', frameon=False)
```

Out[11]: <matplotlib.legend.Legend at 0x116380d68>