INTELLIGENT SYSTEMS LABORATORY

Lesson 6
Exploratory Data Analysis

Mathematics and Statistics for Data Science

Tapanan Yeophantong
Vincent Mary School of Science and Technology
Assumption University

Content

- Components of structured data
- Working with numpy & matplotlib libraries

What is Data?

- Collection of data objects and their attributes.
- An attribute is a property or characteristic of an object, e.g. eye color of a person, temperature, etc.
  - Attributes are also known as variables, fields or features.
- A collection of attributes describe an object.
  - Objects are also known as records, samples, or instances.

Types of Attributes

- Nominal, e.g. ID numbers, eye color, zip codes.
- Ordinal (Ranking), e.g., taste of potato chips on a scale from 1-10), grades, height in {tall, medium, short}.
- Interval, e.g. calendar dates, temperatures (C/F).
- Ratio, e.g. temperature in Kelvin, length, time, counts.

Attribute Properties

- Distinctness: = ≠
- Order: < >
- Addition: + -
- Multiplication: * /

Properties of Attribute Values

- Nominal :- distinctness
- Ordinal :- distinctness & order
- Interval :- distinctness, order & addition
- Ratio :- all 4 properties

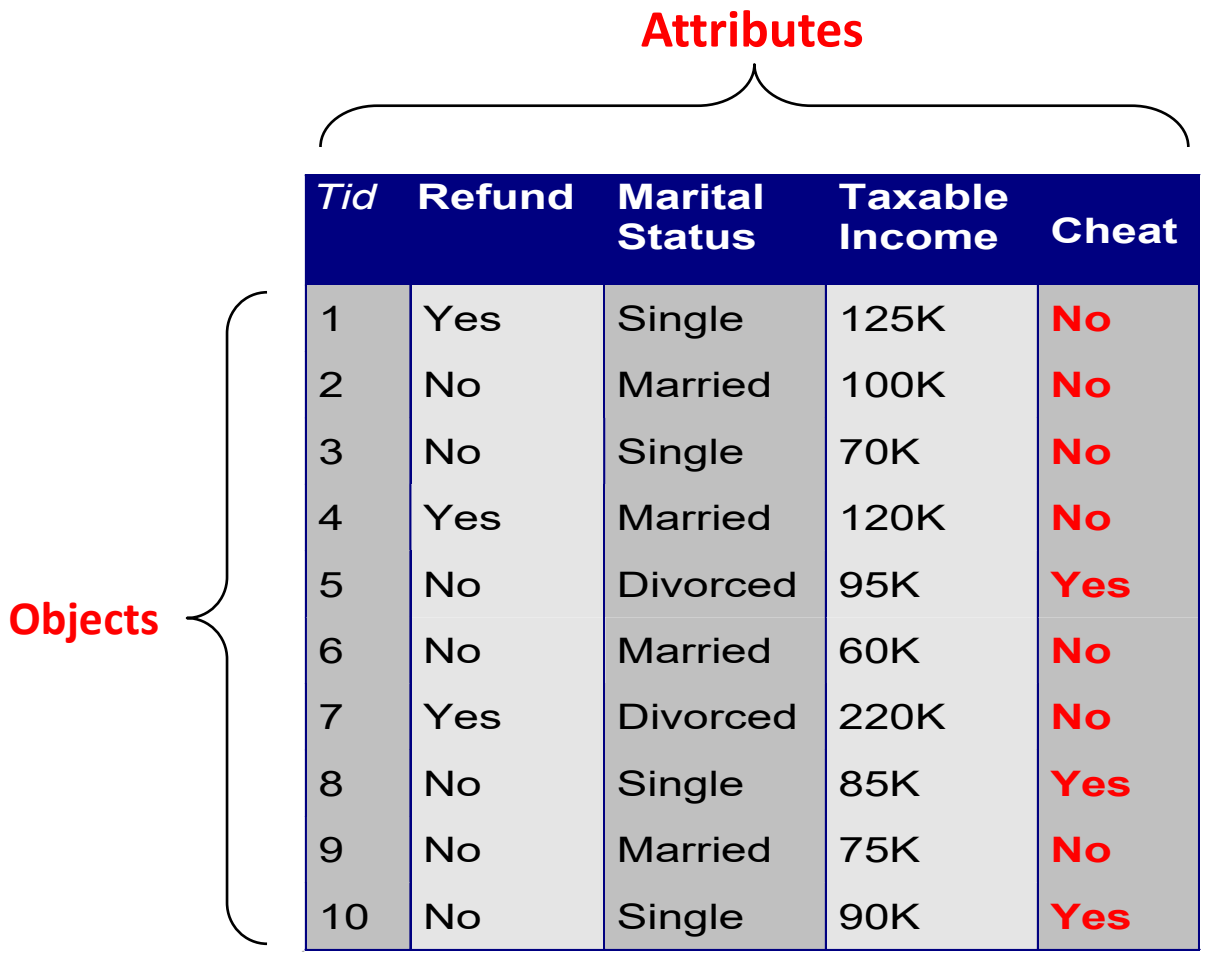
Discrete vs Continuous

- Discrete Attributes
  - Has only a finite or countably infinite set of values.
  - Examples: zip codes, counts, or set of words.
  - Often represented as integer variables.
- Continuous Attributes
  - Has real numbers as attribute values.
  - Examples: temperature, height, or weight.
  - Typically represented as floating points.

Types of Data Sets

- Record Data Sets
  - Data Matrix
  - Document Data
  - Transaction Data
- Graph Data Sets
  - World Wide Web
  - Molecular Structures
- Ordered Data Sets
  - Spatial Data
  - Temporal Data
  - Sequential Data
  - Genetic Sequence

Record Data



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Data Matrix

- Data objects as points in a multi-dimensional space, where each dimension represents a distinct attribute.

Projection of x Load	Projection of y load	Distance	Load	Thickness
10.23	5.27	15.22	2.7	1.2
12.65	6.25	16.22	2.2	1.1

- For instance, an image can be considered a data matrix.

Document Data

- Documents are transformed into a term vector.
- Each term is an attribute of the vector, and the value is the number of times the term occurs in the document.

Data = ['The', 'quick', 'brown', 'fox', 'jumps', 'over', ' the', 'lazy', 'dog']

	The	quick	brown	fox	jumps	over	lazy	dog
Data	2	1	1	1	1	1	1	1

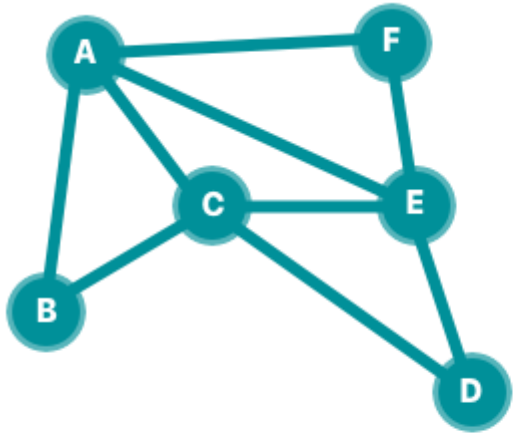
Transaction Data

- Each record (transaction) involves a set of items.
- For example, a set of products purchased by a customer during one shopping trip.

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Graph Data

- For examples, generic graph and HTML links.



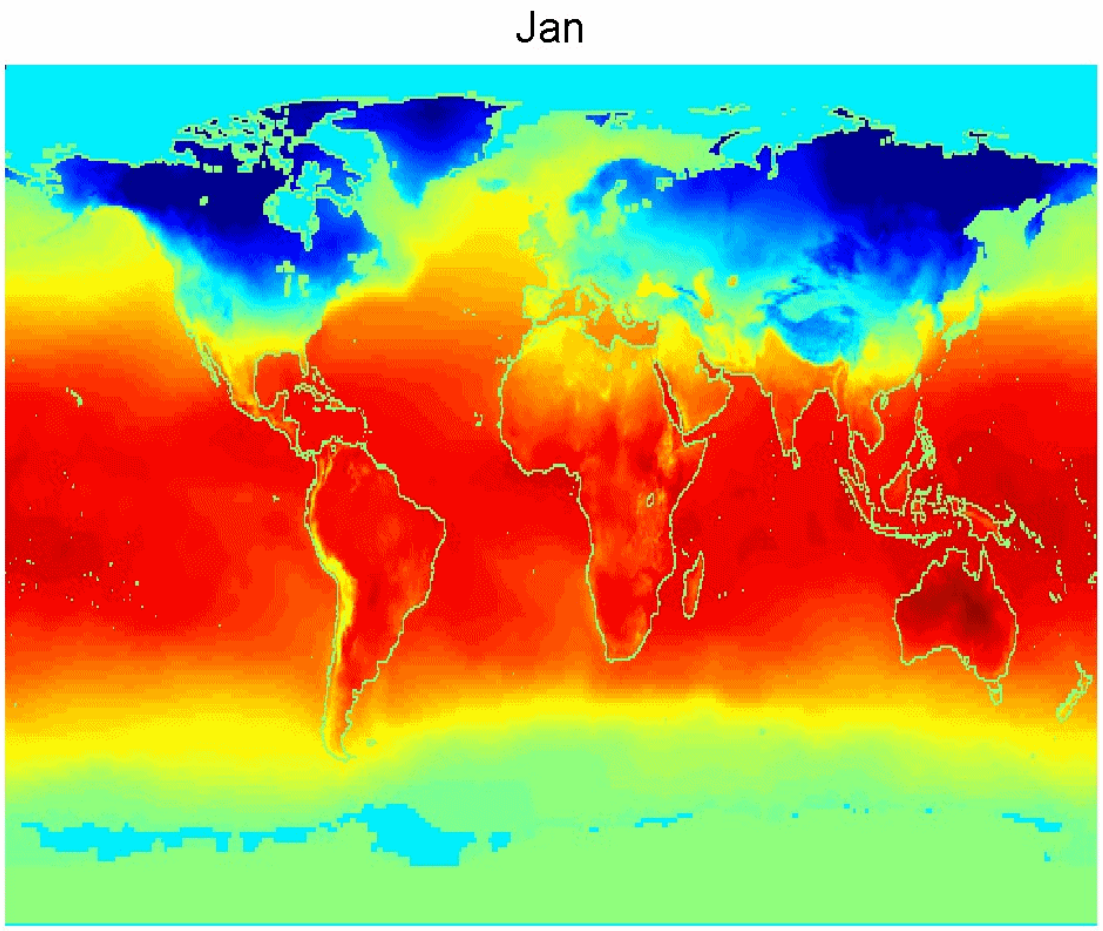
V: {A,B,C,D,E,F}

E: {AB,AC,AF,BC,CD,CE,DE,EF}

=

	A	B	C	D	E	F
A	0	1	1	0	1	1
B	1	0	1	0	0	0
C	1	1	0	1	1	0
D	0	0	1	0	1	0
E	1	0	1	1	0	0
F	1	0	0	0	1	0

Spatio-Temporal Data



Data Quality

- What kinds of data quality problems?
  - Noise and outliers
  - Missing values
  - Duplicate data
- How can we detect problems with the data?
- What can we do about these problems?

Overview of NumPy

- NumPy is a Python library used for working with arrays.
- Pronounce “num-pye” or “num-pee”.
- Faster than using built-in list (very important for ML).

Import NumPy

- To use NumPy:

```
import numpy
import numpy as np
```

- An array object in NumPy is called ndarray:

```
arr = np.array([1, 2, 3, 4, 5])
```

N-D Arrays

```
0D: arr0 = np.array(40)
1D: arr1 = np.array([1, 2, 3, 4, 5])
2D: arr2 = np.array([[1, 2, 3], [4, 5, 6]])
3D: arr3 = np.array([[[1, 2, 3], [4, 5, 6]],
                     [[7, 8, 9], [10, 11, 12]]])
```

Array Indexing

- You can access an array element by referring to its index:

```
print(arr1[0])
print(arr2[0, 1])
print(arr3[0, 1, 0])
```

- Use negative index to access array from the end:

```
print(arr1[-1])
```

Array Slicing

```
arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[1:5])      # ?
print(arr[4:])       # ?
print(arr[:4])       # ?
print(arr[-3:-1])    # ?
print(arr[1:5:2])    # ?
```

Array Shape & Reshape

- The shape is the number of elements in each dimension.

```
arr = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print(arr.shape)
```

- To reshape is to change the shape of the array:

```
newarr = arr.reshape(4,2)
print(newarr)
```

Array Join

```
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])

arr = np.concatenate((arr1, arr2))
arr = np.stack((arr1, arr2), axis=0)
arr = np.hstack((arr1, arr2))
arr = np.vstack((arr1, arr2))
arr = np.dstack((arr1, arr2))
```

Array Split

```
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])

newarr = np.array_split(arr, 3)
newarr = np.array_split(arr, 3, axis=0)
newarr = np.hsplit(arr, 3)
```

vsplit() and dsplit() are also available.

Array Search & Sort

- To search, use where():

```
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x1 = np.where(arr == 4)
x2 = np.where(arr % 2 == 0)
```

- To sort, use sort():

```
print(np.sort(arr))
```

Exercise 1

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

- Create a NumPy array for the matrix on the left.
- Compute sum of:
  - Each row
  - Each column
  - Each submatrices (by colours)
- Display the results.

Overview of Pandas

- Pandas is a library for working with data sets.
- It has the functions for analyzing, cleaning, exploring and manipulating data.
- We use Pandas a lot for working with various types of data.

Import Pandas

- To use Pandas:

```
import pandas
import pandas as pd
```

- There are two important data structures:
  - Pandas Series
  - Pandas DataFrame

Pandas Series

- Like a column in a table:

```
myvar = pd.Series([1, 7, 2])
myvar = pd.Series([1, 7, 2], index = ["x", "y", "z"])
```

- Series as keyword/value pairs:

```
calories = {"day1": 420, "day2": 380, "day3": 390}
myvar = pd.Series(calories)
```

Pandas DataFrames

- A DataFrame is a 2D data structure.

```
data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}
df = pd.DataFrame(data)
df = pd.DataFrame(data, index = ["d1", "d2", "d3"])
```

Getting Data from Files

- Reading CSV files:

```
df = pd.read_csv('datasets/data.csv')
print(df.to_string())
```

- Reading JSON files:

```
df = pd.read_json('datasets/data.json')
print(df.to_string())
```

Exercise 2

- Read data from “customer.csv”.
- Explore the data.

- Compute the statistics for each attribute.

Handling “Dirty” Data

```
df = pd.read_csv('datasets/dirtydata.csv')

df.dropna(inplace = True)
df.fillna(130, inplace = True)

x = df["Calories"].mean()
df["Calories"].fillna(x, inplace = True)
```

Handling Duplicates

- Use duplicated() method to detect duplicates:

```
print(df.duplicated())
```

- Use drop_duplicates() method to remove all duplicates:

```
df.drop_duplicates(inplace = True)
```

Finding Correlations

- Use df.corr() to find correlations among values.

	Duration	Pulse	Maxpulse	Calories
Duration	1.0000	-0.1554	0.0094	0.9227
Pulse	-0.1554	1.0000	0.7865	0.0251
Maxpulse	0.0094	0.7865	1.0000	0.2038
Calories	0.9227	0.0251	0.2038	1.0000

Exercise 3

- Read data from “customer.csv”.
- Explore the data.
- Compute the correlation.

Finding Null Records

```
import pandas as pd

df = pd.read_csv('datasets/dirtydata.csv’)
print(df[df.isnull().any(axis=1)])
```

	Duration	Date	Pulse	Maxpulse	Calories
18	45	'2020/12/18'	90	112	NaN
22	45	NaN	100	119	282.0
28	60	'2020/12/28'	103	132	NaN

Locating Rows & Columns

Syntax: *df.loc[rows,cols]*

```
df.loc[0]          # returns row 0
df.loc[0:5]        # returns rows 0-5
df.loc[[0,5]]      # returns rows 0 and 5

df.loc[:, "Calories"])                          # returns ?
df.loc[[0,5], "Duration":"Maxpulse"])           # returns ?
df.loc[0:5, ["Duration", "Maxpulse"]]           # returns ?
```

Writing to Files

```
df = pd.read_csv('datasets/data.csv')

cf = df.loc[:, ['Duration', 'Calories']]
cf.to_csv('datasets/newdata.csv', index=False)
```

- You can also write DataFrames to JSON (to_json()), Excel (to_excel()) and a lot more!

DataFrames to NumPy Arrays

```
df = pd.read_csv('datasets/data.csv')

cf = df.loc[0:5, ['Duration', 'Calories']]
arr = cf.to_numpy()
```

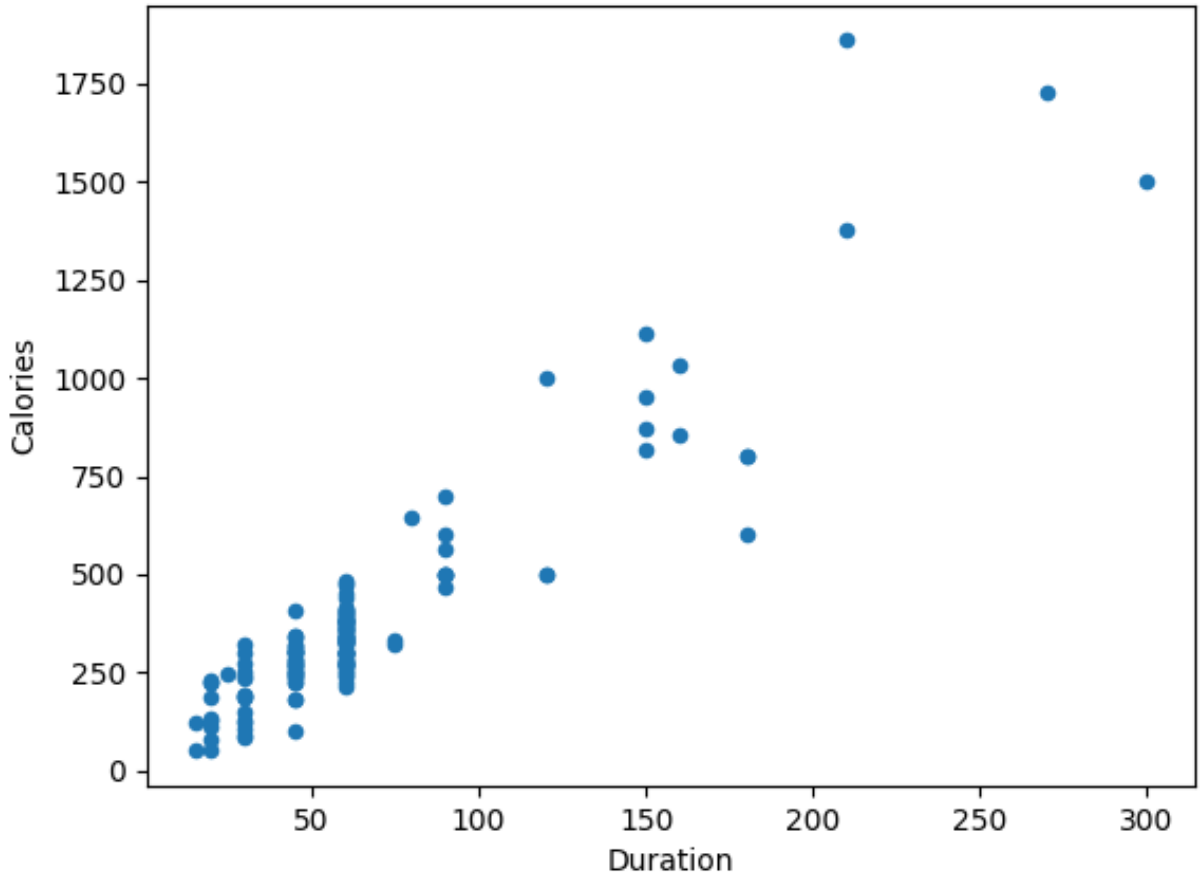
```
[[ 60.  409.1]
 [ 60.  479. ]
 [ 60.  340. ]
 [ 45.  282.4]
 [ 45.  406. ]
 [ 60.  300. ]]
```

DataFrame.plot

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('datasets/data.csv')
df.plot(kind='scatter', x='Duration', y='Calories')
plt.show()

# Can also use:
df.plot.scatter(x='Duration', y='Calories’)
```

Note: pip install matplotlib first.

Scatterplot (calories x duration)



Other Plots

- Line (‘line’, default)
- Vertical Bar (‘bar’) and Horizontal Bar (‘barh’)
- Histogram (‘hist’)
- Boxplot (‘box’)
- Kernel density (‘kde’ or ‘density’)
- Area (‘area’)
- Pie (‘pie’)
- Hexagonal binning plot (‘hexbin’)

Overview of Matplotlib

- Matplotlib is a low-level graph plotting library in Python.
- To use, you must install:

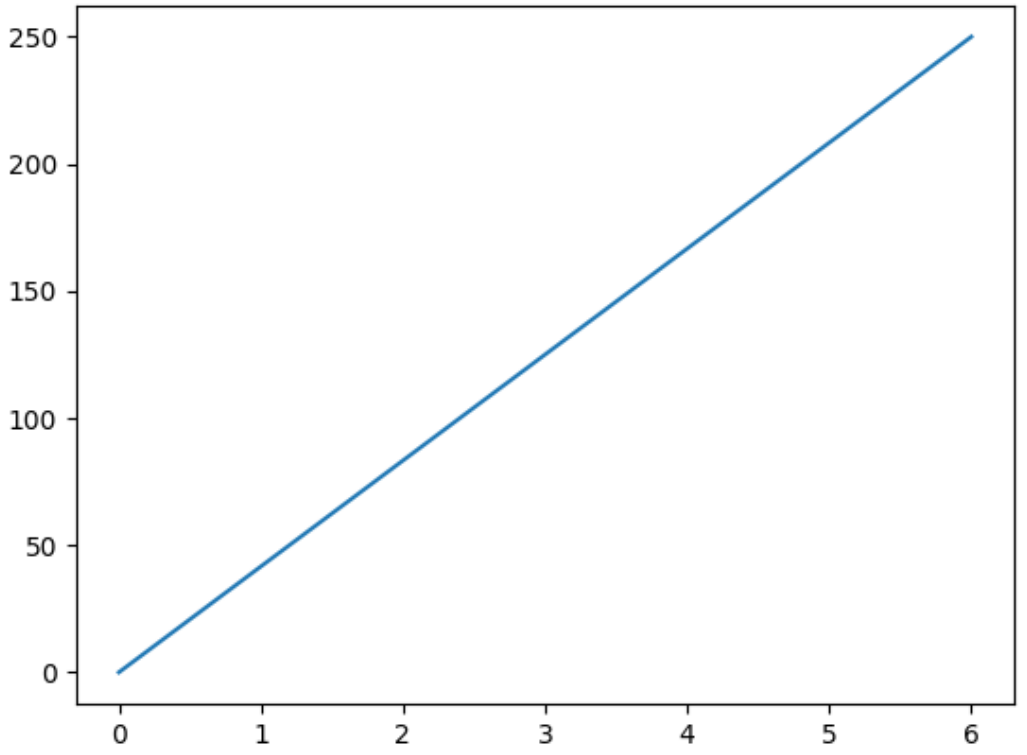
  pip install matplotlib
- Then, add the import module statement:

  import matplotlib

  import matplotlib.pyplot as plt

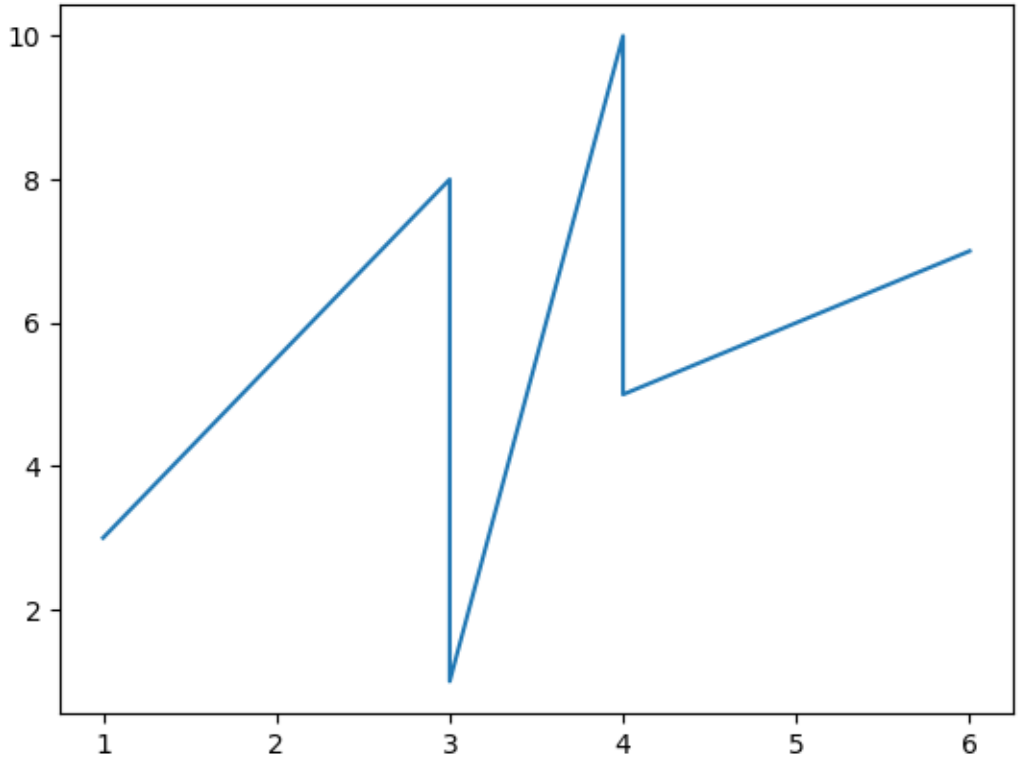
A Simple ‘Line’ Plot

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array([0, 6])
y = np.array([0, 250])

plt.plot(x, y)
plt.show()
```



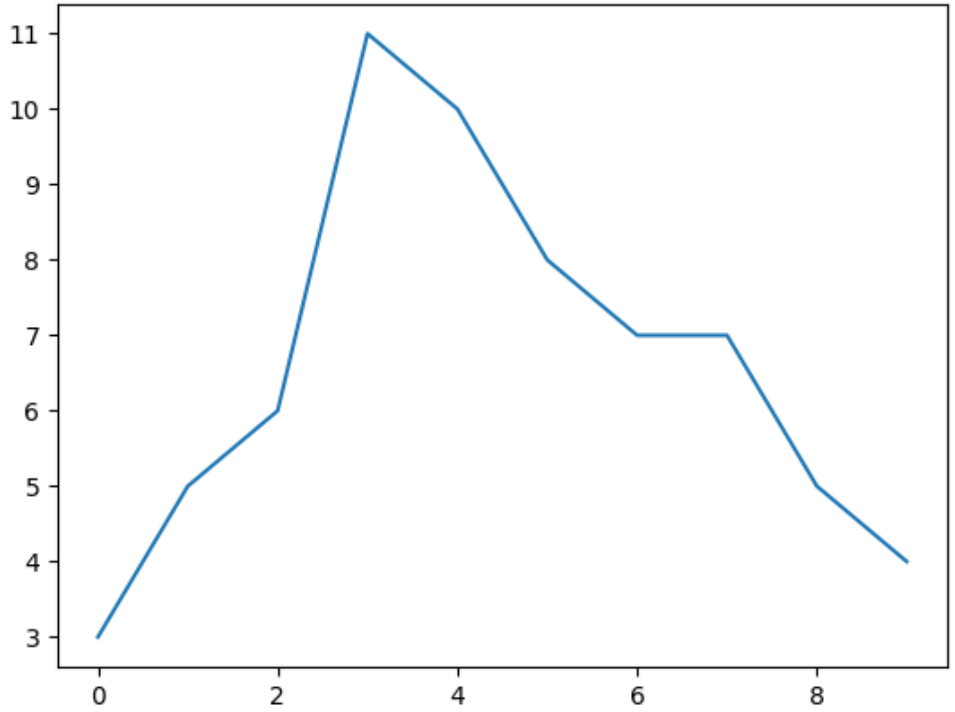
Plotting Multiple Points

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array([1, 3, 3, 4, 4, 6])
y = np.array([3, 8, 1, 10, 5, 7]

plt.plot(x, y)
plt.show()
```



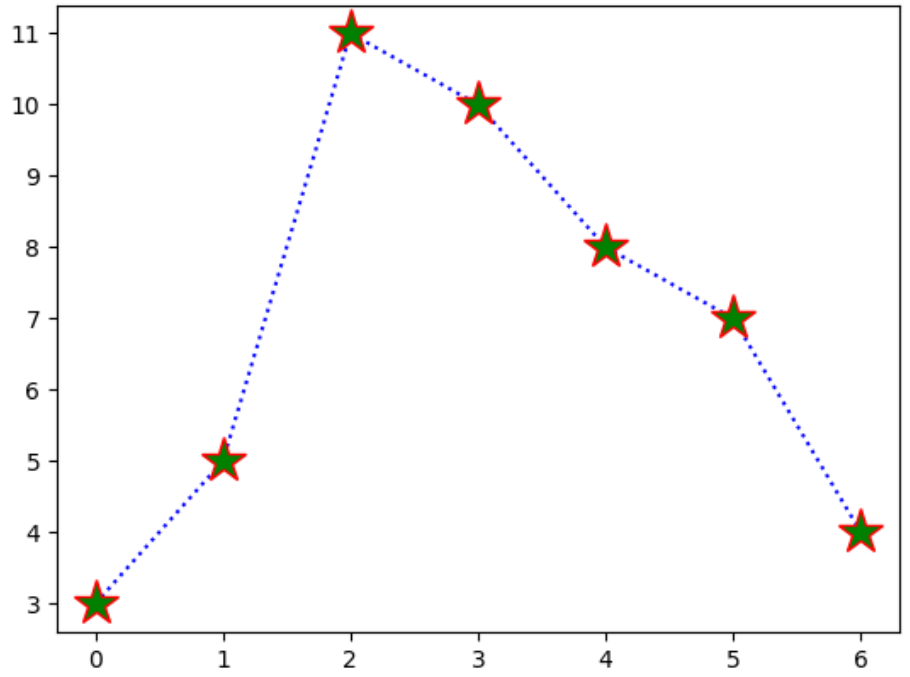
Default Domains (X)

```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([3, 5, 6, 11, 10, 8, 7,
              7, 5, 4])
plt.plot(y)
plt.show()
```



Markers, Lines & Colours

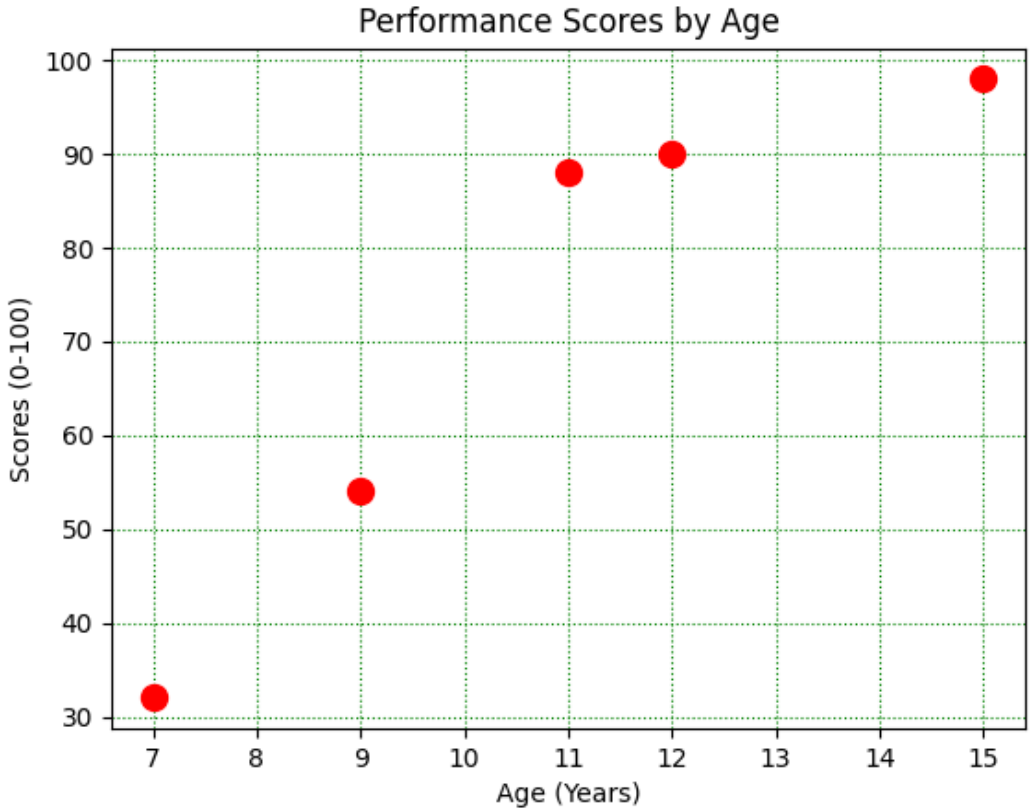
```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([3, 5, 11, 10, 8, 7, 4])

plt.plot(y, '*:b’, ms=20, mec='r’, mfc='g')
plt.show()
```



For marker/line/colour reference, visit:
https://www.w3schools.com/python/matplotlib_markers.asp

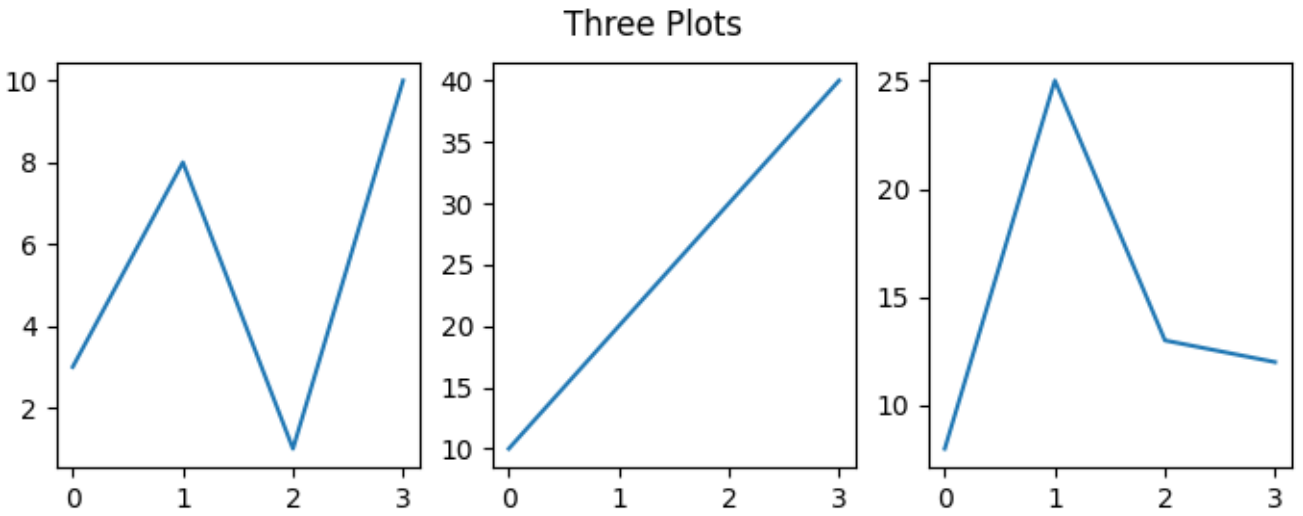
Labels, Titles & Grids

```
x = np.array([7, 9, 11, 12, 15])
y = np.array([32, 54, 88, 90, 98])

plt.title("Performance Scores by Age")
plt.xlabel("Age (Years)")
plt.ylabel("Scores (0-100)")
plt.grid(color='green', linestyle=':')

plt.plot(x, y, 'o r', ms=10)
plt.show()
```



Subplots

```
plt.suptitle("Three Plots")
y1 = np.array([3, 8, 1, 10])
plt.subplot(1, 3, 1)
plt.plot(y1)
y2 = np.array([10, 20, 30, 40])
plt.subplot(1, 3, 2)
plt.plot(y2)
y3 = np.array([8, 25, 13, 12])
plt.subplot(1, 3, 3)
plt.plot(y3)
plt.show()
```

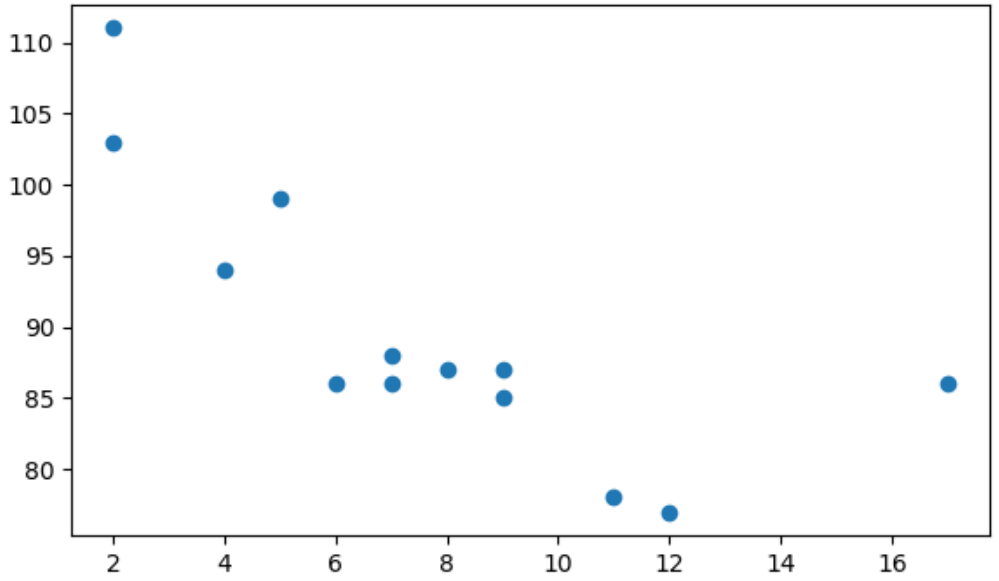


Exercise 4

- Read data from “customer.csv”.
- Extract age and spending score.
- Remove duplicates.
- Plot the data using Matplotlib.

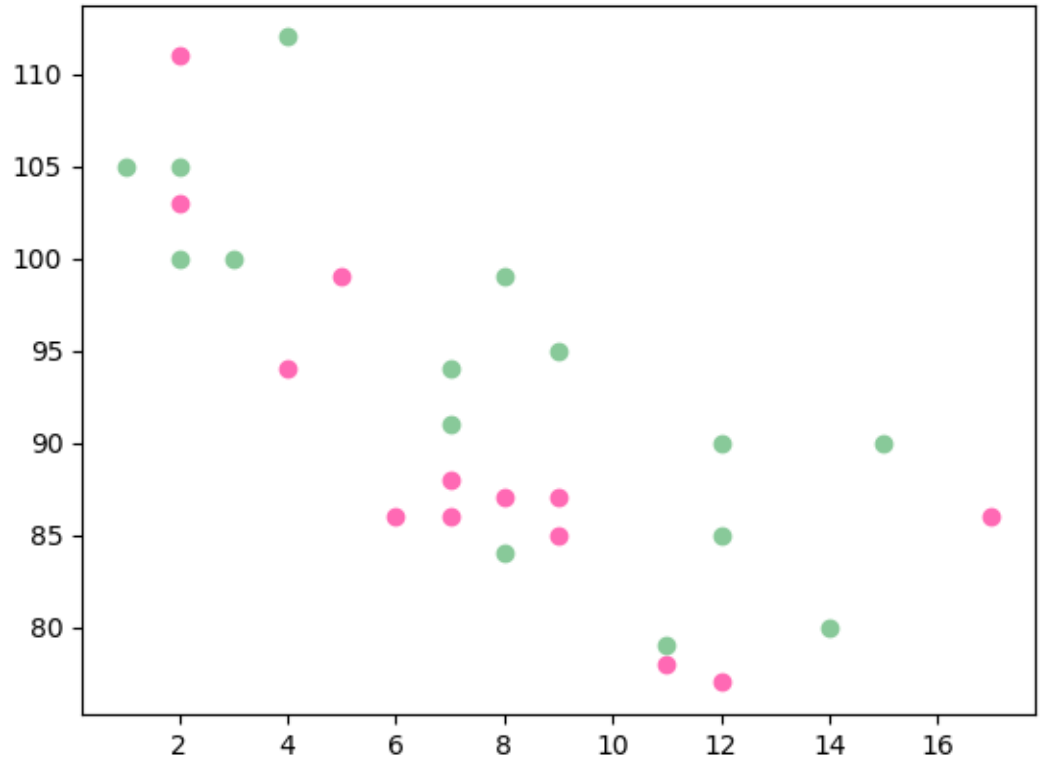
Scatter Plots

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,
              12,9,6])
y = np.array([99,86,87,88,111,86,103,
              87,94,78,77,85,86])

plt.scatter(x, y)
plt.show()
```



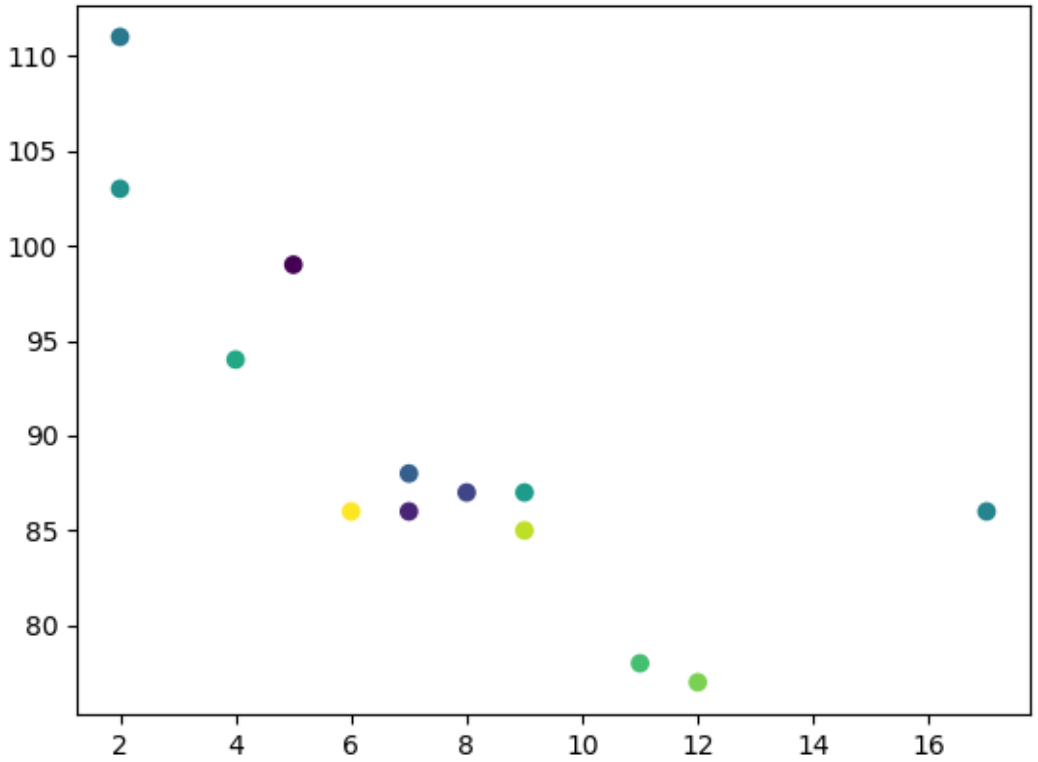
Comparing Plots

```
import matplotlib.pyplot as plt
import numpy as np
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,
                    77,85,86])
plt.scatter(x, y, color = 'hotpink')
x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])
y = np.array([100,105,84,105,90,99,90,95,94,100,
                    79,112,91,80,85])
plt.scatter(x, y, color = '#88c999')
plt.show()
```



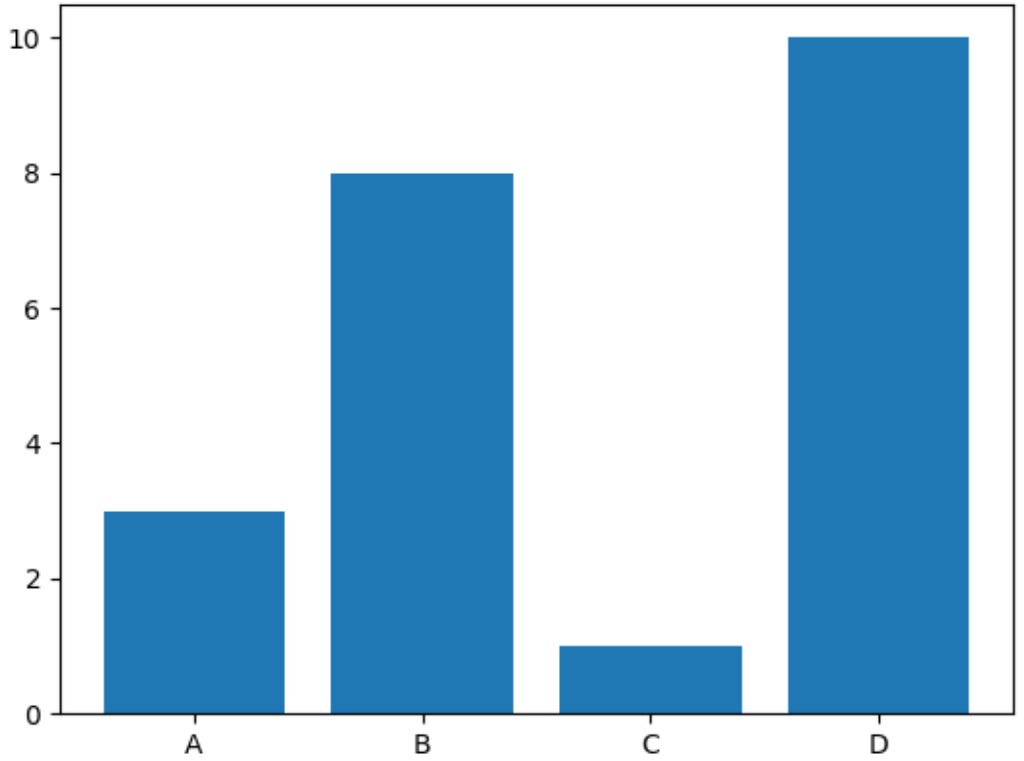
Using ColorMap

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,
                    77,85,86])
colors = np.array([0, 10, 20, 30, 40, 45, 50,
                    55, 60, 70, 80, 90, 100])

plt.scatter(x, y, c=colors, cmap='viridis')
plt.show()
```



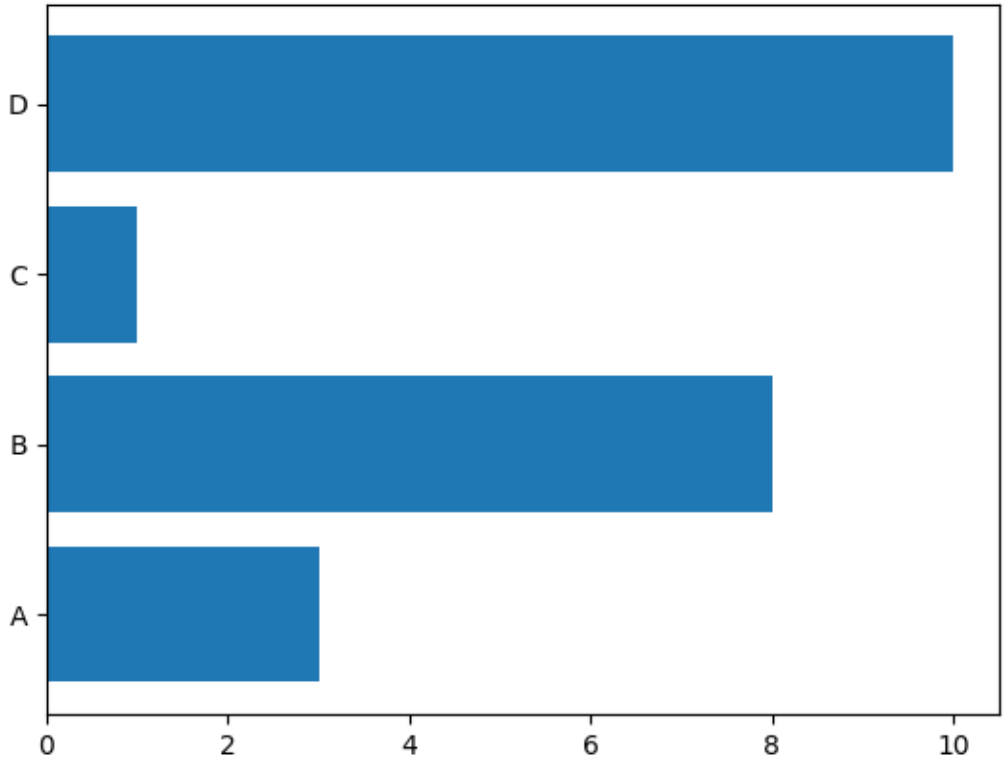
Bar Graphs

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.bar(x,y)
plt.show()
```



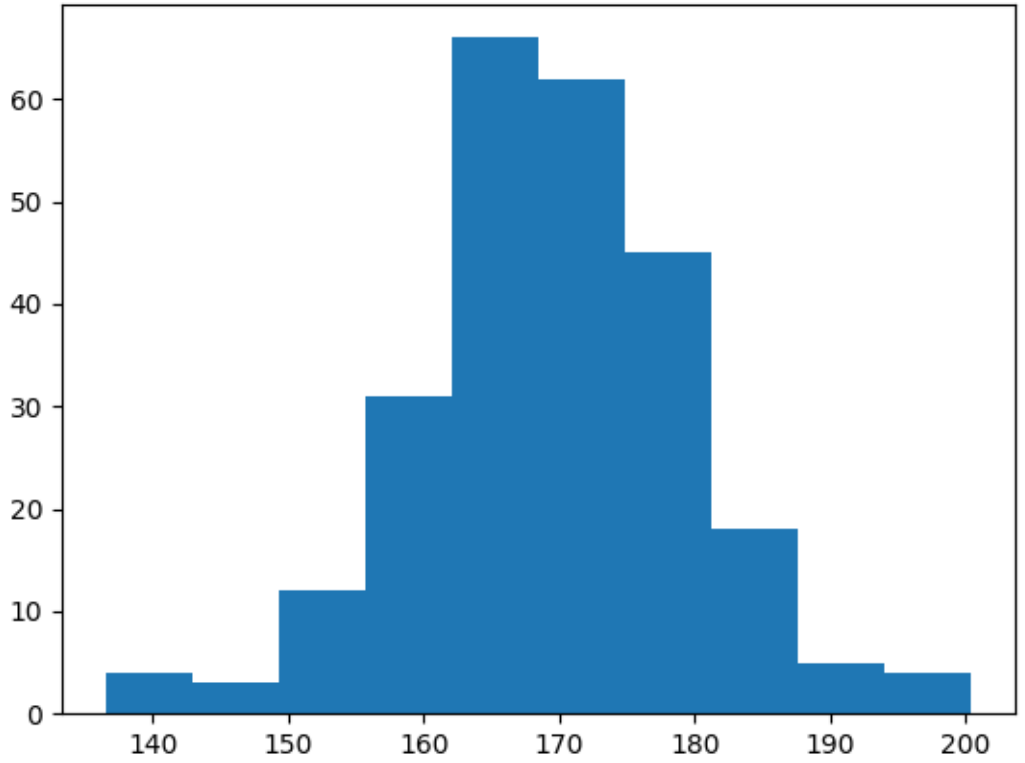
Horizontal Bars

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.barh(x, y)
plt.show()
```



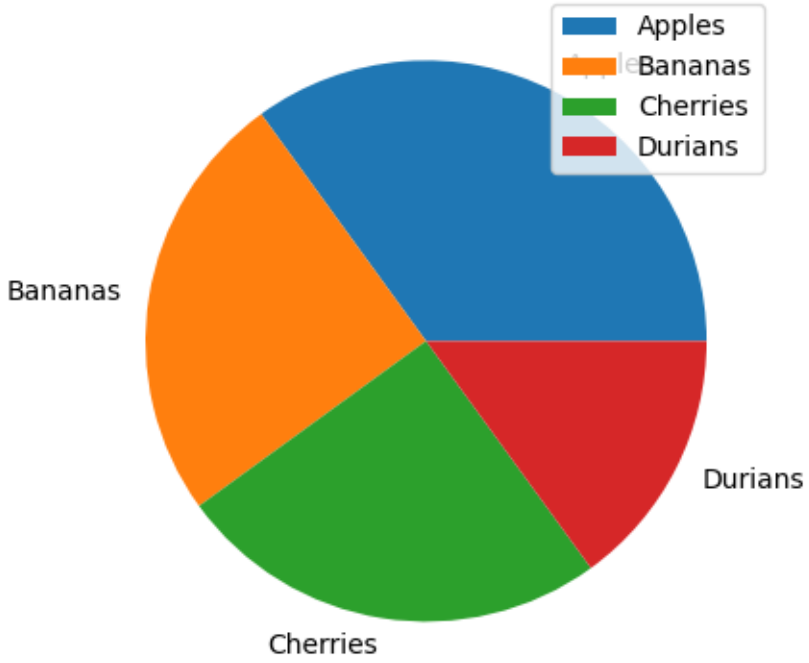
Histograms

```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.normal(170, 10, 250)

plt.hist(x)
plt.show()
```



Pie Charts

```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([35, 25, 25, 15])
mylabels = ["Apples","Bananas","Cherries","Durians"]

plt.pie(y, labels = mylabels)
plt.legend()
plt.show()
```



References

- Matplotlib's Gallery: https://matplotlib.org/stable/gallery/index.html
- Seaborn's Gallery: https://seaborn.pydata.org/examples/index.html

  Note: pip install seaborn first.

Exercise 5

- Read data from “customer.csv”.
- Extract gender, age and spending score.
- Replace gender with numeric values.
- Draw a scatter plot for the data using Matplotlib.

Exercise 6

- Read data from “customer.csv”.
- Extract gender attribute.
- Count the number of Male and Female in the dataset.
- Draw a pie chart for the statistics using Matplotlib.