Data Visualization using Matplotlib

1. Line Plots
2. Bar Charts
3. Pie Charts
4. Stack Plots
5. Histograms
6. Scatter Plots
7. Subplots

1. Creating Plots

```
# Installation: pip install matplotlib/ conda install matplotlib
```

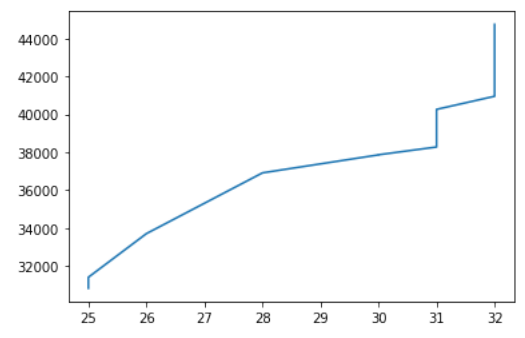
```
import matplotlib.pyplot as plt
import random
```

```
# generating 10 random numbers between 25 to 35
ages = [random.randrange(25,35,1) for ages in range(11)]
ages = sorted(ages, reverse=False)
```

```
# generating 10 random numbers between 30k to 45k

devs = [random.randrange(30000,45000,1) for devs in range(11)]
devs = sorted(devs, reverse=False)
```

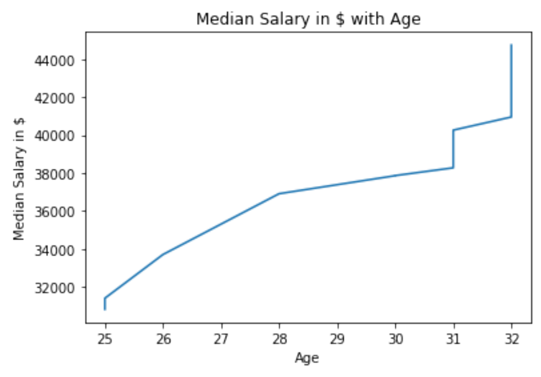
1.1. Plotting Line Plot

```
plt.plot(ages, devs)
plt.show()
```



1.2. Adding title, xlabel and ylabel

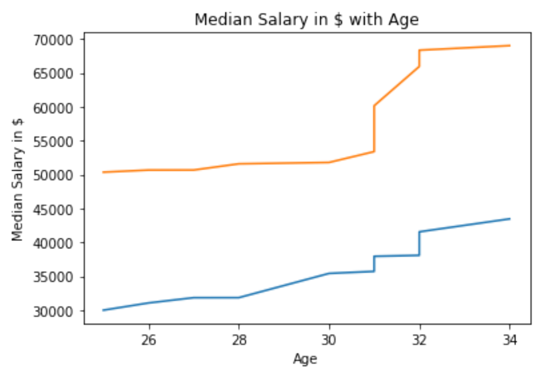
```
plt.plot(ages, devs)
plt.title("Median Salary in $ with Age") # add the title
plt.xlabel("Age") # add xlabel
plt.ylabel("Median Salary in $") #add ylabel
plt.show()
```



1.3. Adding more plot to the same graph

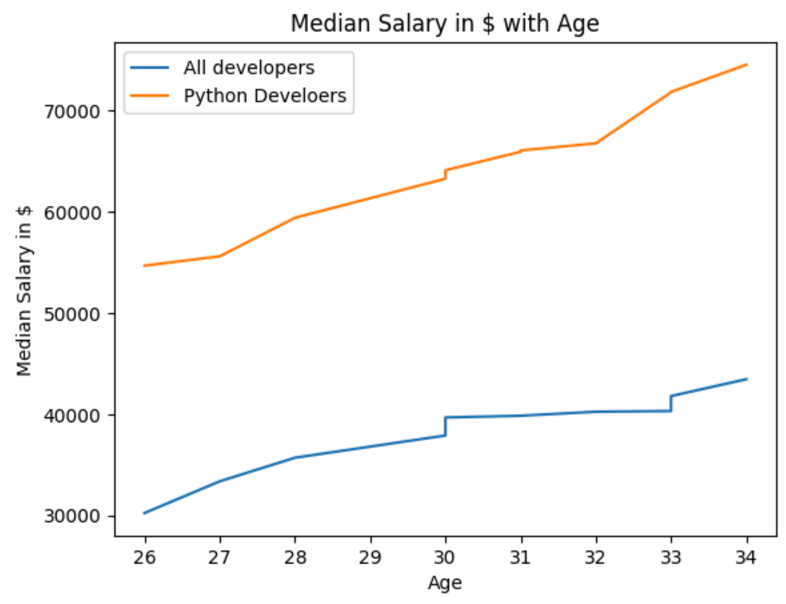
```
#creating 10 random numbers between 50k to 75k
import random
import matplotlib.pyplot as plt
ages = [random.randrange(25,35,1) for ages in range(11)]
ages = sorted(ages, reverse=False)
devs = [random.randrange(30000,45000,1) for devs in range(11)]
devs = sorted(devs, reverse=False)
py_devs = [random.randrange(50000,75000) for py_devs in range(11)]
py_devs = sorted(py_devs, reverse=False)
```

```
plt.plot(ages, devs)
plt.plot(ages, py_devs) # adding other plot to the same figure
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.show()
```



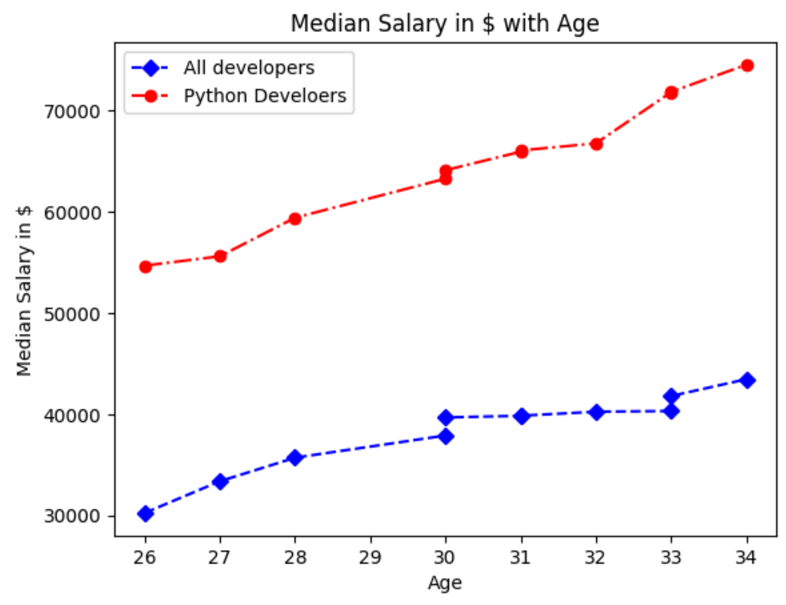
1.4. Adding legend to the plot

```
plt.plot(ages, devs, label = "All developers") # label
plt.plot(ages, py_devs, label = "Python Develoers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.legend() #plot the legend
plt.show()
```



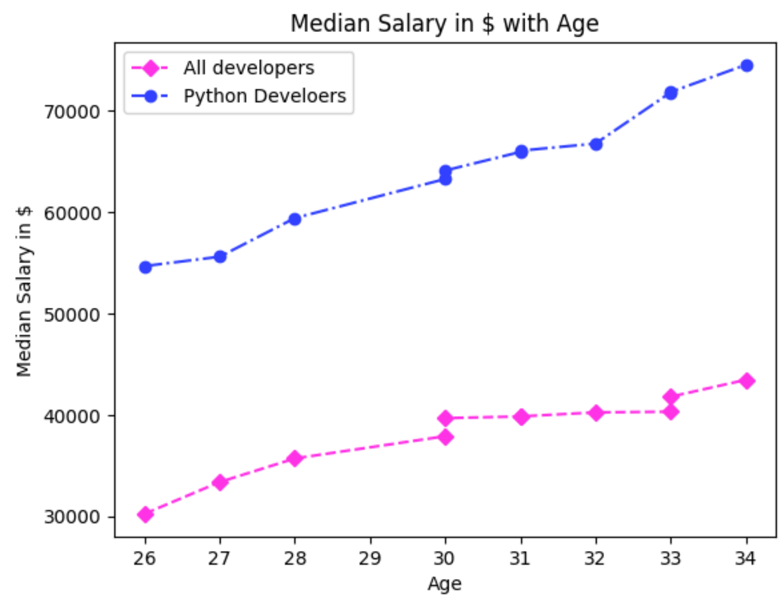
1.5. Setting marker, linestyle and color

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.plot(ages, devs, color="blue", linestyle = "--", marker = "D", label = "All developers")
plt.plot(ages, py_devs, color="red", linestyle = "-.", marker = "o", label = "Python Develoers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.legend()
plt.show()
```



1.6. Hexadecimal code for colors

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.plot(ages, devs, color="#FF33E9", linestyle = "--", marker = "D", label = "All developers")
plt.plot(ages, py_devs, color="#3344FF", linestyle = "-.", marker = "o", label = "Python Develoers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.legend()
plt.show()
```



Adding other plot to the same graph

```
#creating 10 random numbers between 40k to 60k

js_devs = [random.randrange(40000,60000) for js_devs in range(11)]
js_devs = sorted(js_devs, reverse=False)
```

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.plot(ages, devs, color="#FF33E9", linestyle = "--", marker = "D", label = "All developers")
plt.plot(ages, py_devs, color="#3344FF", linestyle = "-.", marker = "o", label = "Python Developers")
plt.plot(ages, js_devs, color="#FF3355", linestyle = ":", marker = "x", label = "Javascript Developers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.legend()
plt.show()
```

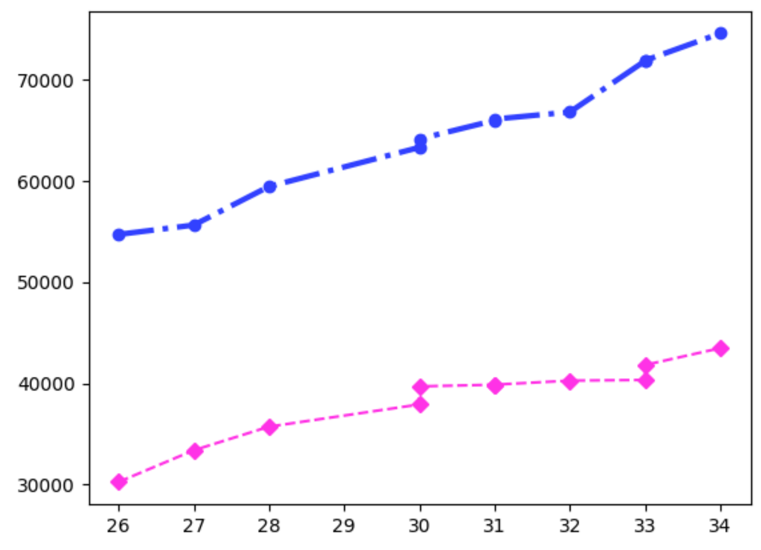
1.7. Changing the line width

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.plot(ages, devs, color="#FF33E9", linestyle = "--", marker = "D", label = "All developers")
plt.plot(ages, py_devs, color="#3344FF", linestyle = "-.", marker = "o", linewidth=3, label = "Python Developers")
plt.plot(ages, js_devs, color="#FF3355", linestyle = ":", marker = "x", label = "Javascript Developers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.legend()
plt.show()
```

1.8. Add padding to the plot

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.plot(ages, devs, color="#FF33E9", linestyle = "--", marker = "D", label = "All developers")
plt.plot(ages, py_devs, color="#3344FF", linestyle = "-.", marker = "o", linewidth=3, label = "Python Developers")
plt.plot(ages, js_devs, color="#FF3355", linestyle = ":", marker = "x", label = "Javascript Developers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.legend()
plt.tight_layout() #adds padding
plt.show()
```

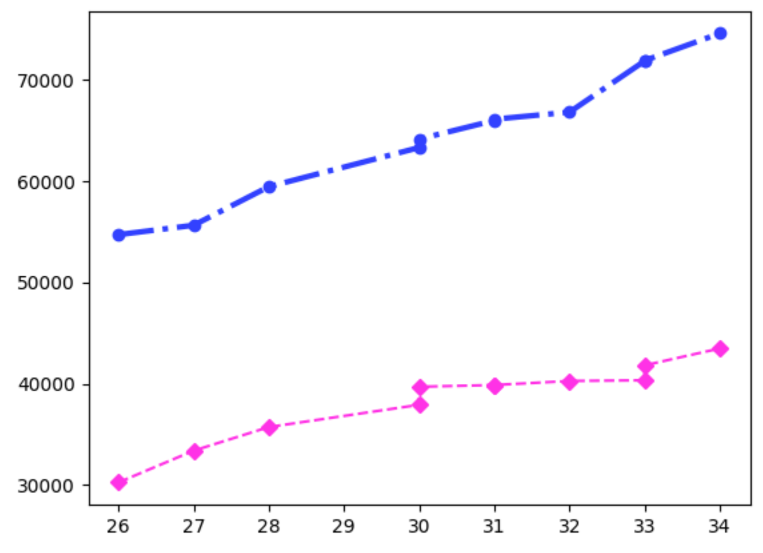
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-6-5d1c2df588d0> in <cell line: 5>()
      3 plt.plot(ages, devs, color="#FF33E9", linestyle = "--", marker = "D", label = "All developers")
      4 plt.plot(ages, py_devs, color="#3344FF", linestyle = "-.", marker = "o", linewidth=3, label = "Python Developers")
----> 5 plt.plot(ages, js_devs, color="#FF3355", linestyle = ":", marker = "x", label = "Javascript Developers")
      6 plt.title("Median Salary in $ with Age")
      7 plt.xlabel("Age")

NameError: name 'js_devs' is not defined
```



1.9. Adding grid to the plot

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.plot(ages, devs, color="#FF33E9", linestyle = "--", marker = "D", label = "All developers")
plt.plot(ages, py_devs, color="#3344FF", linestyle = "-.", marker = "o", linewidth=3, label = "Python Developers")
plt.plot(ages, js_devs, color="#FF3355", linestyle = ":", marker = "x", label = "Javascript Developers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.grid(True)
plt.legend()
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-5-25c33cba3d79> in <cell line: 5>()
      3 plt.plot(ages, devs, color="#FF33E9", linestyle = "--", marker = "D", label = "All developers")
      4 plt.plot(ages, py_devs, color="#3344FF", linestyle = "-.", marker = "o", linewidth=3, label = "Python Developers")
----> 5 plt.plot(ages, js_devs, color="#FF3355", linestyle = ":", marker = "x", label = "Javascript Developers")
      6 plt.title("Median Salary in $ with Age")
      7 plt.xlabel("Age")

NameError: name 'js_devs' is not defined
```



1.10. Changing style of the plot

```
print(plt.style.available)
```

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.style.use('seaborn-bright') #to change the style
plt.plot(ages, devs, label = "All developers")
plt.plot(ages, py_devs, label = "Python Developers")
plt.plot(ages, js_devs, label = "Javascript Developers")
plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")
plt.legend()
plt.show()
```

1.11. Saving the plot

```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

plt.style.use('ggplot')

plt.plot(ages, devs, label = "All developers")
plt.plot(ages, py_devs, label = "Python Developers")
plt.plot(ages, js_devs, label = "Javascript Developers")

plt.title("Median Salary in $ with Age")
plt.xlabel("Age")
plt.ylabel("Median Salary in $")

plt.legend()

plt.savefig("plot.png")#save the plot

plt.show()
```

for Further Reading click the below link

https://matplotlib.org/tutorials/introductory/pyplot.html

https://pythonbasics.org/matplotlib-line-chart/

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

```
from google.colab import drive
drive.mount('/content/drive')
```

```
data = pd.read_csv('/content/drive/My Drive/data/data_gapminder_gdp_oceania.csv',index_col='country')

print(data)
```

Plot data directly from a Pandas dataframe.

- We can also plot Pandas dataframes.
- This implicitly uses matplotlib.pyplot.
- Before plotting, we convert the column headings from a string to integer data type, since they represent numerical values

```
# Extract year from last 4 characters of each column name
# The current column names are structured as 'gdpPercap_(year)',
# so we want to keep the (year) part only for clarity when plotting GDP vs. years
# To do this we use strip(), which removes from the string the characters stated in the argument
# This method works on strings, so we call str before strip()

years = data.columns.str.strip('gdpPercap_')

# Convert year values to integers, saving results back to dataframe

data.columns = years.astype(int)

data.loc['Australia'].plot()
```

Select and transform data, then plot it.

- By default, DataFrame.plot plots with the rows as the X axis.
- We can transpose the data in order to plot multiple series.

```
data.T.plot()
plt.ylabel('GDP per capita')
```

Data can also be plotted by calling the matplotlib plot function directly.

- The command is plt.plot(x, y)
- The color and format of markers can also be specified as an additional optional argument e.g., b- is a blue line, g-- is a green dashed line.

Get Australia data from dataframe

```
Start coding or generate with AI.
```

Plot with both countries

- Select two countries' worth of data.
- Plot with differently-colored markers.
- Create legend.

```
Start coding or generate with AI.
```

Bar plot

ax.bar(x, height, width, bottom, align)

The function makes a bar plot with the bound rectangle of size (x −width = 2; x + width=2; bottom; bottom + height).
The parameters to the function are − x sequence of scalars representing the x coordinates of the bars.
align controls if x is the bar center (default) or left edge.height scalar or sequence of scalars representing the height(s) of the bars.
widthscalar or array-like, optional. the width(s) of the bars default 0.8
bottomscalar or array-like, optional. the y coordinate(s) of the bars default None.align{'center', 'edge'}, optional, default 'center' The function returns a Matplotlib

```
import matplotlib.pyplot as plt
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
langs = ['C', 'C++', 'Java', 'Python', 'PHP']
students = [23,17,35,29,12]
ax.bar(langs,students)
plt.show()
```

```
import numpy as np
import matplotlib.pyplot as plt
data = [[30, 25, 50, 20],
[40, 23, 51, 17],
[35, 22, 45, 19]]
X = np.arange(4)
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(X + 0.00, data[0], color = 'b', width = 0.25)
ax.bar(X + 0.25, data[1], color = 'g', width = 0.25)
ax.bar(X + 0.50, data[2], color = 'r', width = 0.25)
```

1. Write a Python program to draw line charts of the financial data of Alphabet Inc. between October 3, 2016 to October 7, 2016. Sample Financial data (fdata.csv):

Date,Open,High,Low,Close 10-03-16,774.25,776.065002,769.5,772.559998 10-04-16,776.030029,778.710022,772.890015,776.429993 10-05-16,779.309998,782.070007,775.650024,776.469971 10-06-16,779,780.47998,775.539978,776.859985 10-07-16,779.659973,779.659973,770.75,775.080017

Draw multiple lines in a same plot by adding a grid, different color lines, titles, lables and legends.

2. Write a Python programming to display a bar chart of the popularity of programming Languages. Sample data: Programming languages: Java, Python, PHP, JavaScript, C#, C++ Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7

3. Write a Python program to create bar plot from a DataFrame. Sample Data Frame:.
   a b c d e.
   2 4,8,5,7,6
   4 2,3,4,2,6
   6 4,7,4,7,8
   8 2,6,4,8,6
   10 2,4,3,3,2

4. Write a Python program to create a stacked bar plot with error bars.
   Note: Use bottom to stack the women's bars on top of the men's bars.
   Sample Data:

Means (men) = (22, 30, 35, 35, 26)
Means (women) = (25, 32, 30, 35, 29).
Men Standard deviation = (4, 3, 4, 1, 5)
Women Standard deviation = (3, 5, 2, 3, 3)

5.Write a Python programming to create a pie chart of gold medal achievements of five most successful countries in 2022 Summer Olympics.
Read the data from a csv file.
Sample data: medal.csv.