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
```

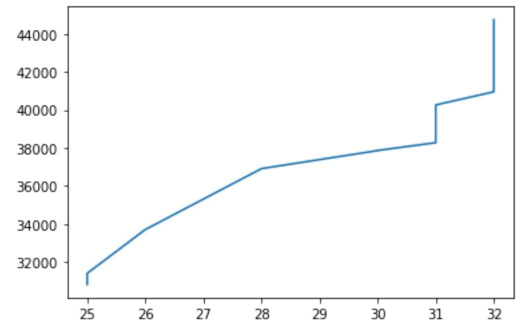
```
import matplotlib.pyplot as plt
import random
```

```
# 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)
```

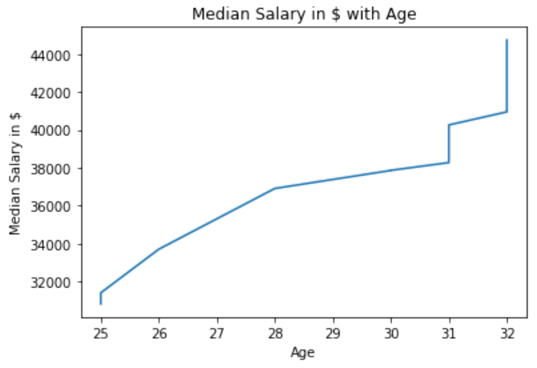
1.1. Plotting Line Plot

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



1.2. Adding title, xlabel and ylabel

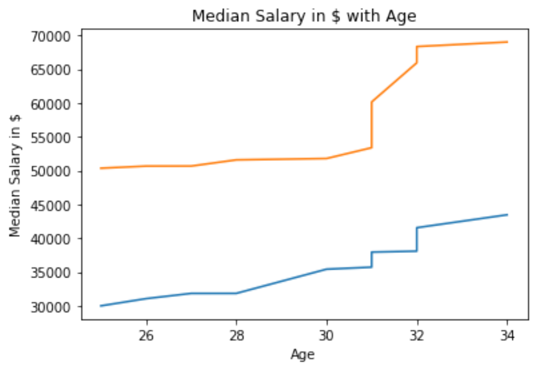
```
plt.plot(ages, devs)
plt.title("Median Salary in $ with Age") # add the title
plt.xlabel("Age") # add xlabel
plt.ylabel("Median Salary in $") #add ylabel
plt.show()
```



1.3. Adding more plot to the same graph

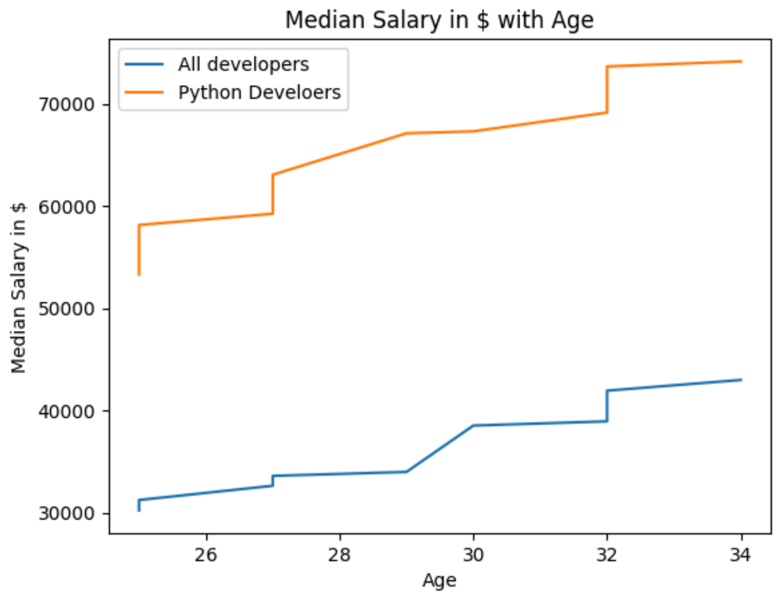
```
#creating 10 random numbers between 50k to 75k
import random
import matplotlib.pyplot as plt
ages = [random.randrange(25,35,1) for ages in range(11)]
ages = sorted(ages, reverse=False)
devs = [random.randrange(30000,45000,1) for devs in range(11)]
devs = sorted(devs, reverse=False)
py_devs = [random.randrange(50000,75000)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



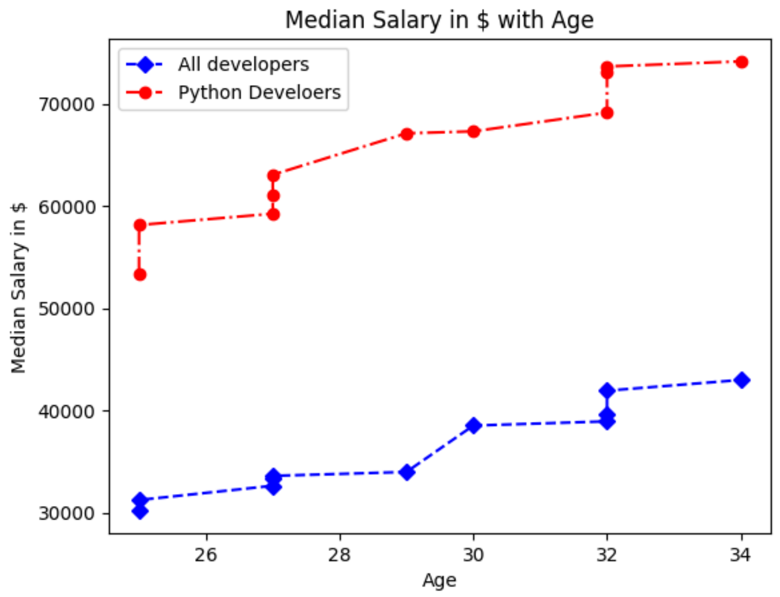
1.4. Adding legend to the plot

```
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()
```



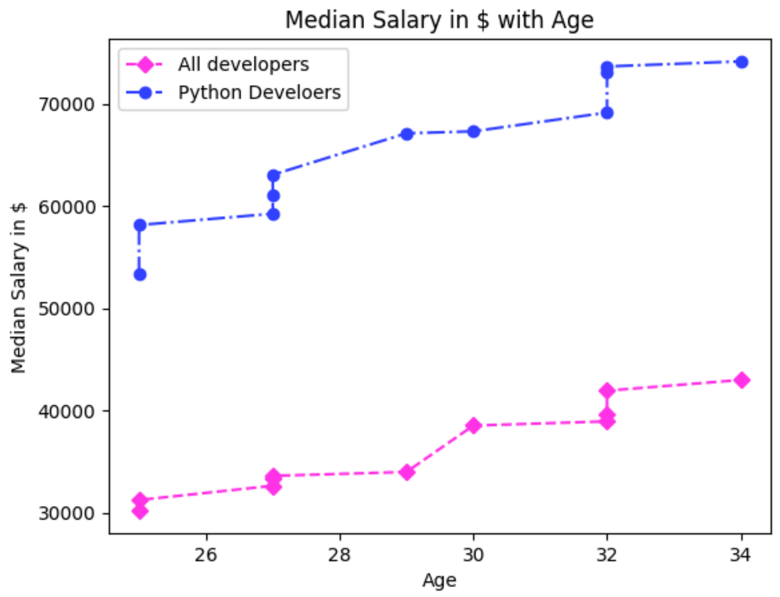
1.5. Setting marker, linestyle and color

```
#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()
```



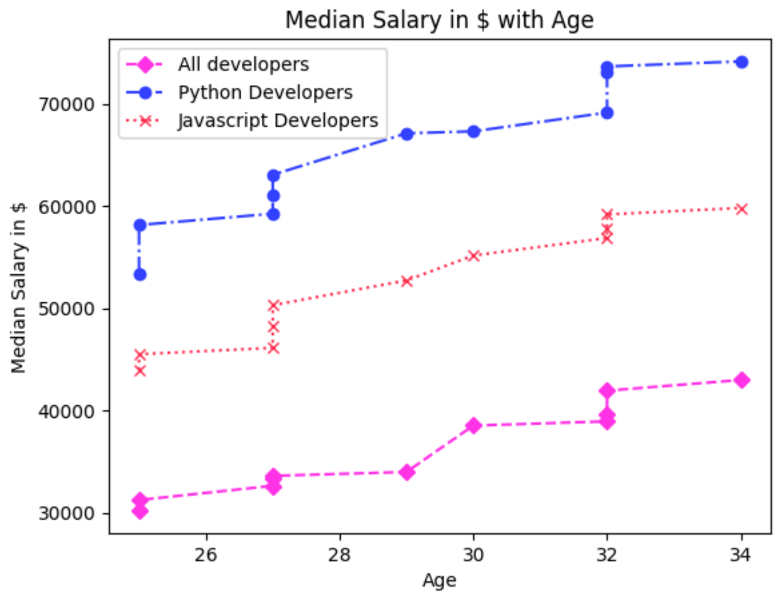
1.6. Hexadecimal code for colors

```
#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()
```



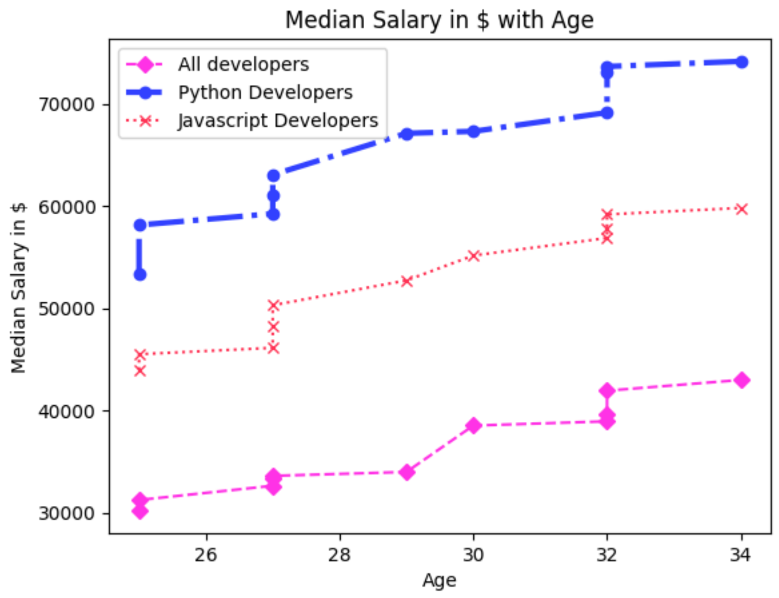
Adding other plot to the same graph

```
#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()
```



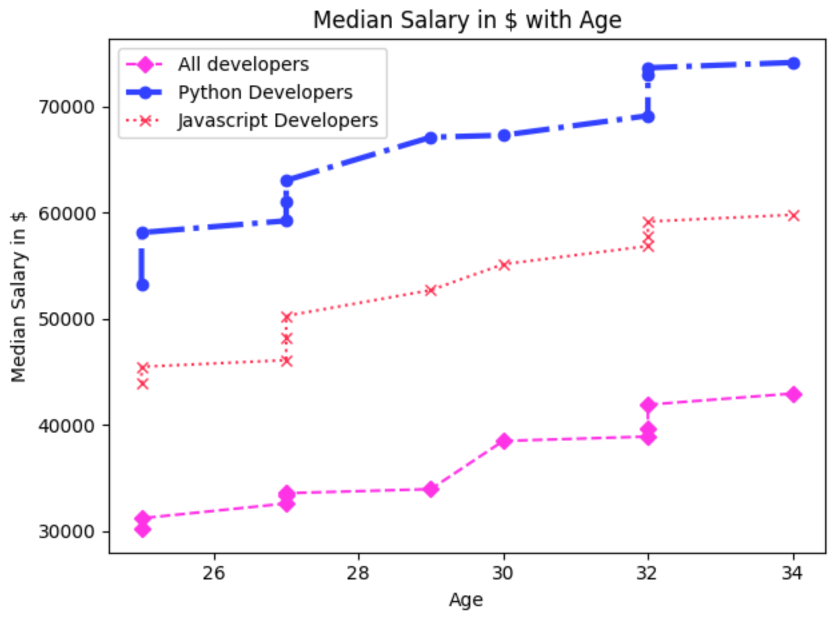
1.7. Changing the line width

```
#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()
```



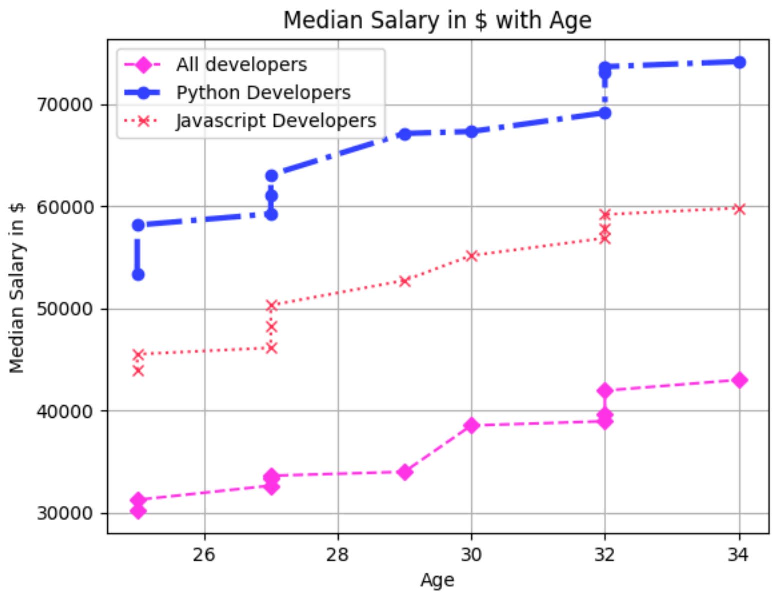
1.8. Add padding to the plot

```
#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()
```



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()
```

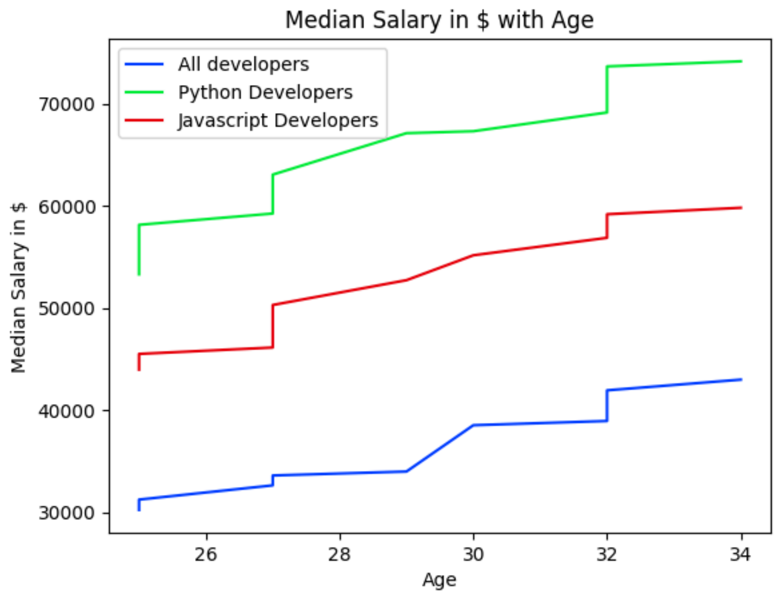


1.10. Changing style of the plot

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

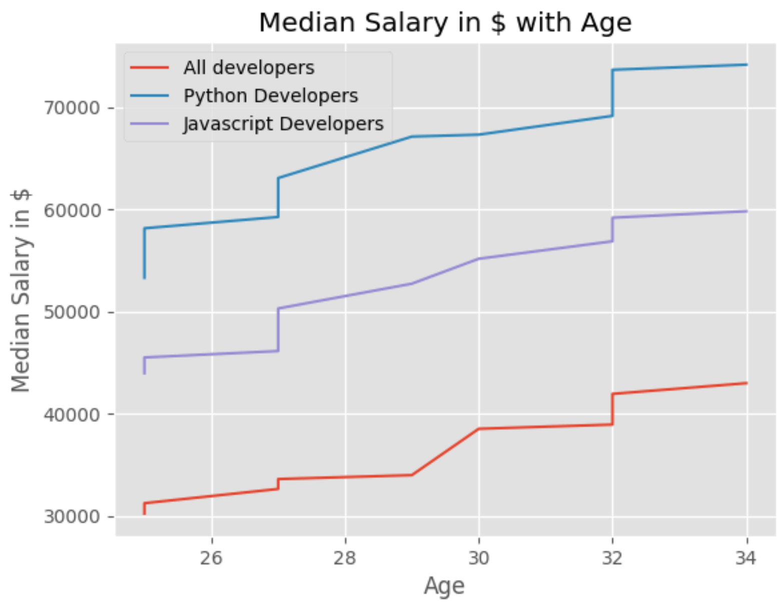
```
#https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html

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()
```

```
<ipython-input-19-d07ee214d04f>:3: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3
  plt.style.use('seaborn-bright') #to change the style
```



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()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-5-77a2eb0e84d7> in <cell line: 7>()
      5 # This method works on strings, so we call str before strip()
      6 
----> 7 years = data.columns.str.strip('gdpPercap_')
      8 
      9 # Convert year values to integers, saving results back to dataframe

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

Select and transform data, then plot it.