A Practical Approach to Timeseries Forecasting using Python

- Basic Data Manipulation in Time Series
- How to Install Packages?
- Basic Plotting and Data Visualization in Python
- Dataset Manipulation and Slicing
- Overview of Time Series Parameters

Shahzaib Hamid
AI Sciences Instructor

Section Overview?

- Pandas, Numpy, MatplotLib and scikit-learn are basic important libraries of python.
- Pandas is used to develop and modify data frames.
- Numpy is one of the most commonly used packages for scientific computing in Python.
- Matplotlib is a cross-platform, data visualization and graphical plotting library for Python.
- Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python.

Packages Installation?

Recommended Tool:

- Anaconda Distribution of Python

Steps:

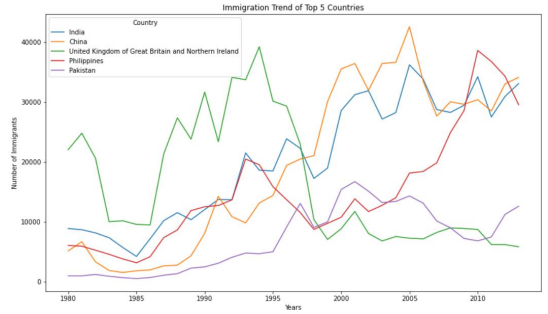
- Visit Anaconda.com/downloads
- Select Windows
- Download the .exe installer
- Open and run the .exe. Installer
- Open the Anaconda Prompt and you are good to go
- After Installation, create new environment on Anaconda
- pip install Jupyter notebook
- Install dependencies

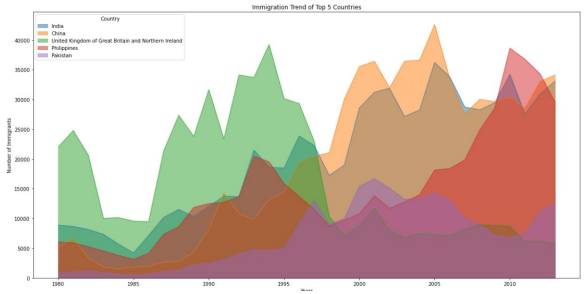
Basic Plotting and Data Visualization in Python

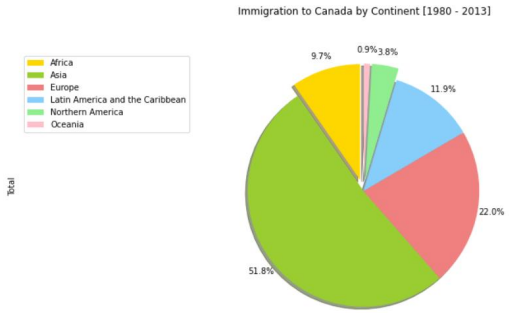
There are mainly three core frameworks for data visualization in Python.

1. Basic matplotlib
2. Area Plots, Histograms, and Bar Plots
3. Pie Charts, Box Plots, Scatter Plots, and Bubble Plots

Overview of Data Visualization







Basic matplotlib

Area Plots, Histograms, and Bar Plots

Pie Charts, Box Plots

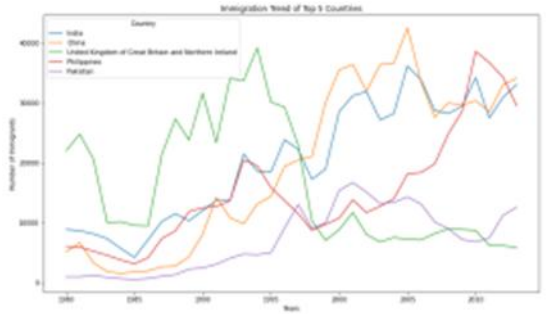
Matplotlib

```
[7]: import matplotlib.pyplot as plt
     import pandas as pd
     import numpy as np
```

	Type	Coverage	OdName	AREA	AreaName	REG	RegName	DEV	DevName	1980	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2
0	Immigrants	Foreigners	Afghanistan	935	Asia	5501	Southern Asia	902	Developing regions	16	...	2978	3436	3009	2652	2111	1746	1758	2203	2635	2
1	Immigrants	Foreigners	Albania	908	Europe	925	Southern Europe	901	Developed regions	1	...	1450	1223	856	702	560	716	561	539	620	
2	Immigrants	Foreigners	Algeria	903	Africa	912	Northern Africa	902	Developing regions	80	...	3616	3626	4807	3623	4005	5393	4752	4325	3774	4
3	Immigrants	Foreigners	American Samoa	909	Oceania	957	Polynesia	902	Developing regions	0	...	0	0	1	0	0	0	0	0	0	
4	Immigrants	Foreigners	Andorra	908	Europe	925	Southern Europe	901	Developed regions	0	...	0	0	1	1	0	0	0	0	1	

5 rows × 43 columns

Country	Continent	Region	DevName	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	3436	3009	2652	2111	1746	1758	2203	2635	2004	58639
Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	1	...	1223	856	702	560	716	561	539	620	603	15699
Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	69	...	3626	4807	3623	4005	5393	4752	4325	3774	4331	69439
American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	0	...	0	1	0	0	0	0	0	0	0	6
Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	2	...	0	1	1	0	0	0	0	1	1	15



Install Dependencies

Load Dataset

Data slicing

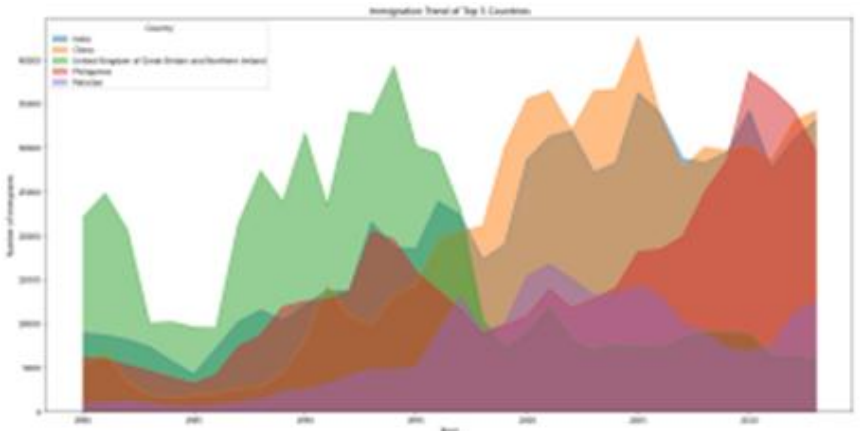
Data visualization using matplotlib

Area Plots, Histograms, and Bar Plots

```
df = pd.read_excel(
    'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DV0101EN-SkillsNetwork/Data%20F
    sheet_name='Canada by Citizenship',
    skiprows=range(20),
    skipfooter=2)

print('Data read into a pandas dataframe!')
```

Country	India	China	United Kingdom of Great Britain and Northern Ireland	Philippines	Pakistan
1980	8880	5123	22045	6051	978
1981	8670	6682	24796	5921	972
1982	8147	3308	20620	5249	1201
1983	7338	1863	10015	4562	900
1984	5704	1527	10170	3801	668

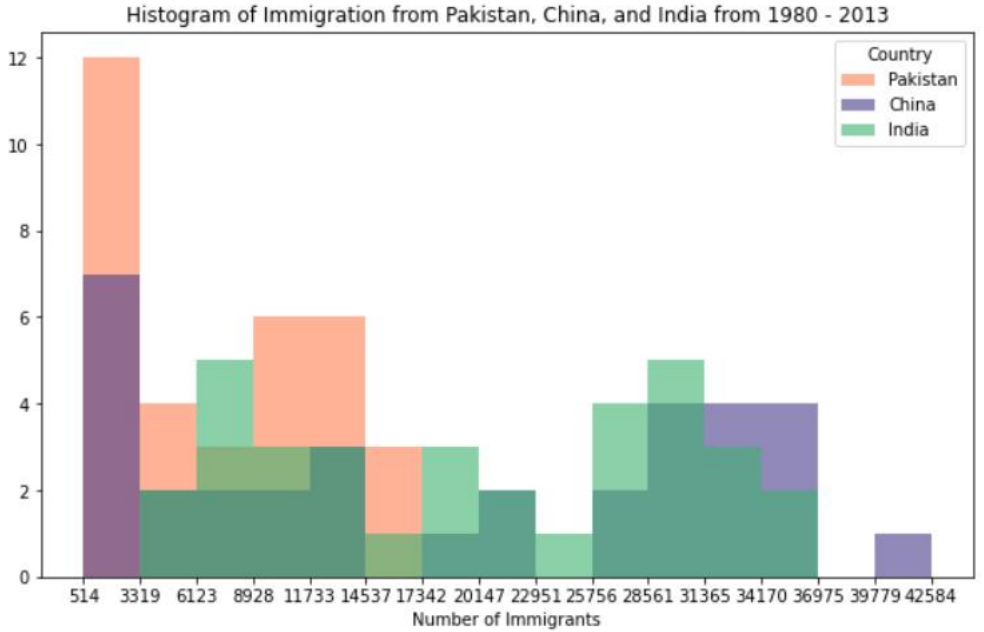


Load Dataset

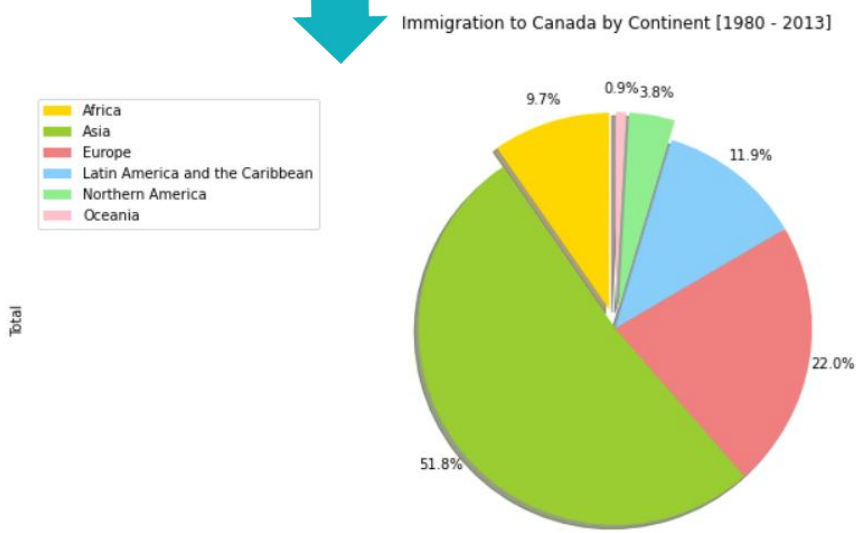
Data Sorting

Data visualization of Area Plot

Histogram and Pie Charts

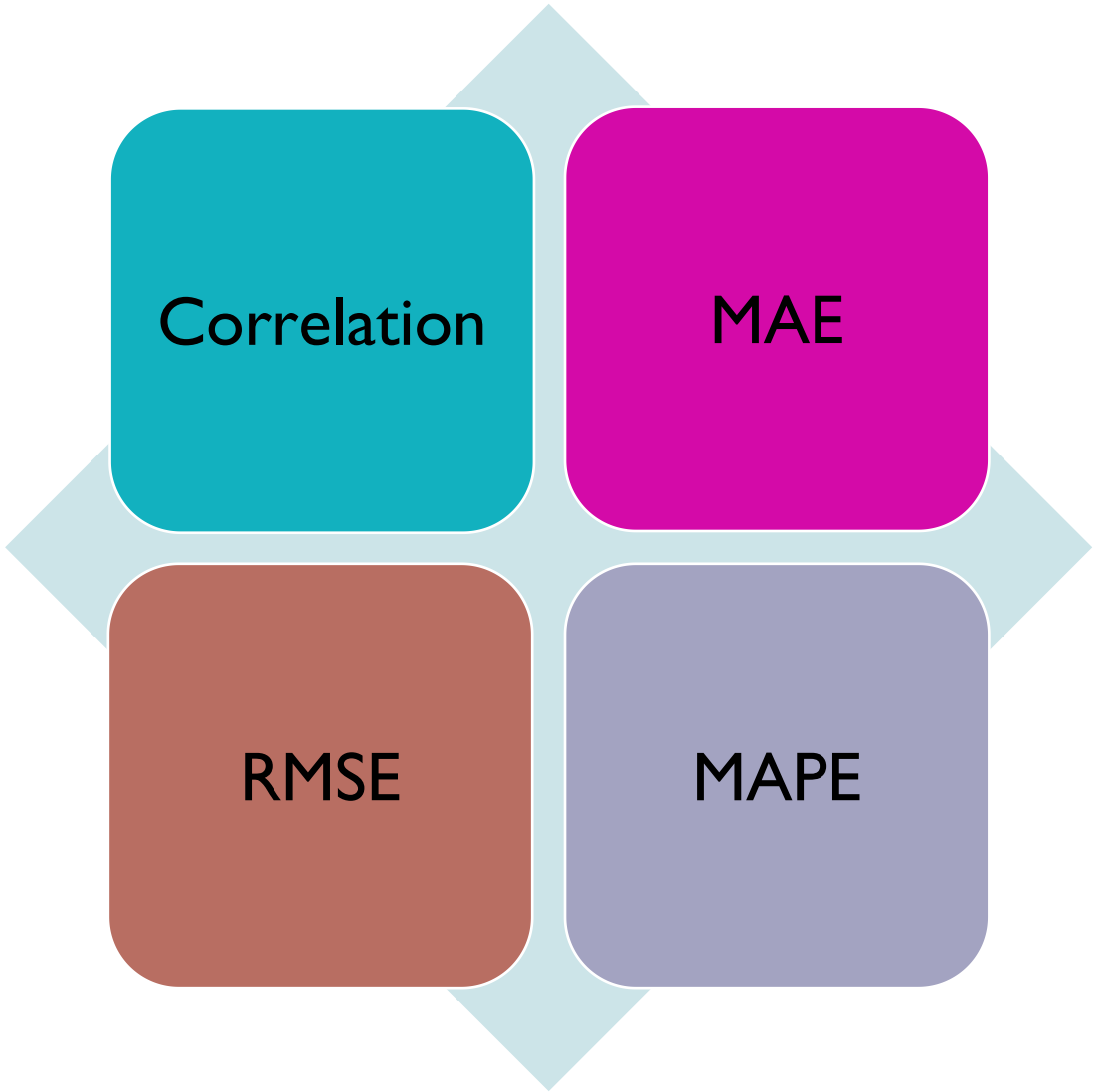


Histogram



Pie Charts

Overview of Time Series Parameters



Correlation

- It can be useful in data analysis and modeling to better understand the relationships between variables.
- The statistical relationship between two variables is referred to as their correlation.

In [ ]:
```
# generate related variables
from numpy import mean
from numpy import std
from numpy.random import randn
from numpy.random import seed
from matplotlib import pyplot
# seed random number generator
seed(1)
# prepare data
data1 = 20 * randn(1000) + 100
data2 = data1 + (10 * randn(1000) + 50)
# summarize
print('data1: mean=%.3f stdv=%.3f' % (mean(data1), std(data1)))
print('data2: mean=%.3f stdv=%.3f' % (mean(data2), std(data2)))
# plot
pyplot.scatter(data1, data2)
pyplot.show()

#Running the example first prints the mean and standard deviation for each variable.
```

- numpy, matplotlib

Mean Absolute Error

- Mean Absolute Error calculates the average difference between the calculated values and actual values.
- It is also known as scale-dependent accuracy as it calculates error in observations taken on the same scale.
- It is used as evaluation metrics for regression models in machine learning.

```
# import the module
from sklearn.metrics import mean_absolute_error as mae

# list of integers of actual and calculated
actual = [2, 3, 5, 5, 9]
calculated = [3, 3, 8, 7, 6]

# calculate MAE
error = mae(actual, calculated)
```

- scikit-learn

Root Mean Square Error

- Root Mean Square Error, which is the square root of value obtained from Mean Square Error function.
- Using RMSE, we can easily plot a difference between the estimated and actual values of a parameter of the model.

```
In [ ]: from sklearn.metrics import mean_squared_error
        import math
        y_actual = [1,2,3,4,5]
        y_predicted = [1.6,2.5,2.9,3,4.1]

        MSE = mean_squared_error(y_actual, y_predicted)

        RMSE = math.sqrt(MSE)
        print("Root Mean Square Error:\n")
        print(RMSE)
```

- scikit-learn

Mean Absolute Percentage Error

- It is a statistical measure to define the accuracy of a machine learning algorithm on a particular dataset.
- Can be considered as a loss function to define the error termed by the model evaluation.
- It helps estimate the accuracy in terms of the differences in the actual v/s estimated values.

```
In [ ]: from sklearn.metrics import mean_absolute_percentage_error
        y_true = [3, -0.5, 2, 7]
        y_pred = [2.5, 0.0, 2, 8]
        mean_absolute_percentage_error(y_true, y_pred)
```

- scikit-learn