Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

AY: 2023-24

Class:	TE	**Semester:**	VI
Course Code:	CSL601	**Course Name:**	Data Analytics and Visualization

Name of Student:	Ojasi Prashant Prabhu
Roll No.:	43
Experiment No.:	5
Title of the Experiment:	Implementation of ARIMA model in Python / R.
Date of Performance:	
Date of Submission:	

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations (BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty : Ms Bhavika Gharat

Signature :

Date :

Experiment No - 8

Aim - Data Visualization: Use Matplotlib and seabom Python libraries for visualization.

Objective:- To understand and apply the Matplotlib and seabom python libraries for visualization using python.

Description-

Data visualization is an easier way of presenting the data, however complex it is, to analyze trends and relationships amongst variables with the help of pictorial representation.

The following are the advantages of Data Visualization

- Easier representation of compels data
- Highlights good and bad performing areas
- Explores relationship between data points
- Identifies data patterns even for larger data points

While building visualization, it is always a good practice to keep some below mentioned points in mind

- Ensure appropriate usage of shapes, colors, and size while building visualization
- Plots/graphs using a co-ordinate system are more pronounced
- Knowledge of suitable plot with respect to the data types brings more clarity to the information
- Usage oflabels, titles, legends and pointers passes seamless information the wider audience

Python Libraries

There are a lot of python libraries which could be used to build visualization like matplotlib, vispy, bokeh, seabom, pygal, folium, plotly, cufflinks, and networkx. Of the many, matplotlib and seabom seems to be very widely used for basic to intermediate level of visualizations.

Matplotlib

It is an amazing visualization library in Python for 2D plots of arrays, It is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. Let's try to understand some of the benefits and features of matplotlib

- It's fast, efficient as it is based on numpy and also easier to build
- Has undergone a lot of improvements from the open source community since inception and hence a better library having advanced features as well
- Well maintained visualization output with high quality graphics draws a lot of users to it
- Basic as well as advanced charts could be very easily built
- From the users/developers point of view, since it has a large community support, resolving issues and debugging becomes much easier

Seaborn

Conceptualized and built originally at the Stanford University, this library sits on top of matplotlib. In a sense, it has some flavors of matplotlib while from the visualization point, it is much better than matplotlib and has added features as well. Below are its advantages

- Built-in themes aid better visualization
- Statistical functions aiding better data insights
- Better aesthetics and built-in plots
- Helpful documentation with effective examples

Nature of Visualization

Depending on the number of variables used for plotting the visualization and the type of variables, there could be different types of charts which we could use to understand the relationship. Based on the count of variables, we could have

- Univariate plot(involves only one variable)
- Bivariate plot(more than one variable in required)

A Univariate plot could be for a continuous variable to understand the spread and distribution of the variable while for a discrete variable it could tell us the count

Similarly, a Bivariate plot for continuous variable could display essential statistic like correlation, for a continuous versus discrete variable could lead us to very important conclusions like understanding data distribution across different levels of a categorical variable. A bivariate plot between two discrete variables could also be developed.

Scatter Plot

Scatter plots or scatter graphs is a bivariate plot having greater resemblance to line graphs in the way they are built. A line graph uses a line on an X-Y axis to plot a continuous function, while a scatter plot relies on dots to represent individual pieces of data. These plots are very useful to see if two variables are correlated. Scatter plot could be 2 dimensional or 3 dimensional.

Syntax: seaborn.scatterplot(x=None , y=None, hue=None , style=None, size=None, data=None, palette=None, hue_order=None, hue_norm=None, sizes=None, size_order=None, size_norm=None, markers=True, style_order=None, x_bins=None, y_bins=None, units=None, estimator=None, ci=95, n_boot= lOOO, alpha='auto', xjitter=None, yjitter=None, legend='brief', ax=None, **kwargs)
Parameters:
x, y: Input data variables that should be numeric.

data: Dataframe where each column is a variable and each row is an observation.

size: Grouping variable that will produce points with different sizes.

style: Grouping variable that will produce points with different markers.

palette: Grouping variable that will produce points with different markers.

markers: Object determining how to draw the markers for different levels.

alpha: Proportional opacity of the points.

Returns: This method returns the Axes object with the plot drawn onto it.

Advantages of a scatter plot

- Displays correlation between variables
- Suitable for large data sets
- Easier to find data clusters
- Better representation of each data point

Histogram

Histograms display counts of data and are hence similar to a bar chart. A histogram plot can also tell us how close a data distribution is to a normal curve. While working out statistical method, it is very important that we have a data which is normally or close to a normal distribution. However, histograms are univariate in nature and bar charts bivariate.

A bar graph charts actual counts against categories e.g. height of the bar indicates the number of items in that category whereas a histogram displays the same categorical variables in bins.

Bins are integral part while building a histogram they control the data points which are within a range. As a widely accepted choice we usually limit bin to a size of 5-20, however this is totally governed by the data points which is present.

Countplot

A countplot is a plot between a categorical and a continuous variable. The continuous variable in this case being the number of times the categorical is present or simply the frequency. In a sense, count plot can be said to be closely linked to a histogram or a bar graph.

Syntax : seabom.countplot(x=None , y=None , hue=None , data=None, order=None, hue_order=None , orient=None, color=None, palette=None , saturation=0.75 , dodge=True , ax=None, **kwargs)
Parameters : This method is accepting the following parameters that are described below:

- x, y: This parameter take names of variables in data or vector data, optional, Inputs for plotting long-form data.
- hue : (optional) This parameter take column name for colour encoding.
- data : (optional) This parameter take DataFrame, array, or list of arrays, Dataset for plotting. Ifx and y are absent, this is interpreted as wide-form. Otherwise it is expected to be long-form.
- order, hue_order : (optional) This parameter take lists of strings. Order to plot the categorical levels in, otherwise the levels are inferred from the data objects.
- orient : (optional)This parameter take "v" | "h", Orientation of the plot (vertical or horizontal). This is usually inferred from the dtype of the input variables but can be used to specify when the "categorical" variable is a numeric or when plotting wide-form data.
- color : (optional) This parameter take matplotlib color, Color for all of the elements, or seed for a gradient palette.
- palette : (optional) This parameter take palette name, list, or diet, Colors to use for the different levels of the hue variable. Should be something that can be interpreted by color_palette(), or a dictionary mapping hue levels to matplotlib colors.

- saturation : (optional) This parameter take float value, Proportion of the original saturation to draw colors at. Large patches often look better with slightly desaturated colors, but set this to 1 if you want the plot colors to perfectly match the input color spec.
- dodge : (optional) This parameter take bool value, When hue nesting is used, whether elements should be shifted along the categorical axis.
- ax : (optional) This parameter take matplotlib Axes, Axes object to draw the plot onto, otherwise uses the current Axes.
- kwargs : This parameter take key, value mappings, Other keyword arguments are passed through to matplotlib.axes.Axes.bar().

Returns: Returns the Axes object with the plot drawn onto it.

It simply shows the number of occurrences of an item based on a certain type of category.In python, we can create a count plot using the seaborn library. Seaborn is a module in Python that is built on top of matplotlib and used for visually appealing statistical plots.

Features	Matplotlib	Seaborn
Functionality	It is utilized for making basic graphs. Datasets are visualised with the help of bargraphs, histograms, piecharts, scatter plots, lines and so on.	Seaborn contains a number of patterns and plots for data visualization. It uses fascinating themes. It helps in compiling whole data into a single plot. It also provides distribution of data.

Syntax	It uses comparatively complex and lengthy syntax. Example: Syntax for bargraph- matplotlib.pyplot.bar(x_axis, y_axis).	It uses comparatively simple syntax which is easier to learn and understand. Example: Syntax for bargraph- seaborn.barplot(x_axis, y_axis).
Dealing Multiple Figures	We can open and use multiple figures simultaneously. However they are closed distinctly. Syntax to close one figure at a time: matplotlib.pyplot.close(). Syntax to close all the figures: matplotlib.pyplot.close("all")	Seaborn sets time for the creation of each figure. However, it may lead to (OOM) out of memory issues
Visualization	Matplotlib is well connected with Numpy and Pandas and acts as a graphics package for data visualization in python. Pyplot provides similar features and syntax as in MATLAB. Therefore, MATLAB users can easily study it.	Seaborn is more comfortable in handling Pandas data frames. It uses basic sets of methods to provide beautiful graphics in python.
Pliability	Matplotlib is a highly customized and robust	Seaborn avoids overlapping of plots with the help of its default themes

Data Frames and Arrays	Matplotlib works efficiently with data frames and arrays.It treats figures and axes as objects. It contains various stateful APis for plotting. Therefore plot() like methods can work without parameters.	Seaborn is much more functional and organized than Matplotlib and treats the whole dataset as a single unit. Seaborn is not so stateful and therefore, parameters are required while calling methods like plot()
Use Cases	Matplotlib plots vanous graphs using Pandas and Numpy	Seaborn is the extended version of Matplotlib which uses Matplotlib along with Numpy and Pandas for plotting graphs

Conclusion-

What is the difference between matplotlib and seaborn?

- Matplotlib: Lower-level library offering extensive control over plot elements but requiring more code.
- Seaborn: Built on top of Matplotlib, providing high-level functions for creating statistical graphics with simpler syntax and a focus on aesthetics.

By default, Plot() function plots a **Plots a line chart by default in both Matplotlib and Seaborn.**

To create a chart, pyplot provides

Matplotlib: Uses functions like plot(), scatter(), bar(), etc., for different chart types.

Seaborn: Provides high-level functions like distplot() for histograms, barplot() for bar charts,.

Which library is used to create statistical graphics in Python?
Seaborn is the library used to create statistical graphics in Python.

Which function is used to create a histogram in Seaborn?
Seaborn: displot() is used to create histograms in Seaborn.

Program 1- Scatter plot (create or upload any suitable data set)

```
# import required modules
from mpl toolkits.mplot3d import Axes3D

# assign axis values
```

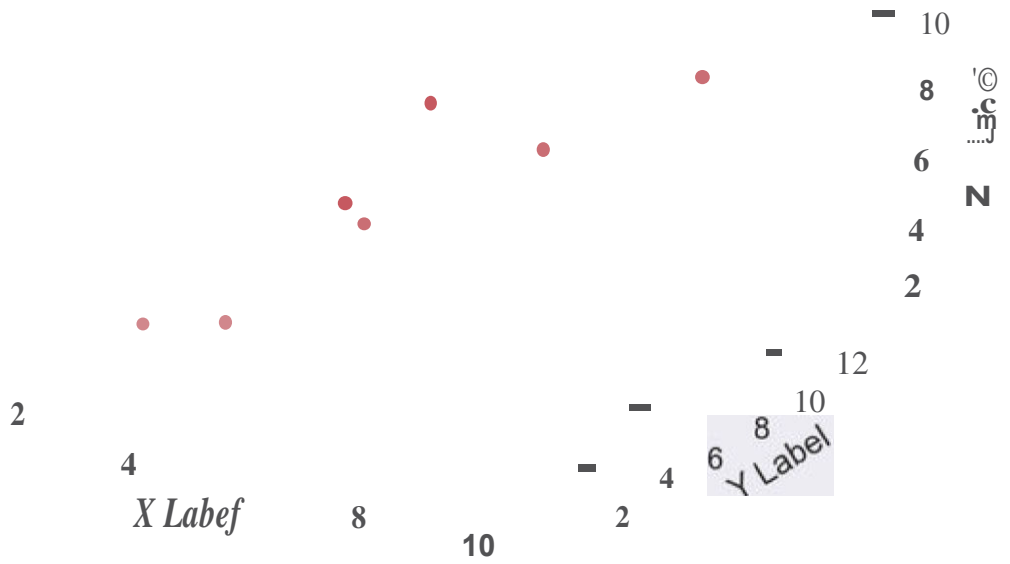
```
x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
y =  [5, 6, 2, 3, 13, 4, 1, 2, 4, 8]
z =  [2, 3, 3, 3, 5, 7, 9, 11, 9, 10]

# adjust size of plot
sns.set(rc={'figure.figsize' : (8, 5)})
fig = plt.figure()
ax = fig.add subplot(lll, projection='3d')
ax.scatter(x, y, z, c='r', marker='o')

# assign labels
ax.set xlabel('X Label'), ax.set ylabel('Y Label'), ax.set zlabel('Z Label')

# display illustration
plt.show()
```



2. Program - pie chart to show cars data

```
# Import required module
import matplotlib.pyplot as plt
import numpy as np

# Creating dataset
```

```
cars = ['AUDI', 'BMW', 'FORD', 'TESLA', 'JAGUAR', 'MERCEDES']
data = [23, 17, 35, 29, 12, 41]

# Creating explode data
explode = (0.1, 0.0, 0.2, 0.3, 0.0, 0.0)

# Creating color parameters
colors = ("orange", "cyan", "brown", "grey", "indigo", "beige")

# Wedge properties
wp = {'linewidth': 1, 'edgecolor': "green"}

# Creating autocpt arguments
def func(pct, allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return "{:.lf}%\n({:d} g)".format(pct, absolute)

# Creating plot
fig, ax = plt.subplots(figsize=(lO, 7))
wedges, texts, autotexts = ax.pie(data, autopct=lambda pct: func(pct,
data), explode=explode, labels=cars,
                                  shadow=True, colors=colors,
startangle=90, wedgeprops=wp,
                                  textprops=dict(color="magenta"))

# Adding legend
ax.legend(wedges, cars, title="Cars", loc="center left",
          bbox to anchor=(1, 0, 0.5, 1))
plt.setp(autotexts, size=B, weight="bold")
ax.set title("Customizing pie chart")

# Show plot
plt.show()
```

Customizing pie chart