**Datta Meghe College of Engineering**

**Department of Information Technology**

**LAB MANUAL**

**Academic Year: 2023-24**

**Semester: VI**

**Course Name: DS using Python Lab (SBL)**

**Course Code: ITL605**

**Rev-2019**

**PROGRAM OUTCOMES AS DEFINED BY NBA (PO)**



ENGINEERING GRADUATES WILL BE ABLE TO**:**

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

**DATTA MEGHE COLLEGE OF ENGINEERING, AIROLI, NAVI MUMBAI DEPARTMENT OF INFORMATION TECHNOLOGY**

**Institute Vision:** To create value - based technocrats to fit in the world of work and research

**Institute Mission:**

* To adapt the best engineering practices
* To empower students to work in the world of technology and research.
* To create competent human beings

**Department Vision:** To develop and foster students for successful careers in the dynamic field of Information Technology.

**Department Mission:**

| **M1:** | To create and disseminate knowledge through research, teaching & learning and to enhance society in meaningful and sustainable ways. |
| --- | --- |
| **M2:** | To impart a suitable environment for students and staff to showcase innovative ideas in the field of IT. |
| **M3:** | To bridge the curriculum gap by facilitating effective interaction among industry and Staff/Students. |

**Program Educational Objectives** (**PEO)**

| **PEO 1** | Develop proficiency as an IT technocrat with an ability to solve a wide range of computational problems in industry, government, or other work environments. |
| --- | --- |
| **PEO 2** | Attain the ability to adapt quickly to new environments and technologies, assimilate new information, and work in multi-disciplinary areas with a strong focus on innovation and entrepreneurship. |
| **PEO 3** | Prepare graduates with the ability of life-long learning to innovate in everchanging global economic and technological environments of the current era. |
| **PEO 4** | Possess the ability to function ethically and responsibly with good cultural values and integrity to apply the best principles and practices of Information Technology towards the society. |

**Program Specific Outcomes (PSO)**

| **PSO1** | Apply Core Information Technology knowledge to develop stable and secure IT system |
| --- | --- |
| **PSO2** | Design, IT infrastructures for an enterprise using concepts of best practices in Information Technology and security domain. |
| **PSO3** | Ability to work in multidisciplinary IT enabled projects for industry and society by adapting latest trends and technologies like Analytics, Blockchain, Cloud, Data science. |

**DATTA MEGHE COLLEGE OF ENGINEERING, AIROLI**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**Course Name: DSPYL (R-19) Course Code: ITL605**

**Year of Study: 2023 Semester: VI**

| LOs | Lab Outcomes | Cognitive Levels |
| --- | --- | --- |
| ITL605.1 | Understand the concept of Data science process and associated terminologies to solve real-world problems | L1 |
| ITL605.2 | Analyze the data using different statistical techniques and visualize the outcome using different types of plots.. | L1,L2,L3,L4 |
| ITL605.3 | Analyze and apply the supervised machine learning techniques like Classification, Regression or Support Vector Machine on data for building the models of data and solve the problems | L1,L2,L3,L4 |
| ITL605.4 | Apply the different unsupervised machine learning algorithms like Clustering, Decision Trees, Random Forests or Association to solve the problems. | L1, L2,L3 |
| ITL605.5 | Design and Build an application that performs exploratory data analysis using Apache Spark | L1,L2,L3,L4,L5,L6 |
| ITL605.6 | Design and develop a data science application that can have data acquisition, processing, visualization and statistical analysis methods with supported machine learning technique to solve the real-world problem | L1,L2,L3,L4,L5,L6 |

| **Datta Meghe**  **College of Engineering, Airoli, Navi Mumbai** | **DEPARTMENT OF INFORMATION TECHNOLOGY** |
| --- | --- |
| **List of Experiments**  **Subject: DSPYL lab Code: ITL605** |

| **Sr. No** |  | **Name of experiment** | **Cos**  **Covere d** | **Page No.** | **Date** | **Signature** |
| --- | --- | --- | --- | --- | --- | --- |
| **1** |  | **Data preparation using NumPy and Panda:** | LO1 |  |  |  |
|  | a | Derive an index field and add it to the data set. |  |  |  |
|  | b | Obtain a listing of all records that are outliers according to the any field. Print out a listing of the 10 largest values for that field. |  |  |  |
| **2** |  | **Data Visualization / Exploratory Data Analysis for the selected data set using Matplotlib** | LO2 |  |  |  |
|  | a | Create a bar graph, contingency table using any 2 variables |  |  |  |
|  | b | Describe what this graphs and tables indicates? |  |  |  |
| **3** |  | **Data Visualization / Exploratory Data Analysis for the selected data set using Seaborn** | LO2 |  |  |  |
|  | a | Create a normalized histogram. |  |  |  |
| **4** |  | **Data Modeling** | LO2 |  |  |  |
|  | a | Partition the data set, for example 75% of the records are included in the training data set and 25% are included in the test data set. Use a bar graph to confirm your proportions. |  |  |  |
|  | b | Validate your partition by performing a two‐sample Z‐test. |  |  |  |
| **5** |  | **Implementation of Statistical Hypothesis Test using Scipy** | LO2 |  |  |  |
|  | a | Normality Tests 1. Shapiro-Wilk Test |  |  |  |
|  | b | Analysis of Variance Test (ANOVA) |  |  |  |

| **6** |  | **Regression Analysis** | LO3 |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | a | Perform Logistic Regression to find out relation between variables. |  |  |  |
| **7** |  | **Classification modeling** | LO3 |  |  |  |
|  | a | Choose a classifier for classification problems. |  |  |  |
| **8** |  | **Clustering** | LO4 |  |  |  |
|  | a | Clustering algorithms for unsupervised classification. |  |  |  |
| **9.** |  | Using any machine learning techniques using available data set to develop a recommendation system. | LO1-LO5 |  |  |  |
| **10** |  | Exploratory data analysis using Apache Spark . | LO5 |  |  |  |
| **MP** |  | Implementation of Mini project based on case study using Data science and Machine learning | LO1-LO6 |  |  |  |

| **Sr.No**  **.** | **Name of the Assignment** | **CO Covered** | **Page No.** | **Date** | **Signature** |
| --- | --- | --- | --- | --- | --- |
| **1.** | Assignment 1 | LO1-LO6 |  | 8.2.23 |  |
| **2.** | Assignment 2 | LO1-LO6 |  | 16.3.23 |  |

**DATTA MEGHE COLLEGE OF ENGINEERING**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**ACADEMIC YEAR 2023-24(TERM II)**

**SUBJECT: DSPYL Lab**

**SEM: VI**

**RUBRICS FOR GRADING EXPERIMENTS**

| **Rubric**  **Number** | **Rubric Title** | **Criteria** | **Marks\***  **(out of 15)** |
| --- | --- | --- | --- |
| **R1** | Understanding of Experiment | Shows adequate knowledge of the experiment/ problem statement given. | **4** |
| Shows some understanding of the experiment/ problem statement given. | **3** |
| Shows hardly any understanding of the experiment/ problem statement given. | **2** |
| **R2** | Appropriateness of python  library/algorithm used and logic | Logic/Algorithm used for solving a problem was best/efficient.  Approach towards the problem was in right direction. | **4** |
| Logic/Algorithm used for solving a problem was just good enough/partially efficient. Approach towards the problem could have been better. | **3** |
| Logic/Algorithm used for solving a problem was poor/inefficient.  Approach towards the problem was  dissatisfying. | **2** |
| **R3** | Result validation | Output was perfectly desired and expected one. It was satisfying all the requirement of problem statement. | **4** |
| Output was partially satisfying the requirement of Problem statement. It was not completely expected/desired one. | **3** |
| Output was not problem specific. It was quite random and vague. | **2** |
| **R4** | LAB ETHICS | -Student's presence and behavior was professional and respectful.  - Implemented the experiment and submitted writeup on time. | **2** |
| - Student need a bit improvement in their presence and behavior to be professional and respectful.  - Implemented the experiment and submitted writeup with the delay of one week i.e. till next Session. | **1** |
| - Student need a significant improvement in their presence and behavior to be professional and respectful.  - Deadline for Implementing the experiment and submitting a writeup has been exceeded by more than a week. | **0** |
| **R5** | PRESENTATION | -Journal was neatly documented and well maintained. | **1** |
| -Journal was documented and maintained in a less satisfactory manner. | **0** |
| -Journal documentation and maintained in poor manner. | **0** |

|  |
| --- |
|  |

EXPERIMENT NO. 1A

AIM:. Derive an index field and add it to the data set.

## **THEORY:**

Python index() is an inbuilt function in Python, which searches for a given element from the start of the list and returns the index of the first occurrence.

**Syntax:** list\_name.index(element, start, end)

### **Parameters:**

* element – The element whose lowest index will be returned.
* start (Optional) – The position from where the search begins.
* end (Optional) – The position from where the search ends.

**Return:** Returns the lowest index where the element appears.

**Error:** If any element which is not present is searched, it raises a ValueError.

### **Types of indexing:**

1. **Implicit Indexing:**

Implicit indexing, also known as zero-based indexing, is the default way of accessing items in a sequence in Python. In this method, the first item in the sequence has an index of 0, the second item has an index of 1, and so on. For example:



### **Explicit Indexing:**

Explicit indexing, also known as named indexing, is a method of accessing items in a sequence using their names or keys. This method is commonly used with dictionaries, where each item is a key-value pair. For example:



***print(my\_dict['city']) # Output: 'New York'***

### **methods available in the Indexing**

1. **‘loc’:**

‘loc’ is a label-based method that is used to select data based on row labels and column names. It takes two arguments, the first argument is for the row labels, and the second argument is for the column names. For example:



### **‘iloc’:**

‘iloc’ is an integer-based method that is used to select data based on integer indices. It takes two arguments, the first argument is for the row indices, and the second argument is for the column indices. For example:



‘‘loc’’ and ‘iloc’’ methods are both available for implicit and explicit indexing in pandas.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To implement the indexing, we have used a dataset named as NBA.csv where the dataset is consisting of basketball players with their all details as age, height, weight, seasons they have play The attributes of the datasets are as follows:

[‘player\_name’, ‘team\_abbreviation’, ‘age’, ‘player\_height’, ‘player\_weight’, ‘college’, ‘country’, ‘draft\_year’, ‘draft\_round’, ‘pts’, ‘reb’, ‘ast’, ‘net\_rating’, ‘oreb\_pct’, ‘dreb\_pct’, ‘usg\_pct’, ‘ts\_pct’, ‘ast\_pct’, ‘season’]



## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Importing the dataset

**Step3:** Printing single column by making players name as an index

**Step4:** Printing multiple columns

**Step5:** Performing the task using ‘loc’ and i’loc’ method

* + Single row
  + Multiple row
  + Selected Rows and Columns
  + All Rows and Some Columns

**Step6:** Performing the task using ‘loc’ and ‘iloc’ method

* + Single row
  + Multiple row
  + Selected Rows and Columns
  + All Rows and Some Columns

**Step7:** Numeric Operations using . ‘loc’ method

**Step8:** Numeric operations using . ‘iloc’ Method

### **PYTHON CODE:**

**Step1:** start

**Step2:** Importing the dataset

**Step3:** Printing single column by making players name as an index



**Step4:** Printing multiple columns

**Step5:** Performing the task using ‘loc’ and i’loc’ method

* + Single row
  + Multiple row
  + Selected Rows and Columns
  + All Rows and Some Columns









**Step6:** Performing the task using ‘loc’ and ‘iloc’ method

* + Single row
  + Multiple row
  + Selected Rows and Columns
  + All Rows and Some Columns







**Step7:** Numeric Operations using . ‘loc’ method



**Step8:** Numeric operations using . ‘iloc’ Method



## **RESULT:**

* By using an indexing, we can retrieve the data which is required to performing a different task or analysing the dataset.
* By using indexing we can directly jump to the required data or attribute.
* We can also do the numeric operations using .iloc and loc method.

## **CONCLUSION:**

By implementing the indexing experiment, we have successfully done the following things

* We have learned what is indexing
* What are the types of indexing implicit & explicit indexing
* Which methods are available in it that is ‘loc’ & ‘iloc’ method
* We also learn how to perform different task using ‘loc’ & ‘iloc’ method which are as follows:

1. Single row
2. Multiple row
3. Selected Rows and Columns
4. All Rows and Some Columns

**AIM:**

# **EXPERIMENT NO. 1B**

1 B .Obtain a listing of all records that are outliers according to the any field. Print out a listing of the 10 largest values for that field.

## **THEORY:**

### **Outliers:**

In statistics, an outlier is an observation that lies an abnormal distance from other values in a random sample from a population. In other words, an outlier is a data point that is significantly different from other data points in a dataset. Outliers can occur due to measurement errors, data entry errors, or natural variation in the data.

### **Types of outliers:**

There are several types of outliers that can occur in a dataset. Some of the common types of outliers are:

1. Point outliers: These are individual data points that are significantly different from other data points in the dataset.
2. Contextual outliers: These are data points that are not necessarily different from other data points in the dataset, but they are unusual in the context of the problem being studied.
3. Collective outliers: These are groups of data points that are collectively different from the rest of the dataset.
4. Masking outliers: These are outliers that are hidden or masked by other outliers in the dataset.
5. Syntactical outliers: These are data points that are outliers simply because of a data entry error or formatting issue.
6. Statistical outliers: These are data points that are outliers based on a statistical criterion, such as a Z-score or interquartile range (IQR).

### **Outlier detection and removal:**

Outlier detection and removal are important steps in data pre-processing to ensure accurate statistical analysis and modelling. Here are some common methods for detecting and removing outliers:

### **Z-score method:**

This method involves calculating the Z-score for each data point in the dataset and removing any data points with a Z-score greater than a certain threshold (usually 3 or 4). The Z-score measures the number of standard deviations a data point is from the mean of the dataset.

### **Interquartile range (IQR) method:**

This method involves calculating the IQR for the dataset and removing any data points that

fall outside a certain range (usually 1.5 times the IQR below the first quartile or above the third quartile). The IQR measures the spread of the middle 50% of the dataset.

### **Visual inspection:**

This method involves plotting the data using box plots, scatter plots, or other visualization techniques and identifying any data points that appear to be significantly different from other data points.

### **Domain knowledge:**

This method involves using knowledge of the problem being studied to identify data points that are unlikely or impossible in the context of the problem.

### **Dataset and Performance Baseline:**

* we will first select a standard machine learning dataset and establish a baseline in performance on this dataset.
* This will provide the context for exploring the outlier identification and removal method of data preparation

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

* We have used House Price Regression Dataset in that we have used the house price regression dataset.
* This dataset has 13 input variables that describe the properties of the house and suburb and requires the prediction of the median value of houses in the suburb in thousands of dollars.
* You can learn more about the dataset here:
* House Price Dataset (housing.csv) House Price Dataset Description (housing.names) No need to download the dataset as we will download it automatically as part of our worked examples.

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Take the normal example of outlier

**Step3:** Learn Interquartile Range Method for outlier

**Step4:** Perform Automatic Outlier Detection

**Step5:** Removal of Outlier

**Step6:**Performing Baseline Model Performance(regression predictive modelling problem) **Step7:** Performing Automatic Outlier Detection-Isolation Forest (tree-based anomaly detection algorithm.)

**Step8:** Minimum Covariance Determinant (simple statistical methods to detect outliers)

**Step9:** Local Outlier Factor

**Step10:** One-Class SVM

### **PYTHON CODE:**

**Step1:** start

**Step2:** Take the normal example of outlier

|  |
| --- |

**Step3:** Learn Interquartile Range Method for outlier

|  |
| --- |

**Step4:** Perform Automatic Outlier Detection



**Step5:** Removal of Outlier

|  |
| --- |

**Step6:** Performing Baseline Model Performance(regression predictive modelling problem)

|  |
| --- |

**Step7:** Performing Automatic Outlier Detection-Isolation Forest (tree-based anomaly detection algorithm.)



**Step8:** Minimum Covariance Determinant (simple statistical methods to detect outliers)

**Step9:** Local Outlier Factor



**Step10:** One-Class SVM



## **RESULT:**

We have fit a linear regression algorithm and evaluate model performance by training the model on the test dataset and making a prediction on the test data and evaluate the predictions using the mean absolute error (MAE).

## **CONCLUSION:**

By implementing this outlier’s experiment, we have successfully done the following things

* We have understood what are the outliers, how to identify and remove them
* Why it is necessary to remove from data
* We learn normal outliers
* We identified the outliers with interquartile range
* We also find the outliers using Local Outlier Factor(LOF) and using Minimum Covariance Determinant
* We also written a code for Automatic Outlier Detection

**AIM:**

# **EXPERIMENT NO. 2**

Create a bar graph, contingency table using any 2 variables.

## **THEORY:**

### **Data Visualization:**

Data visualization is the process of representing data in a graphical or visual format to communicate information more effectively and efficiently. It involves creating charts, graphs, and other visual aids that help to present complex data in a simple, easy-to-understand manner.

The primary purpose of data visualization is to help people better understand and interpret data. It allows them to quickly identify patterns, trends, and relationships that may not be immediately apparent in the raw data. By presenting data in a visual format, it becomes easier to see the big picture and draw meaningful insights from the data.

There are several types of data visualizations, including:

1. Line charts: used to show trends and changes over time.
2. Bar charts: used to compare values across different categories or groups.
3. Scatter plots: used to show the relationship between two variables.
4. Heat maps: used to display data in a two-dimensional format, with colours indicating the magnitude of the data.
5. Tree maps: used to represent hierarchical data structures.

### **Matplotlib:**

Matplotlib is a widely used data visualization library in Python. It provides a range of functions to create different types of plots, including line plots, scatter plots, histograms, bar graphs, and more. Seaborn is another popular library that builds on top of Matplotlib and provides more advanced visualization capabilities, including heat maps, pair plots, and categorical plots.

### **Plotly:**

Plotly and Bokeh are interactive visualization libraries that allow users to create dynamic and interactive visualizations. Plotly provides a range of chart types, including scatter plots, line charts, bar charts, and more. Bokeh, on the other hand, provides tools for creating interactive visualizations that can be used in web applications.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we have used a Loan dataset in which the following attributes are included

[[“Loan\_ID”, “Gender”, “Married”, “Dependents”, “Education”, “Self\_Employed”, “ApplicantIncome”, “CoapplicantIncome”, “LoanAmount”, “Loan\_Amount\_Term”, “Credit\_History”, “Property\_Area”]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Load the Libraries

**Step3:** Load the Data

**Step4:** Code #1: Contingency Table showing correlation between Married and Dependents. **Step5:** Code #2: Contingency Table showing correlation between Purpose and loan status. **Step6:** Code #3: Contingency Table showing correlation between ApplicantIncome + LoanAmount and Loan\_Amount\_Term.

**Step8:** Plotting the bar graph

### **PYTHON CODE:**

**Step1:** start

**Step2:** Load the Libraries



**Step3:** Load the Data

|  |
| --- |
|  |

**Step4:** Code #1: Contingency Table showing correlation between Married and Dependents.



**Step5:** Code #2: Contingency Table showing correlation between Purpose and loan status.



**Step6:** Code #3: Contingency Table showing correlation between ApplicantIncome + LoanAmount and Loan\_Amount\_Term.



**Step8:** Plotting the bar graph

|  |
| --- |



## **RESULT:**

* Contingency Table 1 is showing correlation between Married and Dependents.
* Contingency Table 2 is showing correlation between Purpose and loan status.
* Contingency Table 3 is showing correlation between ApplicantIncome + LoanAmount and Loan\_Amount\_Term.
* We have plotted the graph where it shows relation between applicant income and different property area

## **CONCLUSION:**

By implementing this visualization experiment, we have successfully done the following things

* We learn what is data visualization and how it is useful for data analysis
* We have understood the corelation between different attributes using contingency table
  1. Contingency Table 1 is showing correlation between ‘Married’ and ‘Dependents’ attributes.
  2. Contingency Table 2 is showing correlation between ‘Purpose’ and ‘loan status’ attributes.
  3. Contingency Table 3 is showing correlation between ‘ApplicantIncome’ + ‘LoanAmount’ and ‘Loan\_Amount\_Term’ attributes.
* Now we are able to plot a bar graph of the any attribute with the corelation with other attribute for different datasets which will help to visualize the data with more efficient way

**AIM:**

# **EXPERIMENT NO. 3A**

Visualize data using simple Seaborn functions.

## **THEORY:**

### **Data Visualization:**

Data visualization is the process of representing data in a graphical or visual format to communicate information more effectively and efficiently. It involves creating charts, graphs, and other visual aids that help to present complex data in a simple, easy-to-understand manner.

The primary purpose of data visualization is to help people better understand and interpret data. It allows them to quickly identify patterns, trends, and relationships that may not be immediately apparent in the raw data. By presenting data in a visual format, it becomes easier to see the big picture and draw meaningful insights from the data.

There are several types of data visualizations, including:

1. Line charts: used to show trends and changes over time.
2. Bar charts: used to compare values across different categories or groups.
3. Scatter plots: used to show the relationship between two variables.
4. Heat maps: used to display data in a two-dimensional format, with colours indicating the magnitude of the data.
5. Tree maps: used to represent hierarchical data structures.

Data visualization is important for both data analysis and communication because it makes patterns, trends, and relationships that are not always obvious from raw data visible. It enables users to investigate data, find insights, and persuasively present findings to others.

Choosing the correct kind of chart or graph to represent the data, picking the right colours and labels, and making sure the visualization accurately and clearly depicts the data are all necessary for effective data visualization. When building a visualization, it is crucial to bear in mind both the intended audience and the project's goals.

A wide range of industries, including business, research, healthcare, and social sciences, use data visualization. Data visualization has become a crucial tool for evaluating and presenting complicated information as a result of the proliferation of big data and the increasing significance of data-driven decision making.

### **Libraries which are widely used in the visualization process:**

* **Matplotlib:**

Matplotlib is a widely used data visualization library in Python. It provides a range of functions to create different types of plots, including line plots, scatter plots, histograms, bar graphs, and more.

### **Seaborn:**

Seaborn is another popular library that builds on top of Matplotlib and provides more advanced visualization capabilities, including heat maps, pair plots, and categorical plots.

### **Plotly:**

Plotly and Bokeh are interactive visualization libraries that allow users to create dynamic and interactive visualizations. Plotly provides a range of chart types, including scatter plots, line charts, bar charts, and more. Bokeh, on the other hand, provides tools for creating interactive visualizations that can be used in web applications.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we have used a **“covid\_worldwide.csv”** dataset in which the following attributes are included

As well as we have used the inbuild dataset of seaborn library named as **‘tips’**

[[“Serial Number”, “Country”, “Total Cases”, “Total Deaths”, “Total Recovered”, “Active Cases”, “Total Test Population”]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Import the Libraries

**Step3:** Load the Data

**Step4:** Visualize the dataset using matplotlib and seaborn libraries and our imported dataset

* 1. Scatter plot
  2. Line plot

**Step5:** Visualize the dataset using matplotlib and seaborn libraries and seaborn inbuild dataset

1. Histogram
2. Density Plot
3. Box Plot
4. Violin Plot
5. Heatmap
6. Pair Plot

### **PYTHON CODE:**

**Step1:** start

**Step2:** Import the Libraries

|  |
| --- |

**Step3:** Load the Data



**Step4:** Visualize the dataset using matplotlib and seaborn libraries and our imported dataset

* 1. Scatter plot
  2. Line plot



**Step5:** Visualize the dataset using matplotlib and seaborn libraries and seaborn inbuild dataset

1. Histogram
2. Density Plot
3. Box Plot
4. Violin Plot
5. Heatmap
6. Pair Plot

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |



## **RESULT:**

* As we have plot different graphs to visualize the data in that we get the result that is we get the same output or same visualization of data but in a different form according to different types of graphs.

## **CONCLUSION:**

By implementing this visualization experiment, we have successfully done the following things

* We learn how to visualize the tabular dataset.
* What are different types of graphs that we can use for the visualization.
* How they react differently with dataset by its type of plotting but giving same visualization criteria.
* We learn that what aver the plots we are using to visualize the data it may differ from the plotting type but it will always give the same results.
* Now we can plot a different graph for different dataset to visualize and analyse them with the more efficient way.

# **EXPERIMENT NO. 3B**

## **AIM:**

Create normalized histogram using seaborn.

## **THEORY:**

### **Data Visualization:**

Data visualization is the process of representing data in a graphical or visual format to communicate information more effectively and efficiently. It involves creating charts, graphs, and other visual aids that help to present complex data in a simple, easy-to-understand manner.

The primary purpose of data visualization is to help people better understand and interpret data. It allows them to quickly identify patterns, trends, and relationships that may not be immediately apparent in the raw data. By presenting data in a visual format, it becomes easier to see the big picture and draw meaningful insights from the data.

There are several types of data visualizations, including:

1. Line charts: used to show trends and changes over time.
2. Bar charts: used to compare values across different categories or groups.
3. Scatter plots: used to show the relationship between two variables.
4. Heat maps: used to display data in a two-dimensional format, with colours indicating the magnitude of the data.
5. Tree maps: used to represent hierarchical data structures.

Data visualization is a key aspect of data analysis and communication, as it helps to reveal patterns, trends, and relationships that may not be apparent in raw data. It allows users to explore data, identify insights, and communicate findings to others in a clear and compelling way.

Effective data visualization involves choosing the right type of chart or graph to represent the data, selecting appropriate colours and labels, and ensuring that the visualization accurately and clearly represents the data. It is important to keep in mind the intended audience and the purpose of the visualization when designing it. Data visualization is used in a variety of fields, including business, science, healthcare, and social sciences. With the growth of big data and the increasing importance of data-driven decision making, data visualization has become an essential tool for analysing and communicating complex information.

### **Libraries which are widely used in the visualization process:**

* **Matplotlib:**

Matplotlib is a widely used data visualization library in Python. It provides a range of functions to create different types of plots, including line plots, scatter plots, histograms, bar graphs, and more.

### **Seaborn:**

Seaborn is another popular library that builds on top of Matplotlib and provides more advanced visualization capabilities, including heat maps, pair plots, and categorical plots.

### **Plotly:**

Plotly and Bokeh are interactive visualization libraries that allow users to create dynamic and interactive visualizations. Plotly provides a range of chart types, including scatter plots, line charts, bar charts, and more. Bokeh, on the other hand, provides tools for creating interactive visualizations that can be used in web applications.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we have used the inbuild dataset of seaborn library named as

**“penguins”**

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Import the Libraries

**Step3:** Visualize the dataset using matplotlib and seaborn libraries and seaborn inbuild dataset but only plotting a histogram with different properties of histogram.

* + Assign a variable to x to plot a univariate distribution along the x axis
  + Flip the plot by assigning the data variable to the y axis
  + Check how well the histogram represents the data by specifying a different bin width
  + define the total number of bins to use
  + Add a kernel density estimate to smooth the histogram, providing complementary information about the shape of the distribution
  + If neither x nor y is assigned, the dataset is treated as wide-form, and a histogram is drawn for each numeric column
  + can otherwise draw multiple histograms from a long-form dataset with hue mapping
  + The default approach to plotting multiple distributions is to “layer” them, but you can also “stack” them
  + Overlapping bars can be hard to visually resolve. A different approach would be to draw a step function
  + You can move even farther away from bars by drawing a polygon with vertices in the centre of each bin. This may make it easier to see the shape of the distribution, but use with caution: it will be less obvious to your audience that they are looking at a histogram
  + To compare the distribution of subsets that differ substantially in size, use independent density normalization
  + It is also possible to normalize so that each bar’s height shows a probability, proportion, or percent, which make more sense for discrete variables
  + You can even draw a histogram over categorical variables (although this is an experimental feature
  + When using a hue semantic with discrete data, it can make sense to “dodge” the levels
  + Real-world data is often skewed. For heavily skewed distributions, it is better to define the bins in log space. Compare
  + To the log-scale version
  + There are also several options for how the histogram appears. You can show unfilled bars
  + Or an unfilled step function
  + Step functions, especially when unfilled, make it easy to compare cumulative histograms
  + When both x and y are assigned, a bivariate histogram is computed and shown as a heatmap
  + It is possible to assign a hue variable too, although this will not work well if data from the different levels have substantial overlap
  + Multiple colour maps can make sense when one of the variables is discrete
  + The bivariate histogram accepts all the same options for computation as its univariate counterpart, using tuples to parametrize x and y independently
  + The default behaviour makes cells with no observations transparent, although this can be disabled
  + It is also possible to set the threshold and colormap saturation point in terms of the proportion of cumulative counts
  + To annotate the colormap, add a colorbar

### **PYTHON CODE:**

**Step1:** start

**Step2:** Import the Libraries



**Step3:** Visualize the dataset using matplotlib and seaborn libraries and seaborn inbuild dataset but only plotting a histogram with different properties of histogram.

* + Assign a variable to x to plot a univariate distribution along the x axis
  + Flip the plot by assigning the data variable to the y axis
  + Check how well the histogram represents the data by specifying a different bin width
  + define the total number of bins to use
  + Add a kernel density estimate to smooth the histogram, providing complementary information about the shape of the distribution
  + If neither x nor y is assigned, the dataset is treated as wide-form, and a histogram is drawn for each numeric column
  + can otherwise draw multiple histograms from a long-form dataset with hue mapping
  + The default approach to plotting multiple distributions is to “layer” them, but you can also “stack” them
  + Overlapping bars can be hard to visually resolve. A different approach would be to draw a step function
  + You can move even farther away from bars by drawing a polygon with vertices in the centre of each bin. This may make it easier to see the shape of the distribution, but use with caution: it will be less obvious to your audience that they are looking at a histogram
  + To compare the distribution of subsets that differ substantially in size, use independent density normalization
  + It is also possible to normalize so that each bar’s height shows a probability, proportion, or percent, which make more sense for discrete variables
  + You can even draw a histogram over categorical variables (although this is an experimental feature
  + When using a hue semantic with discrete data, it can make sense to “dodge” the levels
  + Real-world data is often skewed. For heavily skewed distributions, it is better to define the bins in log space. Compare
  + To the log-scale version
  + There are also several options for how the histogram appears. You can show unfilled bars
  + Or an unfilled step function
  + Step functions, especially when unfilled, make it easy to compare cumulative histograms
  + When both x and y are assigned, a bivariate histogram is computed and shown as a heatmap
  + It is possible to assign a hue variable too, although this will not work well if data from the different levels have substantial overlap
  + Multiple colour maps can make sense when one of the variables is discrete
  + The bivariate histogram accepts all the same options for computation as its univariate counterpart, using tuples to parametrize x and y independently
  + The default behaviour makes cells with no observations transparent, although this can be disabled
  + It is also possible to set the threshold and colormap saturation point in terms of the proportion of cumulative counts
  + To annotate the colormap, add a colorbar

























## **RESULT:**

* As we have plot different histogram to visualize the data in that we get the result that is we get the same output or same visualization of data but in a different form according to different properties of histogram.

## **CONCLUSION:**

By implementing this visualization experiment, we have successfully done the following things

* We learn how to visualize and analyse the dataset.
* What are different properties of histogram that we can use for the visualization.
* How they react differently with dataset
* We learn that what aver the plots we are using to visualize the data it may differ from the plotting type but it will always give the same results.
* Now we can plot a different histogram for different dataset to visualize and analyse them with the more efficient way and by using different properties of histogram.

**AIM:**

# **EXPERIMENT NO. 4A**

Partition the data set, for example 80% of the records are included in the training data set and 20% are included in the test data set. Use a bar graph to confirm your proportions.

## **THEORY:**

**Data splitting: -**

Data splitting is the process of dividing a dataset into two or more subsets for the purpose of training and evaluating a machine learning model. This is typically done to prevent overfitting, which is when a model learns the training data too well and performs poorly on new data.

In Python, you can split a dataset into training and testing subsets using the train\_test\_split() function from the sklearn.model\_selection module.

Here is an example:



## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we have used the **‘loan\_data\_set.csv’** dataset and attributes are as follows:

[[“Loan\_ID”, “Gender”, “Married”, “Dependents”, “Education”, “Self\_Employed”, “ApplicantIncome”, “CoapplicantIncome”, “LoanAmount”, “Loan\_Amount\_Term”, “Credit\_History”, “Property\_Area”, “Loan\_Status”]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Import the Libraries

**Step3:** Import the dataset

**Step4:** pre-process the dataset by taking all attributes in 'X' and last attribute in 'y'

**Step5:** Split the data

**Step6:** Plot the graph

### **PYTHON CODE:**

**Step1:** start

**Step2:** Import the Libraries



**Step3:** Import the dataset



**Step4:** pre-process the dataset by taking all attributes in 'X' and last attribute in 'y'

|  |
| --- |

**Step5:** Split the data

|  |
| --- |

**Step6:** Plot the graph



## **RESULT:**

* When you evaluate the predictive performance of any model, it is essential that the process should unbiased. Using ***train\_test\_split()*** from the data science library scikit-learn, you split dataset into subsets that minimize the potential for bias in our evaluation and validation process.

## **CONCLUSION:**

By implementing this visualization experiment, we have successfully done the following things

* We learn how to split the data using train\_test\_split() from the data science library scikit- learn.
* By splitting the data into subset, we can analyse our dataset whether it is predicting or giving proper and expected value.
* if it is giving proper values then we can proceed for testing else we can do the change into the data and again go for testing.
* This will reduce the chance of failure of the models and its predictions or outputs.

# **EXPERIMENT NO 4B**

## **AIM:**

Validate your partition by performing a two‐sample Z‐test.

## **THEORY:**

**Data Modelling:**

The process of developing a mathematical representation of a system or occurrence in the real world is known as data modelling. It entails determining the pertinent variables and their connections before developing a model that may be used to forecast outcomes or uncover new information about the system or phenomena.

Data models come in a wide variety, including:

1. **Models based on statistics:**

These models employ statistical methods to determine links between variables and forecast future results.

1. **Models that use machine learning:**

These models employ algorithms to discover patterns in data and make predictions.

1. **Models for simulation:**

These models simulate a system or process through time, enabling the assessment of various hypotheses and outcomes.

1. **Models for optimization:**

These models employ mathematical methods to determine the best course of action in the presence of limitations and goals.

As it enables us to better understand the data and the underlying relationships between variables, data modelling is a crucial step in the data analysis process. Additionally, it helps us to provide more useful insights and more precise predictions.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we will take the following:

**mean\_iq** = 110

**sd\_iq** = 15/math.sqrt(50)

**alpha** =0.05

**null\_mean** =100

**data** = sd\_iq\*randn(50)+mean\_iq

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Import the Libraries **Step3:** Generate random array **Step4:** Print mean and sd **Step5:** Performing a test

### **PYTHON CODE:**

**Step1:** start



**Step2:** Import the Libraries



**Step3:** Generate random array

|  |
| --- |

**Step4:** Print mean and sd

|  |
| --- |

**Step5:** Performing a test

|  |
| --- |

## **RESULT:**

* With the help of z-test we have found of whether the null hypothesis will accepted or rejected and according to input we get the result as our null hypothesis is rejected.

## **CONCLUSION:**

By implementing this z-test experiment, we have successfully done the following things

* We learn how to perform z-test by taking random values.
* We also learn that if p-value is greater that alpha we accept the null hypothesis where as if it is less then we reject the null hypothesis.

# **EXPERIMENT NO. 5A**

## **AIM:**

Normality Tests : Shapiro-Wilk Test

## **THEORY:**

**Shapiro-Wilk Test:**

The Shapiro-Wilk test is a statistical test used to check the normality of a data sample. It was developed by Samuel Shapiro and Martin Wilk in 1965. The test is particularly useful when the sample size is small (less than 50).

The Shapiro-Wilk test works by comparing the observed distribution of the sample to what would be expected in a normal distribution. It calculates a test statistic (W) based on the differences between the observed values and the expected values and compares it to a critical value to determine whether the sample comes from a normal distribution.

Let’s understand it with an example,



The test statistic (W) is 0.8909, and the p-value is 0.1261. Since the p-value is greater than the significance level of 0.05, we fail to reject the null hypothesis and conclude that the data is normally distributed. In other words, there is not enough evidence to suggest that the data does not come from a normal distribution.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

The primary goal is to establish whether an assumption is true for a specific data set in order to evaluate whether non-parametric tests or other changes to conventional tests should be used. We also utilize the Shapiro Wilk test to find outliers that may be impacting the data distribution.

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Define the hypotheses

**Step3:** Shapiro-Wilk test on the normally distributed sample in Python

**Step4:** Output Interpretation

**Step5:** Shapiro-Wilk test on not normally distributed sample in Python

**Step6:** Output Interpretation

### **PYTHON CODE:**

**Step1:** start

|  |
| --- |

**Step2:** Define the hypotheses

|  |
| --- |

**Step3:** Shapiro-Wilk test on the normally distributed sample in Python



**Step4:** Output Interpretation



**Step5:** Shapiro-Wilk test on not normally distributed sample in Python

|  |
| --- |

**Step6:** Output Interpretation

|  |
| --- |

## **RESULT:**

* Since in the above example, the p-value is 0.73 which is more than the threshold(0.05) which is the alpha(0.05) then we fail to reject the null hypothesis i.e. we do not have sufficient evidence to say that sample does not come from a normal distribution.
* Since in the above example, the p-value is 0.0001 which is less than the alpha(0.05) then we reject the null hypothesis i.e. we have sufficient evidence to say that sample does not come from a normal distribution.

## **CONCLUSION:**

By implementing this z-test experiment, we have successfully done the following things

* We learn how to perform Shapiro-Wilk test on not normally distributed sample.
* We also learn that if p-value is greater that threshold value we accept the null hypothesis where as if it is less then we reject the null hypothesis.

**AIM:**

# **EXPERIMENT NO. 5B**

Analysis of Variance Test (ANOVA)

## **THEORY:**

### **Understanding the ANOVA Test:**

We can think of an Analysis of Variance Test, also known as ANOVA, to generalize the T-tests for multiple groups. Generally, we use the independent T-test in order to compare the mean of the state between two groups. We use ANOVA Test whenever we need a comparison of the means of the state between more than two groups.

ANOVA test checks whether a difference in the average somewhere in the model or not (checking whether there was an overall effect or not); however, this method does not tell us the spot of the difference (if there is one). We can find the spot of the difference between the group by conducting the post hoc tests.

However, in order to perform any tests, we first must define the null and alternate hypotheses: Null Hypothesis: There is no noteworthy difference between the groups.

Alternate Hypothesis:

There is a noteworthy difference between the groups. We can perform an ANOVA Test by comparing two types of variations. The First variation is between the sample means and the other one within each of the samples. The formula shown below describes one-way ANOVA Test statistics.

The output of the ANOVA formula, the F statistic (also known as the F-ratio), enables the analysis of the multiple sets of data in order to determine the variability among the samples and within samples.

Usually, if the p-value belonging to the F is smaller than 0.05, then the null hypothesis is excluded, and the alternative hypothesis is maintained. In the case of the null hypothesis rejection, we can say that the means of all the sets/groups are not equal.

### **ANOVA Test Assumptions**

Before performing an ANOVA test, we must make certain assumptions, as shown below:

We can obtain observations randomly and independently from the population defined by the factor levels. The data for every level of the factor is distributed generally. Case Independent:

The sample cases must be independent of each other. Variance Homogeneity: Homogeneity signifies that the variance between the group needs to be around equal. We can test the assumption of variance homogeneity with the bits of help of tests like the Brown-Forsythe Test or Levane’s Test. We can also test the Normality of the score distributions with the help of

histograms, the kurtosis, or skewness values, or with the help of tests like Kolmogorov-Smirnov, Shapiro-Wilk, or Q-Q plot. We can also determine the assumption of independence from the

study design.

It is quite noteworthy to notice that the ANOVA test is not robust to violating the assumption of independence. This is to inform that even if someone tries to violate the assumptions of Normality or homogeneity, they can conduct the test and trust the findings.

Nevertheless, the outputs of the ANOVA test are unacceptable if the assumption of independence is dishonoured. Usually, the analysis, along with the violations of homogeneity, is

considered robust if we have equal-sized groups. Resuming the ANOVA test along with violations of Normality is usually fine if we have a large sample size.

Understanding the Types of ANOVA Tests

The ANOVA Tests can be classified into three major types. These types are shown below:

1. One-Way ANOVA Test
2. Two-Way ANOVA Test
3. n-Way ANOVA Test

### **One-Way ANOVA Test**

An Analysis of Variance Test that has only one independent variable is known as the One-way ANOVA Test. For instance, a country can assess the differences in the cases of Coronavirus, and a Country can have multiple categories for comparison.

### **Two-Way ANOVA Test**

An Analysis of Variance Test that has two independent variables is known as a Two-way ANOVA test. This test is also known as Factorial ANOVA Test.

For example, expanding the above example, a two-way ANOVA can examine the difference in the cases of Coronavirus (the dependent variable) by Age Group (the first independent

variable) and Gender (the second independent variable). The two-way ANOVA can be utilized in order to examine the interaction among these two independent variables. Interactions

denote that the differences are uneven across all classes of the independent variables. Suppose that the old age group may have higher cases of Coronavirus overall compared to the young age group; however, this difference could vary in countries in Europe compared to countries in Asia.

### **n-Way ANOVA Test**

An Analysis of Variance Test is considered an n-way ANOVA Test if a researcher uses more than two independent variables. Here n represents the number of independent variables we have. This Test is also known as MANOVA Test. For example, we can examine potential differences in cases of Coronavirus using independent variables like Country, Age group, Gender, Ethnicity, and a lot more simultaneously.

An ANOVA Test will provide us a single (univariate) F-value; however, a MANOVA Test will provide us a multivariate F-value.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we are using **Diet\_R.csv** dataset with the following attributes [[“Person”, “gender”, “Age”, “Height”, “pre.weight Diet”, “weight6weeks”]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Importing required libraries **Step3:** Define the hypotheses **Step4:** Load the Data

**Step5:** Understanding the Dataset

**Step6:** Understanding the distribution of Weight

**Step7:** We can also plot a distribution plot for each Gender in the dataset.

**Step8:** Now, we will calculate the mean, median, non-zero count, and standard deviation according to the 'gender'

**Step9:** As we can observe, we have estimated the required statistical measurements based on gender. We can also classify these statistical measurements based on gender as well as diet.

**Step10:** Performing the one-way ANOVA Test **Step11:** Performing the Two-way ANOVA Test **Step12:** Performing the n-way ANOVA Test

### **PYTHON CODE:**

**Step1:** start

|  |
| --- |

|  |
| --- |
|  |
|  |

**Step2:** Importing required libraries



**Step3:** Define the hypotheses

|  |
| --- |

**Step4:** Load the Data



**Step5:** Understanding the Dataset

|  |
| --- |
|  |

**Step6:** Understanding the distribution of Weight

|  |
| --- |

**Step7:** We can also plot a distribution plot for each Gender in the dataset.

|  |
| --- |
|  |
|  |



**Step8:** Now, we will calculate the mean, median, non-zero count, and standard deviation according to the 'gender'

|  |
| --- |

**Step9:** As we can observe, we have estimated the required statistical measurements based on gender. We can also classify these statistical measurements based on gender as well as diet.

|  |
| --- |

**Step10:** Performing the one-way ANOVA Test

|  |
| --- |

**Step11:** Performing the Two-way ANOVA Test

|  |
| --- |

**Step12:** Performing the n-way ANOVA Test

|  |
| --- |

## **RESULT:**

* Since in the above example, the p-value is 0.73 which is more than the threshold(0.05) which is the alpha(0.05) then we fail to reject the null hypothesis i.e., we do not have sufficient evidence to say that sample does not come from a normal distribution.
* Since in the above example, the p-value is 0.0001 which is less than the alpha(0.05) then we reject the null hypothesis i.e., we have sufficient evidence to say that sample does not come from a normal distribution.

## **CONCLUSION:**

By implementing this z-test experiment, we have successfully done the following things

* We learn how to perform Shapiro-Wilk test on not normally distributed sample.
* We also learn that if p-value is greater that threshold value we accept the null hypothesis where as if it is less then we reject the null hypothesis.

# **EXPERIMENT NO. 6**

## **AIM:**

Perform Logistic Regression to find out relation between variables.

## **THEORY:**

**Logistic Regression** is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

### **Logistic Regression Assumptions**

* Binary logistic regression requires the dependent variable to be binary.
* For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
* Only the meaningful variables should be included.
* The independent variables should be independent of each other. That is, the model should have little or no multi-collinearity.
* The independent variables are linearly related to the log odds.
* Logistic regression requires quite large sample sizes.

### **Over-sampling using SMOTE**

With our training data created, I’ll up-sample the no-subscription using the SMOTE algorithm (Synthetic Minority Oversampling Technique). At a high level, SMOTE:

* Works by creating synthetic samples from the minor class (no-subscription) instead of creating copies.
* Randomly choosing one of the k-nearest-neighbours and using it to create a similar, but randomly tweaked, new observations.

### **Recursive Feature Elimination**

Recursive Feature Elimination (RFE) is based on the idea to repeatedly construct a model and choose either the best or worst performing feature, setting the feature aside and then repeating the process with the rest of the features. This process is applied until all features in the dataset are exhausted. The goal of RFE is to select features by recursively considering smaller and smaller sets of features.

### **Compute precision, recall, F-measure and support**

The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier to not label a sample as positive if it is negative.

The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

The F-beta score can be interpreted as a weighted harmonic mean of the precision and recall, where an F-beta score reaches its best value at 1 and worst score at 0.

The F-beta score weights the recall more than the precision by a factor of beta. beta = 1.0 means recall and precision are equally important.

The support is the number of occurrences of each class in y\_test.

### **ROC Curve**

The receiver operating characteristic (ROC) curve is another common tool used with binary classifiers. The dotted line represents the ROC curve of a purely random classifier; a good classifier stays as far away from that line as possible (toward the top-left corner).

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we are using **loan\_data\_set.csv** dataset with the following attributes [[“Loan\_ID” “Gender” “Married” “Dependents” “Education” “Self\_Employed” “ApplicantIncome” “CoapplicantIncome” “LoanAmount” “Loan\_Amount\_Term” “Credit\_History” “Property\_Area” “Loan\_Status”]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Importing required libraries **Step3:** Define the hypotheses **Step4:** Load the Data

**Step5:** Understanding the Dataset **Step6:** Handel the data(Missing values) **Step7:** Handling the categorical data

**Step8:** Performing EDA (Exploratory Data Analysis) to select the feature/s which affects the target attribute/s

**Step9:** Features and target

**Step10:** Training and testing the data

**Step11:** Model selection

**Step12:** Predict the output/result

### **PYTHON CODE:**

**Step1:** start



**Step2:** Importing required libraries



**Step3:** Load the Data



**Step4:** Understanding the Dataset



**Step5:** Handel the data(Missing values)



**Step6:** Handling the categorical data

**Step7:** Performing EDA (Exploratory Data Analysis) to select the feature/s which affects the target attribute/s



|  |
| --- |
|  |

**Step8:** Features and target



**Step9:** Training and testing the data

|  |
| --- |

**Step10:** Model selection

|  |
| --- |

**Step11:** Predict the output/result

|  |
| --- |

## **RESULT:**

Accuracy of logistic regression classifier on test set is 0.84.

This classifier is a good fit for the data as it stays well away from the ROC line in the graph.

## **CONCLUSION:**

We have successfully learned how to:

* Train and test the logistic regression classifier model on a dataset
* Check the accuracy of the logistic regression classifier model
* Plot the ROC Curve for the logistic regression classifier model

# **EXPERIMENT NO. 7**

## **AIM:**

Choose classifier for classification problem.

## **THEORY:**

In machine learning, classification is the problem of identifying to which of a set of categories (sub- populations) a new observation belongs, based on a training set of data containing observations (or instances) whose category membership is known. Couple examples of classification problems are:

(a) deciding whether a received email is a spam or an organic e-mail; (b) assigning a diagnosis of a patient based on observed characteristics of the patient (age, blood pressure, presence, or absence of certain symptoms, etc.)

### **Data Pre-Processing**

Before we can begin to create our first model we first need to load and pre-process. This step ensure that our model will receive a good data to learn from, as they said “a model is only as good as its data”.

**Class Distribution:** Another important thing to make sure before feeding our data into the model is the class distribution of the data. In our case where the expected class are divided into two outcomes, ‘yes’ and ‘no’, a class distribution of 50:50 can be considered ideal.

**Missing Values:** Last thing to check before moving on is missing values. In some case our data might have missing values in some column, this can be caused some reasons such as human error. We can use the is\_null() function from Pandas to check for any missing data and then use the sum() function to see the total of missing values in each column.

**Scale Numeric Data:** Next up, we will scale our numerical data to avoid outlier presence that can significantly affect our model. Using StandardScaler() function from sklearn we can scale each our columns that contains numerical data.

**Encoding Categorical Data:** Same as the numerical data, we also need to pre-process our categorical data from words to number to make it easier for the computer to understands. To do this we will use OneHotEncoder() provided by sklearn.

**Split Dataset for Training and Testing:** To finish up our data pre-processing steps we will split our data into two dataset, training, and testing. In this case because we have enough data, we will split the data with ratio of 80:20 for training and testing respectively.

### **Modelling**

After making sure our data is good and ready, we can continue to building our model. In this notebook we will try to build 4 different models with different algorithm. In this step we will create a baseline model for each algorithm using the default parameters set by sklearn and after building all 4 of our models we will compare them to see which works best for our case. To evaluate our model, we will use the confusion matrix as our base for the evaluation.

1. **Decision Tree:** Decision tree is a tree shaped diagram used to determine a course of action. Each branch of the tree represents a possible decision, occurrence, or reaction.
2. **Random Forest:** Random Forest or Random Decision Forest is a method that operates by constructing multiple decision trees during training phases. The decision of most of the trees is chosen as final decision.
3. **Naive Bayes:** Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable.
4. **K-Nearest Neighbours:** K-Nearest Neighbours (KNN) classify new data by finding k-number of closest neighbours from the training data and then decide the class based on the majority of its neighbours.

### **Model Comparison**

After building all our model, we can now compare how well each model perform. To do this we will create two charts, first is a grouped bar chart to display the value of accuracy, precision, recall, f1, and kappa score of our model, and second a line chart to show the AUC of all our models.

### **Model Optimisation**

On the next part, we will try to optimise our RandomForest model by tuning the hyper parameters available from the scikit-learn library. After finding the optimal parameters we will then evaluate our new model by comparing it against our base line model before.

**Tuning Hyperparameter with GridSearchCV:** We will provide our baseline model (named rf\_grids), scoring method (in our case we will use recall as explained before), and various parameters value we want to try with our model. The GridSearchCV function will then iterate through each parameters combination to find the best scoring parameters. This function also allows us to use cross validation to train our model, where on each iteration our data will be divided into 5 (the number are adjustable from the parameter) fold. The models then will be trained on 4/5-fold of the data leaving the final fold as validation data, this process will be repeated for 5 times until all our folds are used as validation data.

**Evaluating Optimised Model:** After finding the best parameter for the model we can access the best\_estimator\_ attribute of the GridSearchCV object to save our optimised model into variable called best grid. We will calculate the 6-evaluation metrics using our helper function to compare it with our base model on the next step.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we are using **loan\_data\_set.csv** dataset with the following attributes [[“Loan\_ID” “Gender” “Married” “Dependents” “Education” “Self\_Employed” “ApplicantIncome” “CoapplicantIncome” “LoanAmount” “Loan\_Amount\_Term” “Credit\_History” “Property\_Area” “Loan\_Status”]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Importing required libraries

**Step3:** Load the Data

**Step4:** Understanding the Dataset **Step5:** Handel the data(Missing values) **Step6:** Class Distribution

**Step7:** Scale Numeric Data

**Step8:** Handling the categorical data

**Step9:** Features and target

**Step10:** Split Dataset for Training and Testing

**Step11:** model and evaluation algorithm function define

**Step12:** Algorithms

* + 1. Decision Tree
    2. Random Forest
    3. Naive Bayes
    4. K-Nearest Neighbours **Step13:** Create a grouped bar graph **Step14:** Prediction

### **PYTHON CODE:**

**Step1:** start

|  |
| --- |

**Step2:** Importing required libraries

|  |
| --- |

**Step3:** Load the Data

|  |
| --- |

**Step4:** Understanding the Dataset

|  |
| --- |

**Step5:** Handel the data(Missing values)

|  |
| --- |

**Step6:** Class Distribution

|  |
| --- |

**Step7,8:** Scale Numeric Data



**Step9:** Features and target

|  |
| --- |

**Step10:** Split Dataset for Training and Testing

|  |
| --- |

**Step11:** model and evaluation algorithm function define

|  |
| --- |

**Step12:** Algorithms

* + 1. Decision Tree
    2. Random Forest
    3. Naive Bayes
    4. K-Nearest Neighbours

|  |
| --- |
|  |

|  |
| --- |

**Step13:** Create a grouped bar graph





**Step14:** Prediction

|  |
| --- |

## **RESULT:**

The result show that our optimised performed little bit better than the original model. The optimised models show an increase in 4 out of the 6 metrics but perform worse in the other metrics, especially the recall with -3.89% decrease. Because we want to focus on predicting as many actual positive values as possible, we should stick with our original model for the prediction because it has higher recall score.

## **CONCLUSION:**

We have successfully learned how to:

* Train and test the different classifier model like random forest decision tree, naive bayes, KNN
* Check the accuracy of the all-regression classifier model
* Plot the ROC Curve for the logistic regression classifier model

# **EXPERIMENT NO. 8**

## **AIM:**

Clustering algorithms for unsupervised classification.

## **THEORY:**

**Unsupervised learning** is a machine learning algorithm that searches for previously unknown patterns within unlabelled data sets. The most prominent methods of unsupervised learning are cluster analysis and principal component analysis.

### **Clustering**

In clustering, the data is divided into several groups with similar traits.



In the image above, the left is raw data without classification, while the right is clustered based on its features. When an input is given which is to be predicted then it checks in the cluster it belongs to base on its features, and the prediction is made.

### **K-MEANS CLUSTERING IN PYTHON**

K-means clustering is an iterative unsupervised clustering algorithm that aims to find local maxima in each iteration. Initially, desired number of clusters are chosen. We program the algorithm to group the data into three classes by passing the parameter “n\_clusters” into our k-means model. Randomly, three points (inputs) are assigned into three clusters. Based on the centroid distance between each point, the next given inputs are segregated into respected clusters and the centroids are re-computed for all the clusters.

### **HIERARCHICAL CLUSTERING**

As its name implies, hierarchical clustering is an algorithm that builds a hierarchy of clusters. This algorithm begins with all the data assigned to a cluster, then the two closest clusters are joined into the same cluster. The algorithm ends when only a single cluster is left.

### **DBSCAN CLUSTERING**

Density-based spatial clustering of applications with noise, or DBSCAN, is a popular clustering algorithm used as a replacement for k-means in predictive analytics. To run it does not require an input for the number of clusters but it does need to tune two other parameters.

The scikit-learn implementation provides a default for the eps and min\_samples parameters, but you’re generally expected to tune those. The eps parameter is the maximum distance between two data points to be considered in the same neighbourhood. The min\_samples parameter is the minimum amount of data points in a neighbourhood to be considered a cluster.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

We’ll use the Iris dataset to make predictions. The dataset contains a set of 150 records under four attributes: petal length, petal width, sepal length, sepal width, and three iris classes: setosa, virginica and versicolor. We'll feed the four features of our flower to the unsupervised algorithm and it will predict which class the iris belongs to.

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** Start

**Step2:** Importing required libraries

**Step3:** Load the Data **Step4:** Pre-Processing **Step5:** K-Means Clustering

**Step6:** Hierarchical Clustering **Step7:** t-SNE Clustering **Step8:** DBSCAN Clustering

### **PYTHON CODE:**

**Step1:** start

**Step2:** Importing required libraries

**Step3:** Load the Data

**Step4:** Pre-Processing



**Step5:** K-Means Clustering

|  |
| --- |



**Step7:** t-SNE Clustering



**Step8:** DBSCAN Clustering



## **RESULT:**

The results of the four clustering algorithms that are implemented are represented as the output of the python code. In case of K-Means, the output is in the form of numerical data. The hierarchical clustering produces a graphical output as a hierarchy. In case of t-SNE and DBSCAN, the clusters are formed using a scatter plot.

## **CONCLUSION:**

We have successfully implemented the below clustering algorithms on a dataset in Python:

* K-Means Clustering
* Hierarchical Clustering
* T-SNE Clustering
* DBSCAN Clustering

# **EXPERIMENT NO. 9**

## **AIM:**

Using any machine learning techniques using available data set to develop a recommendation system.

## **THEORY:**

A recommendation system is an artificial intelligence or AI algorithm, usually associated with machine learning, that uses Big Data to suggest or recommend additional products to consumers. These can be based on various criteria, including past purchases, search history, demographic information, and other factors.

### **Book recommender system**

A book recommendation system is a type of recommendation system where we have to recommend similar books to the reader based on his interest. The books recommendation system is used by online websites which provide ebooks like google play books, open library, good Read's, etc. The main objective is to create a book recommendation system for users. Recommender systems are really critical in some industries as they can generate a huge amount of income when they are efficient or also be a way to stand out significantly from competitors.

### **Types Of Recommendation System**

A recommendation system is usually built using 3 techniques which are content-based filtering, collaborative filtering, and a combination of both.

* Content-Based Filtering: The algorithm recommends a product that is similar to those which used as watched. In simple words, In this algorithm, we try to find finding item look alike. For example, a person likes to watch Sachin Tendulkar shots, so he may like watching Ricky Ponting shots too because the two videos have similar tags and similar categories.
* Collaborative-based Filtering: Collaborative based filtering recommender systems are based on past interactions of users and target items. In simple words here, we try to search for the look-alike customers and offer products based on what his or her lookalike has chosen. Let us understand with an example. X and Y are two similar users and X user has watched A, B, and C movie. And Y user has watched B, C, and D movie then we will recommend A movie to Y user and D movie to X user.
* Hybrid Filtering Method: It is basically a combination of both the above methods. It is a too complex model which recommends product based on your history as well based on similar users like you.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we are using **tmdb\_5000\_credits.csv and tmdb\_5000\_movies.csv**

dataset with the following attributes [[ 'id', 'tittle', 'cast', 'crew' ]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Import the libraries

**Step3:** Load the Data

**Step4:** Choose and merge the column

**Step5:** Description of data

**Step6:** Calculation based on the IMDB formula **Step7:** Sort movies based on score calculated above **Step8:** popularity graph plot

**Step9:** Content Based Filtering

* + 1. Plot description based Recommender

**Step10:** Calling the function

### **PYTHON CODE:**

**Step1:** start

**Step2:** Import the libraries

|  |
| --- |

**Step3:** Load the Data

|  |
| --- |

**Step4:** Choose and merge the column



**Step5:** Description of data

|  |
| --- |

**Step6:** Calculation based on the IMDB formula

|  |
| --- |

**Step7:** Sort movies based on score calculated above

|  |
| --- |

**Step8:** popularity graph plot

|  |
| --- |

**Step9:** Content Based Filtering

* + 1. Plot description based Recommender

**Step10:** Calling the function

|  |
| --- |

## **Conclusion:**

* Recommender system has the ability to predict whether a particular user would prefer an item or not based on the user's profile. Recommender systems are beneficial to both service providers and users
* A book recommendation system can take into account many parameters like book content and book quality by filtering user reviews.

# **EXPERIMENT NO 10**

## **AIM:**

Exploratory data analysis using Apache Spark

## **THEORY:**

Apache Spark is a unified analytics engine for large-scale data processing with built-in modules for SQL, streaming, machine learning, and graph processing. Spark can run on Apache Hadoop, Apache Mesos, Kubernetes, on its own, in the cloud—and against diverse data

sources.

Spark enhances machine learning because data scientists can focus on the data problems they really care about while transparently leveraging the speed, ease, and integration of Spark's unified platform.

### **PySpark**

PySpark is the Python API for Apache Spark, an open source, distributed computing framework and set of libraries for real-time, large-scale data processing. If you are already familiar with Python and libraries such as Pandas, then PySpark is a good language to learn to

create more scalable analyses and pipelines.

Apache Spark is basically a computational engine that works with huge sets of data b processing them in parallel and batch systems. Spark is written in Scala, and PySpark was released to support the collaboration of Spark and Python. In addition to providing an API for

Spark, PySpark helps you interface with Resilient Distributed Datasets (RDDs) by leveraging the Py4j library.

The key data type used in PySpark is the Spark dataframe. This object can be thought of as a table distributed across a cluster, and has functionality that is like dataframes in R and Pandas. If you want to do distributed computation using PySpark, then you will need to perform operations on Spark dataframes and no other Python data types.

One of the key differences between Pandas and Spark dataframes is eager versus lazy execution. In PySpark, operations are delayed until a result is requested in the pipeline. For example, you can specify operations for loading a data set from Amazon S3 and applying several transformations to the dataframe, but these operations will not be applied immediately. Instead, a graph of transformations is recorded, and once the data are actually needed, for example when writing the results back to S3, then the transformations are applied as a single pipeline operation. This approach is used to avoid pulling the full dataframe into memory, and enables more effective processing across a cluster of machines. With Pandas dataframes, everything is pulled into memory, and every Pandas operation is applied immediately.

Py4J is a popular library which is integrated within PySpark and allows Python to dynamically interface with JVM (Java Virtual Machine) objects. PySpark features quite a few libraries for writing efficient programs. Furthermore, there are various external libraries that are also compatible, including:

* PySparkSQL - A PySpark library to apply SQL-like analysis on a huge amount of structured or semi-structured data. You can also use SQL queries with PySparkSQL.

## **PROBLEM SATEMENT / DEFINATION / DATASET DESCRIPTION:**

To perform this experiment, we are using **Life\_Expectancy.csv** dataset with the following attributes [[“Country”, “Year”, “Status”, “Life”, “xpectancy”, “Adult Mortality”, “Population”,…..ETC]]

## **SOLUTION:**

### **ALGORITHM:**

**Step1:** start

**Step2:** Installation of libraries **Step3:** Import the libraries **Step4:** Creating a Spark Session **Step5:** Data Visualization **Step6:** Histogram

**Step7:** Line Plot **Step8:** Bar Plot **Step9:** Pie Chart **Step10:** Box Plot

### **PYTHON CODE:**

**Step1:** start

**Step2:** Installation of libraries



**Step3:** Import the libraries

|  |
| --- |

**Step4:** Creating a Spark Session



**Step5:** Data Visualization

|  |
| --- |





**Step6:** Histogram

|  |
| --- |

|  |
| --- |

|  |
| --- |
|  |

|  |
| --- |

|  |
| --- |

**Step7:** Line Plot



|  |
| --- |

|  |
| --- |

**Step8:** Bar Plot

|  |
| --- |
|  |

|  |
| --- |

**Step9:** Pie Chart

|  |
| --- |

|  |
| --- |

**Step10:** Box Plot

|  |
| --- |
|  |

## **Conclusion:**

Using PySpark, integration and work with RDDs in Python programming language becomes easy. There are numerous features that make PySpark such an amazing framework when it comes to working with huge dataset