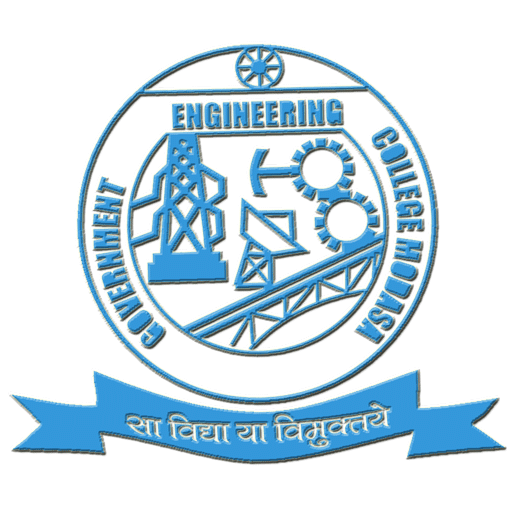
**A Laboratory Manual for**

Python for Data Science

**(3150713)**

**B.E. Semester 5**

**(Computer Engineering)**



**Directorate of Technical Education, Gandhinagar, Gujarat**

**Government Engineering College Modasa**

**Certificate**

This is to certify that Mr./Ms. Bharad Mihir Dipakbhai Enrollment No.210160107012 of B.E. Semester Computer Engineering of this Institute (GTU Code:016) has satisfactorily completed the Practical / Tutorial work for the subject Python for Data Science (3150713) for the academic year 2023-24.

Place: GEC Modasa

Date: 08/11/2003

**Name and Sign of Faculty member**

**Head of the Department**

**Preface**

Main motto of any laboratory/practical/field work is for enhancing required skills as well as creating ability amongst students to solve real time problem by developing relevant competencies in psychomotor domain. By keeping in view, GTU has designed competency focused outcome-based curriculum for engineering degree programs where sufficient weightage is given to practical work. It shows importance of enhancement of skills amongst the students and it pays attention to utilize every second of time allotted for practical amongst students, instructors and faculty members to achieve relevant outcomes by performing the experiments rather than having merely study type experiments. It is must for effective implementation of competency focused outcome-based curriculum that every practical is keenly designed to serve as a tool to develop and enhance relevant competency required by the various industry among every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual is designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual students can go through the relevant theory and procedure in advance before the actual performance which creates an interest and students can have basic idea prior to performance. This in turn enhances pre-determined outcomes amongst students. Each experiment in this manual begins with competency, industry relevant skills, course outcomes as well as practical outcomes (objectives). The students will also achieve safety and necessary precautions to be taken while performing practical.

This manual also provides guidelines to faculty members to facilitate student centric lab activities through each experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve the outcomes. It also gives an idea that how students will be assessed by providing rubrics.

Data Science is about data gathering, analysis and decision-making. Data Science is about finding patterns in data, through analysis, and make future predictions. By using Data Science, companies are able to make:

* Better decisions (should we choose A or B)
* Predictive analysis (what will happen next?)
* Pattern discoveries (find pattern, or maybe hidden information in the data)

Data Science is used in many industries in the world today, e.g. banking, consultancy, healthcare, and manufacturing. Python is an open-source, interpreted, high-level language and provides a great approach to data science, machine learning, and research purposes. It is one of the best languages for data science to use for various applications & projects. When it comes to dealing with mathematical, statistical, and scientific functions, [Python](https://www.geeksforgeeks.org/python-programming-language/learn-python-tutorial/) has great utility.

Utmost care has been taken while preparing this lab manual however always there is chances of improvement. Therefore, we welcome constructive suggestions for improvement and removal of errors if any.

**Practical – Course Outcome matrix**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Course Outcomes (COs):**  1. Apply various Python data structures to effectively manage various types of data.  2. Explore various steps of data science pipeline with role of Python.  3. Design applications applying various operations for data cleansing and transformation.  4. Use various data visualization tools for effective interpretations and insights of data.  5. Perform data Wrangling with Scikit-learn applying exploratory data analysis. | | | | | | |
| **Sr. No.** | **Objective(s) of Experiment** | **CO1** | **CO2** | **CO3** | **CO4** | **CO5** |
| 1. | Develop a program to understand the control structures of python**.** | √ |  |  |  |  |
| 2. | Develop a program to learn different types of structures (list, dictionary, tuples) in python. | √ |  |  |  |  |
| 3. | Develop a program that reads a .csv dataset file using Pandas library and display the following content of the dataset.  a) First five rows of the dataset  b) Complete data of the dataset  c) Summary or metadata of the dataset. | √ | √ |  |  |  |
| 4. | Develop a program that shows application of slicing and dicing over the rows and columns of the dataset. | √ | √ |  |  |  |
| 5. | Develop a program that shows usage of aggregate function over the input dataset. a) describe b) max c) min d) mean e) median f) count g) std h) Corr | √ | √ |  |  |  |
| 6. | Develop a program that applies split and merge operations on the datasets. | √ | √ |  |  |  |
| 7. | Develop a program that shows the various data cleaning tasks over the dataset. a) Identifying the null values. b) Identifying the empty values.  c) Identifying the incorrect timestamp | √ | √ | √ |  |  |
| 8. | Develop a program that shows usage of following NumPy array operations: a) any() b) all() c) isnan() d) isinf() e) isfinite() f) isinf() g) zeros() h) isreal() i) iscomplex() j) isscalar() k) less() l) greater() m) less\_equal() n) greater\_equal() |  | √ |  |  |  |
| 9. | Develop a program that shows usage of following NumPy library vector functions. a) arrange() b) reshape() c) linspace() d) randint() e) dot() |  | √ |  |  |  |
| 10. | Write a program to display below plot using matplotlib libraryFor Values of X:[1,2,3,...,49], Values of Y (thrice ofX):[3,6,9,12,...,144,147] |  |  |  | √ |  |
| 11. | Write a program to display below bar plot using matplotlib library For value  Languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']  popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7] |  |  |  | √ |  |
| 12. | Write a program to display below bar plot using matplotlib library For below data display pie plot  languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']  popuratity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]  colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b"] |  |  |  | √ |  |
| 13. | Write a program to display below bar plot using matplotlib library For 200 random points for both X and Y display scatter plot |  |  |  | √ |  |
| 14. | Develop a program that reads .csv file from the url: (https://github.com/chris1610/pbpython/blob/master/data/sample salesv3.xlsx?raw=true) and plot the data of the dataset stored in the .csv file. |  |  | √ | √ |  |
| 15. | Write a text classification pipeline using a custom preprocessor and CharNGramAnalyzer using data from Wikipedia articles as a training set.   * Evaluate the performance on some held out test sets. |  |  | √ | √ | √ |
| 16. | Write a text classification pipeline to classify movie reviews as either positive or negative.   * Find a good set of parameters using grid search. * Evaluate the performance on a held out test set. |  |  | √ | √ | √ |

**Industry Relevant Skills**

The following industry relevant competency are expected to be developed in the student by undertaking the practical work of this laboratory.

1. Programming Languages
2. Mathematics, Statistical Analysis, and Probability
3. Data Mining
4. Machine Learning and AI
5. Data Visualization

**Guidelines for Faculty members**

1. Teacher should provide the guideline with demonstration of practical to the students with all features.
2. Teacher shall explain basic concepts/theory related to the experiment to the students before starting of each practical
3. Involve all the students in performance of each experiment.
4. Teacher is expected to share the skills and competencies to be developed in the students and ensure that the respective skills and competencies are developed in the students after the completion of the experimentation.
5. Teachers should give opportunity to students for hands-on experience after the demonstration.
6. Teacher may provide additional knowledge and skills to the students even though not covered in the manual but are expected from the students by concerned industry.
7. Give practical assignment and assess the performance of students based on task assigned to check whether it is as per the instructions or not.
8. Teacher is expected to refer complete curriculum of the course and follow the guidelines for implementation.

**Instructions for Students**

1. Students are expected to carefully listen to all the theory classes delivered by the faculty members and understand the COs, content of the course, teaching and examination scheme, skill set to be developed etc.
2. Students shall organize the work in the group and make record of all observations.
3. Students shall develop maintenance skill as expected by industries.
4. Student shall attempt to develop related hand-on skills and build confidence.
5. Students shall make a small project/application in Python.
6. Student shall develop the habits of evolving more ideas, innovations, skills etc. apart from those included in scope of manual.
7. Student shall refer technical magazines and data books.
8. Student should develop a habit of submitting the experimentation work as per the schedule and s/he should be well prepared for the same.

**Common Safety Instructions**

Students are expected to

1. Switch on the PC carefully (not to use wet hands)
2. Shutdown the PC properly at the end of your Lab
3. Carefully Handle the peripherals (Mouse, Keyboard, Network cable etc)
4. Use Laptop in lab after getting permission from Teacher

**Index**

**(Progressive Assessment Sheet)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr. No. | Objective(s) of Experiment | Page No. | Date of performance | Date of submission | Assessment  Marks | Sign. of  Teacher with date | Remarks |
| 1. | Develop a program to understand the control structures of python. |  |  |  |  |  |  |
| 2. | Develop a program to learn different types of structures (list, dictionary, tuples) in python. |  |  |  |  |  |  |
| 3. | Develop a program that reads a .csv dataset file using Pandas library and display the following content of the dataset. |  |  |  |  |  |  |
| 4. | Develop a program that shows application of slicing and dicing over the rows and columns of the dataset. |  |  |  |  |  |  |
| 5. | Develop a program that shows usage of aggregate function over the input dataset**.** |  |  |  |  |  |  |
| 6. | Develop a program that applies split and merge operations on the datasets |  |  |  |  |  |  |
| 7. | Develop a program that shows the various data cleaning tasks over the dataset. a) Identifying the null values. b) Identifying the empty values.  c) Identifying the incorrect timestamp |  |  |  |  |  |  |
| 8. | Develop a program that shows usage of following NumPy array operations: a) any() b) all() c) isnan() d) isinf() e) isfinite() f) isinf() g) zeros() h) isreal() i) iscomplex() j) isscalar() k) less() l) greater() m) less\_equal() n) greater\_equal() |  |  |  |  |  |  |
| 9. | Develop a program that shows usage of following NumPy library vector functions. a) arrange() b) reshape() c) linspace() d) randint() e) dot() |  |  |  |  |  |  |
| 10. | Write a program to display below plot using matplotlib libraryFor Values of X:[1,2,3,...,49], Values of Y (thrice ofX):[3,6,9,12,...,144,147] |  |  |  |  |  |  |
| 11. | Write a program to display below bar plot using matplotlib library For value  Languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']  popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7] |  |  |  |  |  |  |
| 12. | Write a program to display below bar plot using matplotlib library For below data display pie plot  languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'] |  |  |  |  |  |  |
| 13. | Write a program to display below bar plot using matplotlib library For 200 random points for both X and Y display scatter plot |  |  |  |  |  |  |
| 14. | Develop a program that reads .csv file from the url: (https://github.com/chris1610/pbpython/blob/master/data/sample salesv3.xlsx?raw=true) and plot the data of the dataset stored in the .csv file |  |  |  |  |  |  |
| 15. | Write a text classification pipeline using a custom preprocessor and CharNGramAnalyzer using data from Wikipedia articles as a training set.  Evaluate the performance on some held out test sets |  |  |  |  |  |  |
| 16. | Write a text classification pipeline to classify movie reviews as either positive or negative.   * Find a good set of parameters using grid search.   Evaluate the performance on a held out test set. |  |  |  |  |  |  |
| Total | | | | |  |  |  |

**Experiment No: 1**

**Develop a program to understand the control structures of python.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Basic understanding of Python programming language
* Understanding of Python control structures
* Ability to use Python's built-in functions and libraries
* Familiarity with Python's syntax
* Problem-solving skills

**Relevant CO: CO1**

**Objectives:** (a) To learn and understand the different control structures in Python, such as loops, conditional statements, and functions..

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Conditional statements: Conditional statements in Python allow you to execute certain blocks of code based on whether a certain condition is true or false. The two main types of conditional statements in Python are "if" statements and "if-else" statements.

Loops: Loops in Python allow you to repeat a block of code multiple times, either for a fixed number of times or until a certain condition is met. The two main types of loops in Python are "for" loops and "while" loops.

Functions: Functions in Python allow you to encapsulate blocks of code and reuse them throughout your program. Functions can accept parameters and return values, making them a powerful tool for organizing and structuring your code.

Scope: Scope in Python refers to the region of your program where a variable or function is visible and accessible. Understanding scope is critical for avoiding errors and ensuring that your code is organized and easy to maintain.

Error handling: Error handling in Python involves detecting and responding to errors that may occur during program execution. Proper error handling can help you avoid crashes and ensure that your program continues to run smoothly.

**Safety and necessary Precautions:**

1. Data validation.
2. Check the data types.
3. Input sanitization.
4. Error Handling and Secure coding practices.
5. Use comments.
6. Test your code.

**Procedure:**

1. Plan the program structure and flow: Develop a plan for the program structure, including the control structures that will be included, and the flow of the program logic.
2. Implement the control structures in Python: Write the code to implement the different control structures in Python, including conditional statements, loops, and functions.
3. Test and debug the program: Conduct thorough testing of the program to ensure that it is functioning correctly and identify and troubleshoot any errors or bugs.
4. Refine and optimize the program: Refine the program as needed to improve performance and optimize its functionality, based on user feedback and testing results.
5. Document the program: Provide clear documentation of the program's purpose, functionality, and limitations, as well as any potential security risks or necessary precautions.
6. Deploy and maintain the program: Deploy the program for use by users, and maintain it by addressing any issues or bugs that arise and providing updates and new features as needed.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is a conditional statement in Python?

# What is a loop in Python?

# What is the difference between a "for" loop and a "while" loop in Python?

# What is a function in Python?

# What is scope in Python?

# Suggested Reference:

# <https://docs.python.org/3/library/>

# <https://www.tutorialspoint.com/python/>

# <https://www.geeksforgeeks.org/>

# <https://realpython.com/>

# <https://www.w3schools.com/python/>

# References used by the students:

# GeeksofGeeks

# Python.com

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 2**

**Develop a program to learn different types of structures (list, dictionary, tuples) in python.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Basic programming concepts: You should have a good grasp of basic programming concepts such as variables, data types, conditional statements, loops, and functions.
* Python programming language: You should have a good understanding of Python syntax, data structures, and standard library functions.
* Sequences: Sequences are ordered collections of elements that can be accessed by their index or key. You should have a good understanding of the different types of sequences such as string, tuple, list, dictionary, and set, and their respective properties.
* String manipulation: You should know how to manipulate them using methods such as slicing, concatenation, and formatting.
* Collection manipulation: Collections such as lists, tuples, dictionaries, and sets can be manipulated using methods such as append, insert, remove, pop, and sort.
* Iteration: You should know how to use for loops and list comprehensions to iterate over sequences.
* Conditional statements: You should know how to use conditional statements to check for specific conditions in sequences.
* Functions: You should know how to define functions that operate on sequences and return values.

**Relevant CO: CO1**

**Objectives:** (a) To learn how to manipulate and access their elements, iterate over them, perform conditional operations on them, and use them in functions.

(b) To learn how to select the appropriate sequence type for a given task based on its properties and performance characteristics.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

1. In Python programming language, there are four built-in sequence types: strings, lists, tuples, and ranges. Additionally, Python includes the set and dictionary data structures, which are implemented as unordered collections of unique and key-value pairs, respectively.
2. The string data type in Python represents a sequence of characters and is immutable, meaning its contents cannot be changed once it is created. Strings can be manipulated using various methods such as slicing, concatenation, and formatting.
3. Lists and tuples are similar in many ways, but tuples are immutable, whereas lists are mutable. Lists and tuples can hold elements of any data type and can be indexed and sliced like strings. However, lists offer additional methods such as append, insert, remove, and pop that allow for manipulation of the list's contents.
4. Dictionaries are another important sequence type in Python and are implemented as unordered collections of key-value pairs. Each element in a dictionary consists of a key and a corresponding value. Dictionaries can be used to store and retrieve data quickly based on the key.
5. Sets are collections of unique elements that are unordered and mutable. Sets are often used to perform set operations such as union, intersection, and difference.

**Safety and necessary Precautions:**

1. Use of proper data validation.
2. Secure data storage.
3. Proper error handling.
4. Testing and debugging.
5. Keeping software up to date.
6. Proper code formatting and documentation.

**Procedure:**

1. Create a string variable using single or double quotes.

Use string methods like upper(), lower(), strip(), split(), join(), and replace() to manipulate the string as needed.

Use indexing and slicing to access specific characters or substrings within the string.

1. Create a tuple variable using parentheses.

Use indexing and slicing to access specific elements or subsets within the tuple.

Tuples are immutable, so you cannot add, remove or modify elements once created.

1. Create a list variable using square brackets.

Use indexing and slicing to access specific elements or subsets within the list.

Use list methods like append(), insert(), remove(), pop(), extend(), and sort() to modify the list as needed.

Lists are mutable, so you can add, remove or modify elements once created.

1. Create a dictionary variable using curly braces or the dict() constructor.

Use keys to access values within the dictionary.

Use dictionary methods like keys(), values(), and items() to access different parts of the dictionary.

Use del or pop() to remove elements from the dictionary.

Use assignment to add or modify elements in the dictionary.

1. Create a set variable using curly braces or the set() constructor.

Use set methods like add(), remove(), pop(), union(), and intersection() to modify or perform operations on the set.

Sets do not allow duplicate elements, so adding the same element multiple times will only add it once.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What method can you use to convert a string to uppercase in Python?

# What is the difference between a tuple and a list in Python?

# How do you add an element to a list in Python?

# How do you access a value in a dictionary using its key in Python?

# What is a set in Python?

# Suggested Reference:

# <https://docs.python.org/3/library/>

# <https://www.tutorialspoint.com/python/>

# <https://www.geeksforgeeks.org/>

# <https://realpython.com/>

# <https://www.w3schools.com/python/>

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 3**

**Develop a program that reads a .csv dataset file using Pandas library and display the following content of the dataset.**

**a) First five rows of the dataset**

**b) Complete data of the dataset**

**c) Summary or metadata of the dataset.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Knowledge of Python programming language and its libraries, particularly the Pandas library.
* Understanding of the structure of .csv files and how to read and manipulate them using Pandas.
* Familiarity with the different methods and functions available in Pandas, such as "head()", "print()", "display()", "info()", and "describe()".
* Ability to write and debug code, and troubleshoot errors that may arise when working with datasets.
* Experience in working with datasets, including data cleaning, data wrangling, and data analysis.
* Ability to understand the content and structure of datasets, and use them to derive insights and information.

**Practical skills:**

* Writing code to load a .csv dataset file into a Pandas DataFrame using the "read\_csv()" function.
* Using the "head()" method to display the first five rows of the dataset.
* Using the "print()" function or "display()" method to display the complete data of the dataset.
* Using the "info()" method or "describe()" method to display the summary or metadata of the dataset.
* Handling errors and exceptions that may arise when working with datasets.
* Writing clean and efficient code that is easy to read and maintain.
* Testing the program with different datasets to ensure its accuracy and reliability.

**Relevant CO: CO1, CO2**

**Objectives:** (a) To read and load the .csv dataset file into a Pandas DataFrame.

(b) To display the first five rows of the dataset using the "head()" method.

(c) To display the complete data of the dataset using the "print()" function or "display()" method.

(d) To display the summary or metadata of the dataset using the "info()" method or "describe()" method.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Pandas is a popular data manipulation library for Python, widely used in data science and machine learning. It provides a powerful and flexible toolset for working with structured data, including loading, manipulating, and analyzing datasets in various formats, including .csv files

**Safety and necessary Precautions:**

1. Data security, quality and privacy.
2. Memory and performance optimization.
3. Error handling and exception handling.
4. Use comments.
5. Test your code.

**Procedure:**

1. Import the Pandas library: To use the Pandas library in Python, it is essential to import it into your program. You can do this by using the "import pandas as pd" statement.
2. Load the dataset: The next step is to load the dataset into a Pandas DataFrame using the "read\_csv()" function. This function takes the path to the .csv file as an argument and returns a DataFrame object that contains the data from the file.
3. Display the first five rows: To display the first five rows of the dataset, you can use the "head()" method. This method returns the first five rows of the DataFrame by default, but you can specify the number of rows you want to display as an argument.
4. Display the complete data: To display the complete data of the dataset, you can use the "print()" function or "display()" method. This will output the entire DataFrame to the console or Jupyter Notebook.
5. Display summary or metadata: To display the summary or metadata of the dataset, you can use the "info()" method or "describe()" method. The "info()" method provides information about the DataFrame, including the number of rows and columns, data types, and memory usage. The "describe()" method provides statistical summary of the dataset, including count, mean, standard deviation, minimum, maximum, and quartiles for each column.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What library should be used to read a .csv dataset file in Python?

# Which method is used to read a .csv file using Pandas library?

# How can you display the first five rows of the dataset using Pandas?

# How can you display the complete data of the dataset using Pandas?

# How can you display the summary or metadata of the dataset using Pandas?

# Suggested Reference:

# Official Pandas documentation: https://pandas.pydata.org/docs/

# "Python for Data Analysis" by Wes McKinney: https://www.oreilly.com/library/view/python-for-data/9781491957653/

# "Python Data Science Handbook" by Jake VanderPlas: https://jakevdp.github.io/PythonDataScienceHandbook/

# Pandas tutorial by DataCamp: https://www.datacamp.com/community/tutorials/pandas-tutorial-dataframe-python

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 4**

**Develop a program that shows application of slicing and dicing over the rows and columns of the dataset.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Knowledge of Python programming language.
* Familiarity with Pandas library.
* Ability to read and load dataset files.
* Familiarity with slicing and dicing operations.
* Understanding of data indexing.
* Familiarity with data cleaning and preprocessing.
* Knowledge of data visualization.
* Problem-solving skills.
* Strong analytical and statistical skills.

**Relevant CO: CO1, CO2**

**Objectives:** (a) To gain insights into the dataset and extract meaningful information from it.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Slicing and dicing are powerful operations that allow data analysts to manipulate data by selecting specific subsets of data from a larger dataset. These operations are widely used in data analysis and are a crucial aspect of data manipulation.

In the context of Python, slicing refers to extracting specific portions of data from a larger data structure, such as a list, tuple, or DataFrame. Slicing is performed by specifying the start and end indices of the portion of data to be extracted. For example, in a list of numbers, slicing can be used to extract the first three numbers or the last five numbers. In a DataFrame, slicing can be used to extract specific rows or columns based on specific conditions or criteria.

Dicing, on the other hand, refers to grouping and aggregating data based on specific criteria. This involves dividing the data into smaller subsets based on specific categories or conditions and performing aggregation functions on each subset. For example, in a dataset containing sales data, dicing can be used to group the data by product type, region, or time period and calculate the total sales for each group.

In Python, the Pandas library provides powerful tools for slicing and dicing data in a DataFrame. The .loc and .iloc methods are used for slicing rows and columns based on specific conditions or criteria. The .groupby method is used for grouping data based on specific categories, and aggregation functions such as .sum(), .mean(), and .count() can be used to perform calculations on each group. The .pivot\_table method is used for creating pivot tables, which provide a summarized view of the data by grouping and aggregating data based on specific categories.

**Safety and necessary Precautions:**

1. Backup the original data.
2. Validate the data.
3. Check the output.
4. Secure the data.
5. Use appropriate tools: the Pandas library provides powerful tools for data manipulation and analysis.

**Procedure:**

1. Load the dataset: Load the dataset into Python using the Pandas library's read\_csv function.
2. Explore the dataset: Use the head, tail, and info functions to explore the dataset and get a sense of its structure and contents.
3. Slice and dice the data: Use the Pandas DataFrame's indexing and slicing operations to select specific rows and columns of the dataset. Examples of slicing operations include loc, iloc, and [ ].
4. Apply filtering: Use Boolean indexing to filter rows of the dataset based on specific criteria.
5. Aggregate the data: Use the groupby function to group the data by specific columns and apply aggregation functions such as sum, mean, and count.
6. Visualize the data: Use visualization libraries such as Matplotlib or Seaborn to create visualizations of the sliced and diced data.
7. Refine and iterate: Refine the analysis and iterate as needed based on the insights gained from the analysis.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is the purpose of slicing and dicing in data analysis?

# Which function of the Pandas library is used to load a .csv dataset file into Python?

# What is the difference between loc and iloc in Pandas DataFrame indexing?

# How can Boolean indexing be used to filter rows of a dataset based on specific criteria?

# What is the purpose of aggregation functions in data analysis?

# Which visualization libraries can be used to create visualizations of the sliced and diced data?

# What is the importance of documenting the slicing and dicing process during data analysis?

# What is the advantage of iterating and refining the analysis during the slicing and dicing process?

# Can slicing and dicing be applied only to numerical data or can it also be applied to categorical data?

# How can the insights gained from slicing and dicing be used to make data-driven decisions?

# Suggested Reference:

# "Python for Data Analysis" by Wes McKinney

# "Python Data Science Handbook" by Jake VanderPlas

# "Pandas User Guide" on the Pandas documentation website

# "Data Wrangling with Pandas" course on DataCamp

# "Data Manipulation with Pandas" course on Coursera

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 5**

**Develop a program that shows usage of aggregate function over the input dataset. a) describe b) max c) min d) mean e) median f) count g) std h) Corr**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Knowledge of the input dataset format (e.g. CSV, Excel, JSON) and how to load it into a data structure in Python using libraries like Pandas.
* Understanding of the different aggregate functions available in Pandas, such as describe, max, min, mean, median, count, std, and corr.
* Familiarity with the syntax of Pandas functions for applying aggregate functions, such as groupby, apply, and agg.
* Ability to interpret and analyze the results of the aggregate functions to gain insights about the dataset.

**Practical skills:**

* Loading the input dataset into a Pandas DataFrame object.
* Applying the desired aggregate functions to the DataFrame using the appropriate syntax.
* Displaying the results of the aggregate functions in a user-friendly format, such as a table or chart.
* Handling any errors or exceptions that may arise during the data manipulation process.

**Relevant CO: CO1, CO2**

**Objectives:** (a) To understand the concept of aggregate functions and their usage in data analysis.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

In data analysis, aggregate functions are used to calculate summary statistics over a dataset. These functions are applied to columns or rows of a dataset to calculate values like the maximum, minimum, mean, median, count, standard deviation, and correlation.

Here is a brief overview of the aggregate functions:

a) describe: This function generates descriptive statistics that summarize the central tendency, dispersion, and shape of a dataset's distribution.

b) max: This function is used to find the maximum value of a column or row.

c) min: This function is used to find the minimum value of a column or row.

d) mean: This function is used to find the average value of a column or row.

e) median: This function is used to find the median value of a column or row.

f) count: This function is used to count the number of non-null values in a column or row.

g) std: This function is used to calculate the standard deviation of a column or row.

h) Corr: This function is used to calculate the correlation between columns or rows of a dataset.

In Python, these aggregate functions can be applied using the Pandas library. The groupby() function is used to group data based on a specified column, and the aggregate functions can then be applied to the grouped data.

**Safety and necessary Precautions:**

1. Make sure that the input dataset is clean and well-formatted.
2. Check the data types of the columns in the dataset.
3. Be careful when working with large datasets, as some aggregate functions may require a lot of computational power and memory.
4. Double-check the output of the aggregate functions to ensure that they make sense and match the expected results.

**Procedure:**

1. Import necessary libraries: You will need to import Pandas library to load the dataset and perform various operations on it.
2. Load the dataset: Load the dataset in a Pandas dataframe using the read\_csv() function. Make sure the dataset is in a CSV format and is saved in your working directory.
3. Check the dataset: Print the first few rows of the dataset using the head() function to check if the dataset is loaded correctly.
4. Describe the dataset: Use the describe() function to get the summary statistics of the dataset, such as count, mean, standard deviation, minimum, and maximum values.
5. Apply aggregate functions: Apply the aggregate functions such as max(), min(), mean(), median(), count(), std(), and corr() on the dataset.
6. Display the results: Display the results of the aggregate functions to the user.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is the purpose of using aggregate functions in a dataset?

# Which aggregate function calculates the average of a numerical column?

# Which of the following aggregate functions calculates the correlation between two numerical columns?

# Which of the following aggregate functions returns the number of non-missing values in a column?

# What is the purpose of using the describe() function in Pandas?

# Suggested Reference:

# https://pandas.pydata.org/docs/

# https://numpy.org/doc/stable/

# References used by the students:

# GeeksofGeek

# Python.com

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 6**

**Develop a program that applies split and merge operations on the datasets.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Understanding of Data Structures
* Knowledge of Programming Languages
* Familiarity with Data Manipulation Libraries
* Understanding of Splitting and Merging Operations
* Proficiency in Using IDEs and Text Editors
* Problem Solving and Troubleshooting Skills

**Relevant CO: CO1, CO2**

**Objectives:** (a) To split large datasets into smaller ones for ease of handling and processing.

(b) To consolidate information and make it easier to analyze.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Python provides several built-in functions and libraries for performing split and merge operations on datasets. Here are some examples:

Splitting a Dataset:

Using the pandas split() method: The split() method is a built-in function in Python that can be used to split a string into a list of substrings based on a specified delimiter. This can be useful for splitting a dataset into smaller chunks.

Using the numpy.array\_split() function: The numpy.array\_split() function can be used to split a numpy array into smaller arrays of equal or nearly equal size.

Merging Datasets:

Using the pandas.concat() function: The pandas.concat() function can be used to concatenate pandas dataframes along a specified axis.

Using the numpy concatenate() function: The concatenate() function can be used to merge two or more arrays into a single array.

**Safety and necessary Precautions:**

1. Check for data consistency.
2. Avoid overwriting original data.
3. Check for duplicates.
4. Handle missing data.
5. Test the code thoroughly.

**Procedure:**

1. Define the input datasets: Determine the input datasets and their format. It could be CSV files, Excel files, or other file types. Also, define the delimiter or separator character for splitting the data.
2. Load the datasets: Load the datasets into the program using the appropriate libraries and functions. Check that the data is loaded correctly and perform any necessary data cleaning or formatting.
3. Split the datasets: Use the appropriate function or library to split the datasets into smaller chunks. Specify the size or number of chunks to create and ensure that the resulting datasets are consistent and valid.
4. Merge the datasets: Use the appropriate function or library to merge the datasets into a single dataset. Specify the method of merging and ensure that the resulting dataset is consistent and valid.
5. Handle missing or duplicate data: Check for any missing or duplicate data in the merged dataset and handle them appropriately. You can choose to remove the records with missing data or impute the missing values.
6. Perform calculations or analysis: Once the datasets are merged, you can perform any necessary calculations or analysis on the resulting dataset. This could include aggregating data, calculating averages, or performing statistical analysis.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What are the key steps involved in developing a program that applies split and merge operations on datasets?

# What library or function can be used to split the input datasets into smaller chunks?

# What should you do if the merged dataset contains missing or duplicate data?

# What should you do after developing the program?

# Suggested Reference:

# [https://docs.python.org/3/library/](https://docs.python.org/3/library/operator.html)

# "Python Data Science Handbook" by Jake VanderPlas.

# "Python for Data Analysis" by Wes McKinney.

# Pandas documentation.

# NumPy documentation.

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 7**

**Develop a program that shows the various data cleaning tasks over the dataset. a) Identifying the null values. b) Identifying the empty values c) Identifying the incorrect timestamp**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Basic understanding of Python programming language
* Familiarity with data cleaning techniques, including identifying null and empty values, handling incorrect timestamps, and removing outliers.
* Knowledge of statistical methods and data visualization techniques to identify anomalies and outliers in the data.
* Familiarity with data cleaning libraries and tools, such as Pandas and NumPy in Python
* Problem-solving skills

**Relevant CO: CO1, CO2, CO3**

**Objectives:** (a) To identify and handle missing or incomplete data in the dataset.

(b) To identify and handle invalid or incorrect data in the dataset.

(c) To remove duplicate data in the dataset.

(d) To standardize data formats and values to ensure consistency across the dataset.

(e) To handle outliers and extreme values that may skew data analysis results.

(f) To ensure data accuracy and completeness for reliable data analysis.

(g) To improve data quality by reducing errors and inconsistencies in the dataset.

(h) To prepare the dataset for further analysis and modeling..

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Data cleaning is an essential step in the data preparation process that involves identifying and handling missing, incorrect, or inconsistent data in the dataset. In Python, data cleaning is typically performed using libraries such as NumPy and Pandas, which provide functions for data manipulation and analysis.

The theory behind data cleaning in Python involves several key steps:

Importing data: The first step in data cleaning is to import the data into Python using the appropriate library and data format. Common data formats include CSV, Excel, and JSON.

Identifying missing data: Once the data is imported, the next step is to identify missing data in the dataset. This can be done using the isnull() function in Pandas, which returns a Boolean value indicating whether a value is missing or not.

Handling missing data: Once missing data is identified, the next step is to handle it appropriately. This can be done by either removing the rows or columns with missing values or imputing the missing values with a suitable value such as the mean or median of the column.

Identifying incorrect data: After handling missing data, the next step is to identify incorrect data in the dataset, such as values that are outside the expected range or format. This can be done using statistical techniques such as data visualization and analysis.

Handling incorrect data: Once incorrect data is identified, the next step is to handle it appropriately. This can be done by removing the outliers or replacing the incorrect values with a suitable value such as the median or mode of the column.

Standardizing data formats and values: To ensure consistency across the dataset, it is often necessary to standardize data formats and values. This can be done by converting data types, renaming columns, or applying formatting rules.

Removing duplicates: Duplicate data can skew analysis results and should be removed from the dataset. This can be done using the drop\_duplicates() function in Pandas.

Quality control: The final step in data cleaning is to perform quality control checks to ensure that the data is accurate, complete, and consistent. This involves comparing the cleaned dataset to the original dataset and verifying that the data has been cleaned appropriately.

**Safety and necessary Precautions:**

1. Backup data.
2. Use secure and updated software.
3. Access control.
4. Data privacy.
5. Data encryption
6. Error handling.
7. Test and validate.

**Procedure:**

1. Import the required libraries: Import the necessary libraries such as pandas, numpy, and matplotlib to read, manipulate and visualize the dataset.
2. Load the dataset: Load the dataset into the program using a pandas dataframe.
3. Identify null values: Use the isnull() function to identify null values in the dataset. If any null values are found, decide on a strategy to handle them. This could involve replacing null values with a mean or median value, dropping the null values or imputing them with a different value.
4. Identify empty values: Use the empty() function to identify empty values in the dataset. Empty values are those values that contain nothing (not even null). If any empty values are found, decide on a strategy to handle them. This could involve replacing empty values with a mean or median value, dropping the empty values or imputing them with a different value.
5. Identify incorrect timestamp: Use the to\_datetime() function to convert the timestamp column to a datetime object. This will identify any incorrect timestamp values. If any incorrect timestamp values are found, decide on a strategy to handle them. This could involve dropping the rows with incorrect timestamp values or imputing them with a different value.
6. Remove duplicates: Use the drop\_duplicates() function to remove any duplicate rows in the dataset.
7. Data normalization: Use the normalization technique to transform the data into a standard format to make it more consistent and easier to analyze.
8. Data standardization: Use the standardization technique to transform the data into a standard scale to make it more consistent and easier to analyze.
9. Save the cleaned dataset: Save the cleaned dataset to a new file for future use.
10. Visualize the cleaned dataset: Use matplotlib or other visualization libraries to create visualizations of the cleaned dataset to better understand the data and identify any further cleaning that may be required.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is the first step in developing a program for data cleaning in Python?

# How can null values be identified in a dataset?

# How can empty values be handled in a dataset?

# How can incorrect timestamp values be identified in a dataset?

# What is the purpose of data normalization in data cleaning?

# Suggested Reference:

# Data Cleaning with Python" course on DataCamp.

# "Data Cleaning in Python: A Complete Guide" on Towards Data Science.

# "Data Cleaning with Python and Pandas: Detecting Missing Values" on Real Python.

# "Cleaning Data with Python" on Kaggle.

# "Data Cleaning Techniques in Python" on Analytics Vidhya

# References used by the students: (Sufficient space to be provided)

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 8**

**Develop a program that shows usage of following NumPy array operations: a) any() b) all() c) isnan() d) isinf() e) isfinite() f) isinf() g) zeros() h) isreal() i) iscomplex() j) isscalar() k) less() l) greater() m) less\_equal() n) greater\_equal()**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Understanding of Data Structures
* Knowledge of Programming Languages
* Familiarity with Data Manipulation Libraries
* Proficiency in Using IDEs and Text Editors
* Problem Solving and Troubleshooting Skills

**Relevant CO: CO2**

**Objectives:** (a) To perform complex mathematical and logical operations on large arrays and matrices efficiently.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

NumPy is a popular Python library for scientific computing that provides efficient and powerful array operations. It enables users to work with multidimensional arrays and perform a variety of mathematical and logical operations on them.

Here are the explanations of some of the NumPy array operations mentioned in the question:

a) any(): It returns True if any of the elements of an array evaluate to True, and False otherwise

b) all(): It returns True if all the elements of an array evaluate to True, and False otherwise.

c) isnan(): It returns an array of the same shape as the input array, with True where the corresponding element of the input array is NaN (Not a Number), and False elsewhere.

d) isinf(): It returns an array of the same shape as the input array, with True where the corresponding element of the input array is +/-inf (positive or negative infinity), and False elsewhere.

e) isfinite(): It returns an array of the same shape as the input array, with True where the corresponding element of the input array is finite (i.e., not NaN, +/-inf), and False elsewhere.

f) isinf(): It returns an array of the same shape as the input array, with True where the corresponding element of the input array is +/-inf (positive or negative infinity), and False elsewhere.

g) zeros(): It returns a new array of the specified shape and data type, filled with zeros.

h) isreal(): It returns an array of the same shape as the input array, with True where the corresponding element of the input array is real, and False where it is complex.

i) iscomplex(): It returns an array of the same shape as the input array, with True where the corresponding element of the input array is complex, and False where it is real.

j) isscalar(): It returns True if the input is a scalar (i.e., a single value, not an array), and False otherwise.

k) less(): It returns an array of the same shape as the input arrays, with True where the corresponding element of the first input array is less than the corresponding element of the second input array, and False otherwise.

l) greater(): It returns an array of the same shape as the input arrays, with True where the corresponding element of the first input array is greater than the corresponding element of the second input array, and False otherwise.

m) less\_equal(): It returns an array of the same shape as the input arrays, with True where the corresponding element of the first input array is less than or equal to the corresponding element of the second input array, and False otherwise.

n) greater\_equal(): It returns an array of the same shape as the input arrays, with True where the corresponding element of the first input array is greater than or equal to the corresponding element of the second input array, and False otherwise.

**Safety and necessary Precautions:**

1. Make sure to import NumPy correctly.
2. Use appropriate data types.
3. Watch out for NaN and Inf.
4. Be careful with memory usage.
5. Test the program thoroughly.

**Procedure:**

1. Import the NumPy library: To use NumPy array operations, you need to import the NumPy library into your Python environment. You can do this using the import statement.
2. Create a NumPy array: You need to create a NumPy array to perform the various operations. You can create an array using the np.array() function.
3. Use the array operations: Once you have created the array, you can use various NumPy array operations such as any(), all(), isnan(), isinf(), isfinite(), zeros(), isreal(), iscomplex(), isscalar(), less(), greater(), less\_equal(), and greater\_equal().
4. Print the output: After performing the operations, you should print the output to see the results.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What does the NumPy function 'any()' return?

# What is the purpose of the NumPy function 'isnan()'?

# What does the NumPy function 'zeros()' do?

# What does the NumPy function 'isreal()' do?

# What is the purpose of the NumPy function 'less()'?

# Suggested Reference:

# NumPy User Guide: https://numpy.org/doc/stable/user/index.html

# NumPy Tutorial: https://www.tutorialspoint.com/numpy/index.htm

# NumPy Cheat Sheet: https://s3.amazonaws.com/assets.datacamp.com/blog\_assets/Numpy\_Python\_Cheat\_Sheet.pdf

# NumPy Array Operations: https://www.geeksforgeeks.org/numpy-array-manipulation-python/

# NumPy Array Operations and Functions: https://www.w3schools.com/python/numpy\_array\_operators.asp

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 9**

**Develop a program that shows usage of following NumPy library vector functions. a) arrange() b) reshape() c) linspace() d) randint() e) dot()**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Understanding of Data Structures
* Knowledge of Programming Languages
* Familiarity with Data Manipulation Numpy Libraries
* Proficiency in Using IDEs and Text Editors
* Problem Solving and Troubleshooting Skills

**Relevant CO: CO2**

**Objectives:** (a) To provide efficient and powerful tools for working with large arrays and matrices in Python, along with a wide range of mathematical and scientific functions for manipulating and analyzing these arrays.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Here is a brief theory for each of the NumPy vector functions:

a) arrange(): This function is used to create a one-dimensional array with evenly spaced values within a specified range. The function takes in three arguments: start (optional), stop, and step (optional). The start argument is the starting value of the sequence (inclusive), the stop argument is the ending value of the sequence (exclusive), and the step argument is the step size between values. For example, np.arange(0, 10, 2) creates an array with values [0, 2, 4, 6, 8].

b) reshape(): This function is used to reshape an array into a new shape without changing its data. The function takes in one argument: the new shape of the array, specified as a tuple of integers. For example, np.reshape(my\_array, (3, 4)) reshapes the array my\_array into a 3x4 matrix.

c) linspace(): This function is used to create a one-dimensional array with evenly spaced values between a specified range. The function takes in three arguments: start, stop, and num (optional). The start argument is the starting value of the sequence, the stop argument is the ending value of the sequence, and the num argument is the number of values to generate. For example, np.linspace(0, 1, 5) creates an array with values [0., 0.25, 0.5, 0.75, 1.].

d) randint(): This function is used to generate an array of random integers within a specified range. The function takes in three arguments: low (optional), high, and size (optional). The low argument is the lower bound of the range (inclusive), the high argument is the upper bound of the range (exclusive), and the size argument is the shape of the output array. For example, np.random.randint(0, 10, size=(2, 3)) generates a 2x3 array of random integers between 0 and 10.

e) dot(): This function is used to perform matrix multiplication between two arrays. The function takes in two arguments: the two arrays to be multiplied. The arrays must have compatible shapes for matrix multiplication. For example, if A is a 2x3 array and B is a 3x2 array, np.dot(A, B) performs matrix multiplication between A and B and returns a 2x2 array.

Overall, these NumPy vector functions are commonly used for manipulating and analyzing arrays in scientific computing and data analysis. By using these functions in a program, you can efficiently perform operations on large arrays and matrices in Python.

**Safety and necessary Precautions:**

1. Install NumPy from a trusted source.
2. Keep NumPy updated.
3. Understand data types.
4. Avoid modifying arrays in place.
5. Use vectorized operations.
6. Handle exceptions and errors.

**Procedure:**

1. Import the NumPy library: Begin your program by importing the NumPy library using the import statement.
2. Create an array: Create an array using one of the NumPy functions such as arrange() or linspace(). You can also create an array from an existing data source such as a CSV file.
3. Reshape the array: Use the reshape() function to reshape the array to the desired shape. For example, you can reshape a one-dimensional array into a two-dimensional array.
4. Generate random numbers: Use the randint() function to generate an array of random integers within a specified range.
5. Perform matrix multiplication: Use the dot() function to perform matrix multiplication between two arrays.
6. Print the results: Print the resulting arrays to the console using the print() function.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is the purpose of the NumPy library?

# Which NumPy function can be used to create an array with evenly spaced values?

# Which NumPy function can be used to generate an array of random integers within a specified range?

# How can you perform matrix multiplication between two arrays in NumPy?

# What is the purpose of the reshape() function in NumPy?

# Suggested Reference:

# https://numpy.org/doc/stable/

# https://numpy.org/doc/stable/user/index.html

# https://s3.amazonaws.com/assets.datacamp.com/blog\_assets/Numpy\_Python\_Cheat\_Sheet.pdf

# <https://numpy.org/devdocs/user/quickstart.html>

# https://www.datacamp.com/community/tutorials/python-numpy-tutorial

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 10**

**Write a program to display below plot using matplotlib library. For Values of X:[1,2,3,...,49], Values of Y (thrice of X):[3,6,9,12,...,144,147]**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Understanding the basics of data visualization
* Familiarity with Python programming language
* Knowledge of the different types of plots and when to use them
* Knowledge of the syntax and parameters for different matplotlib functions
* Understanding of data structures like arrays and data frames

**Practical skills:**

* Ability to create different types of plots such as line plots, scatter plots, bar plots, etc.
* Ability to customize the appearance of plots including labels, colors, legends, and titles
* Ability to add text, annotations, and shapes to the plots
* Ability to work with multiple plots and subplots
* Ability to export plots in different file formats like png, pdf, svg, etc.
* Ability to integrate matplotlib with other Python libraries like NumPy and Pandas.

**Relevant CO: CO4**

**Objectives:** (a) To create informative and visually appealing data visualizations that enable users to explore, understand, and communicate complex data.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Matplotlib is a Python library that provides a variety of tools for creating high-quality data visualizations. It is one of the most popular data visualization libraries due to its ease of use and versatility. The library is built on NumPy and provides a range of options for creating different types of plots and graphs, including line plots, scatter plots, bar charts, histograms, and many more.

The main components of the Matplotlib library are:

pyplot module: This is the main module of Matplotlib, which provides a simple interface for creating plots and charts. It is a collection of functions that allow users to create plots with minimal coding.

Figure and Axes objects: The Figure object is the top-level container for all the plot elements. It represents the entire plot and contains one or more Axes objects. The Axes object is the individual plot area where data is plotted.

Plotting functions: Matplotlib provides a range of plotting functions that can be used to create different types of plots and charts. These functions include plot(), scatter(), bar(), hist(), and many more.

Customization options: Matplotlib allows users to customize the appearance of plots in various ways, including changing the plot color, adding labels, titles, and legends, adjusting the axis limits, and more.

To use Matplotlib, you first need to import the library and its pyplot module. Then, you can create a figure object and one or more axes objects using the subplots() function. After that, you can use the various plotting functions to create different types of plots and customize them as needed.

Overall, Matplotlib provides a powerful and flexible tool for creating data visualizations in Python. With its wide range of options and customization features, it can be used for a variety of data analysis and communication tasks.

**Safety and necessary Precautions:**

1. Keep Matplotlib libraries up-to-date.
2. Use Comments
3. Test your code.

**Procedure:**

1. Import the required libraries - Matplotlib and NumPy.
2. Create two NumPy arrays for X and Y values using np.arange() and multiplication.
3. Create a figure and an axis object using plt.subplots().
4. Use a x.plot() function to plot X and Y values as a line plot.
5. Customize the plot with axis labels and a title.
6. Display the plot using plt.show() function.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is Matplotlib?

# What are the two basic types of plots in Matplotlib?

# How can you change the color of a plot in Matplotlib?

# How can you add a legend to a plot in Matplotlib?

# What is the function used to save a plot to a file in Matplotlib?

# Suggested Reference:

# https://matplotlib.org/stable/index.html

# <https://realpython.com/python-matplotlib-guide/>

# Matplotlib Tutorial by Corey Schafer: https://www.youtube.com/playlist?list=PL-osiE80TeTvipOqomVEeZ1HRrcEvtZB\_

# Python Data Science Handbook by Jake VanderPlas: https://jakevdp.github.io/PythonDataScienceHandbook/

# Mastering Matplotlib by Duncan M. McGreggor and Paul Ivanov: https://www.packtpub.com/product/mastering-matplotlib-second-edition/9781800565547

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 11**

**Write a program to display below bar plot using matplotlib library. For value**

**Languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']**

**Popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Understanding the basics of data visualization
* Familiarity with Python programming language
* Knowledge of the different types of plots and when to use them
* Knowledge of the syntax and parameters for different matplotlib functions
* Understanding of data structures like arrays and data frames

**Practical skills:**

* Ability to create different types of plots such as line plots, scatter plots, bar plots, etc.
* Ability to customize the appearance of plots including labels, colors, legends, and titles
* Ability to add text, annotations, and shapes to the plots
* Ability to work with multiple plots and subplots
* Ability to export plots in different file formats like png, pdf, svg, etc.
* Ability to integrate matplotlib with other Python libraries like NumPy and Pandas.

**Relevant CO: CO4**

**Objectives:** (a) To learn how to interpret and analyze data visualizations, and to use them to draw insights and make informed decisions.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

A bar plot is a type of chart that displays data as rectangular bars. The length or height of each bar is proportional to the value of the data it represents. Bar plots are useful for comparing the values of different categories or groups.

Matplotlib is a popular data visualization library in Python that provides a wide range of functions for creating different types of plots, including bar plots.

Use the bar() function to create the bar plot by passing the languages and popularity lists as arguments. The bar() function automatically generates the rectangular bars for each category and sets their lengths proportional to the values in the popularity list.

**Safety and necessary Precautions:**

1. Keep Matplotlib libraries up-to-date.
2. Use Comments
3. Test your code.

**Procedure:**

1. Define the data for the plot as lists or arrays.
2. Use the bar() function to create the plot, passing the data as arguments.
3. Customize the plot by changing the colors, labels, and other attributes.
4. Add a title and labels to the plot to provide context and improve its readability.
5. Display the plot using the show() function.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is a bar plot?

# Which library is used to create a bar plot in Python?

# What are the steps involved in creating a bar plot using Matplotlib?

# What is the correct syntax to create a bar plot using Matplotlib?

# What are the parameters required by the bar() function to create a bar plot?

# Suggested Reference:

# https://matplotlib.org/stable/index.html

# <https://realpython.com/python-matplotlib-guide/>

# Matplotlib Tutorial by Corey Schafer: https://www.youtube.com/playlist?list=PL-osiE80TeTvipOqomVEeZ1HRrcEvtZB\_

# Python Data Science Handbook by Jake VanderPlas: https://jakevdp.github.io/PythonDataScienceHandbook/

# Mastering Matplotlib by Duncan M. McGreggor and Paul Ivanov: https://www.packtpub.com/product/mastering-matplotlib-second-edition/9781800565547

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 12**

**Write a program to display below bar plot using matplotlib library For below data display pie plot**

**Languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']**

**Popuratity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]**

**Colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b"]**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Understanding the basics of data visualization
* Familiarity with Python programming language
* Knowledge of the different types of plots and when to use them
* Knowledge of the syntax and parameters for different matplotlib functions
* Understanding of data structures like arrays and data frames

**Practical skills:**

* Ability to create different types of plots such as line plots, scatter plots, bar plots, etc.
* Ability to customize the appearance of plots including labels, colors, legends, and titles
* Ability to add text, annotations, and shapes to the plots
* Ability to work with multiple plots and subplots
* Ability to export plots in different file formats like png, pdf, svg, etc.
* Ability to integrate matplotlib with other Python libraries like NumPy and Pandas.

**Relevant CO: CO1, CO4**

**Objectives:** (a) To learn how to interpret and analyze data visualizations, and to use them to draw insights and make informed decisions.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

A bar plot is a type of chart that displays data as rectangular bars. The length or height of each bar is proportional to the value of the data it represents. Bar plots are useful for comparing the values of different categories or groups.

Matplotlib is a popular data visualization library in Python that provides a wide range of functions for creating different types of plots, including bar plots.

Use the bar() function to create the bar plot by passing the languages and popularity lists as arguments. The bar() function automatically generates the rectangular bars for each category and sets their lengths proportional to the values in the popularity list.

**Safety and necessary Precautions:**

1. Keep Matplotlib libraries up-to-date.
2. Use Comments
3. Test your code.

**Procedure:**

1. Import the necessary libraries (matplotlib.pyplot)
2. Define the data to be used (Languages, Popularity, Colors)
3. Create a figure object and set the figure size
4. Define the title of the plot and add the data to be displayed (Popularity) and their corresponding labels (Languages)
5. Set the colors of the pie chart using the Colors list
6. Add a legend to the chart with the labels and colors used
7. Display the plot.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What libraries do you need to import to create the pie chart using matplotlib?

# What is the purpose of defining the Colors list in the program?

# What is the purpose of setting the figure size in the program?

# How do you add a legend to the pie chart in matplotlib?

# What is the purpose of the Popularity list in the program?

# Suggested Reference:

# https://matplotlib.org/stable/index.html

# <https://realpython.com/python-matplotlib-guide/>

# Matplotlib Tutorial by Corey Schafer: https://www.youtube.com/playlist?list=PL-osiE80TeTvipOqomVEeZ1HRrcEvtZB\_

# Python Data Science Handbook by Jake VanderPlas: <https://jakevdp.github.io/PythonDataScienceHandbook/>

# Mastering Matplotlib by Duncan M. McGreggor and Paul Ivanov: https://www.packtpub.com/product/mastering-matplotlib-second-edition/9781800565547

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 13**

**Write a program to display below bar plot using matplotlib library For 200 random points for both X and Y display scatter plot.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Understanding the basics of data visualization
* Familiarity with Python programming language
* Knowledge of the different types of plots and when to use them
* Knowledge of the syntax and parameters for different matplotlib functions
* Understanding of data structures like arrays and data frames

**Practical skills:**

* Ability to create different types of plots such as line plots, scatter plots, bar plots, etc.
* Ability to customize the appearance of plots including labels, colors, legends, and titles
* Ability to add text, annotations, and shapes to the plots
* Ability to work with multiple plots and subplots
* Ability to export plots in different file formats like png, pdf, svg, etc.
* Ability to integrate matplotlib with other Python libraries like NumPy and Pandas.

**Relevant CO: CO4**

**Objectives:** (a) To learn how to interpret and analyze data visualizations, and to use them to draw insights and make informed decisions.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

In Matplotlib, a scatter plot is a chart type that displays data as a collection of points with the position determined by the values of two variables. Each point on the scatter plot represents an observation, and the position of the point on the X-Y axis is determined by the values of the two variables.

A scatter plot is useful for exploring the relationship between two continuous variables. It can be used to identify patterns or trends in the data and to detect the presence of outliers or unusual observations. Scatter plots can also be used to assess the correlation between the two variables.

Matplotlib provides the scatter() function for creating scatter plots. The function takes two arrays, one for the X-axis data and one for the Y-axis data, as its input arguments. Additional parameters can be used to customize the appearance of the scatter plot, such as the color, size, and transparency of the points.

**Safety and necessary Precautions:**

1. Keep Matplotlib libraries up-to-date.
2. Use Comments
3. Test your code.

**Procedure:**

1. Import necessary libraries: We will need the Matplotlib and NumPy libraries for this task.
2. Generate random data for the X and Y axes: We can use the NumPy library to generate random data for both the X and Y axes
3. Create a scatter plot: We can use the scatter method of the Matplotlib library to create a scatter plot. We need to pass the X and Y data as arguments and specify the marker style and color using the marker and c parameters, respectively
4. Add title and labels: We can add a title and labels for the X and Y axes using the title, xlabel, and ylabel methods of the Matplotlib library.
5. Set axes limits: We can set the limits for the X and Y axes using the xlim and ylim methods of the Matplotlib library.
6. Display the plot: We can display the plot using the show method of the Matplotlib library.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is a scatter plot?

# What is the function used for creating scatter plots in Matplotlib?

# What are the input arguments for the scatter() function?

# What can a scatter plot be used for?

# Can the appearance of the scatter plot be customized?

# 

# Suggested Reference:

# https://matplotlib.org/stable/index.html

# <https://realpython.com/python-matplotlib-guide/>

# Matplotlib Tutorial by Corey Schafer: https://www.youtube.com/playlist?list=PL-osiE80TeTvipOqomVEeZ1HRrcEvtZB\_

# Python Data Science Handbook by Jake VanderPlas: <https://jakevdp.github.io/PythonDataScienceHandbook/>

# Mastering Matplotlib by Duncan M. McGreggor and Paul Ivanov: https://www.packtpub.com/product/mastering-matplotlib-second-edition/9781800565547

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 14**

**Develop a program that reads .csv and plot the data of the dataset stored in the .csv file file from the url: (https://github.com/chris1610/pbpython/blob/master/data/sample salesv3.xlsx?raw=true)**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Data analysis, data visualization, file handling and programming.

**Practical skills:**

* Reading data from a CSV file using Python's pandas library.
* Pre-processing and cleaning data as required.
* Plotting data using the matplotlib library.
* Dealing with missing values or null values in the data (if any).
* Analyzing and interpreting the data visualizations to draw insights and make conclusions.

**Relevant CO: CO3, CO4**

**Objectives:** (a) To analyze and visualize the data in an efficient and effective way.

(b) To identify patterns, trends, and outliers in the data.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Reading a .csv file from a URL and plotting the data is a common data analysis and visualization task in many fields. Here are the main steps involved in this process:

Importing the necessary libraries: To read and plot the .csv file, we typically use the pandas and matplotlib libraries. We need to import them at the beginning of our program.

Loading the data from the URL: We can use the pandas library's read\_csv function to read the data from the URL. We need to provide the URL of the .csv file as an argument to this function.

Data cleaning and preparation: Once we have loaded the data, we may need to clean and prepare it for visualization. This may include dropping unnecessary columns, filling missing values, and transforming the data.

Data visualization: Once the data is cleaned and prepared, we can use matplotlib's various plotting functions to create visualizations such as line plots, scatter plots, bar plots, and more. We can customize the plot with various parameters such as colors, labels, titles, and more.

Displaying the plot: After creating the plot, we need to display it using the show function provided by the matplotlib library.

**Safety and necessary Precautions:**

1. Validate inputs.
2. Handle errors.
3. Secure the program
4. Optimize performance
5. Test and review.

**Procedure:**

1. Import the necessary libraries: You will need the pandas library to read the .csv file, and matplotlib library to create the plot.
2. Read the .csv file from the URL: Use the pandas library to read the .csv file from the URL and store it as a DataFrame object.
3. Preprocess the data: Preprocess the data as required. This may involve cleaning the data, removing duplicates, handling missing values, and converting data types.
4. Visualize the data: Use the matplotlib library to create a visualization of the data. You can create scatter plots, line graphs, histograms, and other types of visualizations based on the data.
5. Save or display the visualization: Save the visualization to a file or display it on the screen, depending on the user requirements.
6. Test and validate the program: Test the program thoroughly to ensure that it works as expected for various input datasets. Validate the results against the expected output and fix any issues or errors.
7. Document the program: Document the program by providing clear and concise comments in the code and a user manual that explains how to use the program.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What library is required to read a .csv file in Python?

# What library is required to create plots in Python?

# What is the first step in developing a program that reads a .csv file from a URL and plots the data?

# How do you read a .csv file from a URL in Python using the pandas library?

# How do you create a scatter plot of two columns from a DataFrame using the matplotlib library?

# How do you save a plot to a file using the matplotlib library?

# Suggested Reference:

# Pandas documentation on reading a CSV file from a URL:

# https://pandas.pydata.org/pandas-docs/stable/user\_guide/io.html#reading-csv-files

# Matplotlib documentation on creating plots:

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

# Real Python tutorial on reading and writing CSV files in Python:

# https://realpython.com/python-csv/

# DataCamp tutorial on data visualization with Matplotlib:

# https://www.datacamp.com/community/tutorials/matplotlib-tutorial-python

# Towards Data Science tutorial on creating visualizations with Pandas and Matplotlib:

# <https://towardsdatascience.com/data-visualization-with-pandas-and-matplotlib-8dadc69f2f79>

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 15**

**Write a text classification pipeline using a custom preprocessor and CharNGramAnalyzer using data from Wikipedia articles as a training set.**

**Evaluate the performance on some held out test sets.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Proficiency in Python programming language.
* Familiarity with the scikit-learn library for machine learning.
* Knowledge of text preprocessing techniques, such as tokenization, stop word removal, stemming, and lemmatization.
* Understanding of feature extraction techniques, such as bag-of-words and character n-grams.
* Ability to evaluate the performance of a text classification model using metrics such as accuracy, precision, recall, and F1 score.
* Knowledge of cross-validation techniques for evaluating model performance on held-out test sets.
* Familiarity with data collection and preprocessing techniques for building a training set from Wikipedia articles.

**Relevant CO: CO3, CO4, CO5**

**Objectives:** (a) To develop a machine learning model that can accurately classify text documents into predefined categories that can be used for various applications such as sentiment analysis, spam detection, and topic modeling.

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

Text classification is the task of assigning predefined categories or labels to text documents based on their content. A text classification pipeline typically consists of several stages, including data preprocessing, feature extraction, model training, and evaluation.

In the context of Wikipedia articles, the first step in building a text classification pipeline is to collect a dataset of articles with their corresponding labels. These labels can be either manually assigned or obtained from existing metadata such as categories or tags.

Once a dataset is obtained, the next step is data preprocessing. This typically involves text normalization, tokenization, stop word removal, and stemming/lemmatization. The goal of data preprocessing is to clean the text and reduce its dimensionality while retaining the relevant information for classification.

After preprocessing, the text is converted into numerical features that can be used as input to a machine learning model. A popular technique for feature extraction is the bag-of-words model, which represents each document as a vector of word frequencies. However, this approach may not capture the semantic meaning of words and their relationships in the text.

An alternative approach is to use character n-grams, such as CharNGramAnalyzer, which captures the sequence of characters in the text. This method is particularly useful for capturing the morphology and syntax of the text and can improve the performance of the classifier.

The final stage in the text classification pipeline is model training and evaluation. A common approach is to use supervised learning algorithms such as Naive Bayes, Logistic Regression, or Support Vector Machines. The performance of the model is evaluated using metrics such as accuracy, precision, recall, and F1 score on held-out test sets.

In summary, building a text classification pipeline using a custom preprocessor and CharNGramAnalyzer involves data preprocessing, feature extraction, model training, and evaluation. This approach can be particularly useful for text classification tasks where the meaning and relationships of words are important.

**Safety and necessary Precautions:**

1. Data privacy.
2. Bias and fairness.
3. Model accuracy and reliability
4. Ethical considerations
5. Test and review.

**Procedure:**

Collect and preprocess the data: Download a set of Wikipedia articles that represent the different categories you want to classify (e.g., sports, politics, entertainment, etc.). Preprocess the data by removing any unnecessary characters, converting all text to lowercase, and removing any stop words.

Split the data: Split the preprocessed data into two sets: training and test sets. The training set will be used to train the model, while the test set will be used to evaluate the model's performance.

Feature extraction: Extract the features from the preprocessed text using CharNGramAnalyzer. This will convert each text document into a vector of features that can be used as input to the classification model.

Train the model: Train a text classification model using the extracted features and the training set. You can use any machine learning algorithm, such as Naive Bayes, SVM, or Neural Networks.

Evaluate the model: Use the trained model to classify the test set and evaluate its performance using metrics such as accuracy, precision, recall, and F1-score.

Tune the model: If the model's performance is not satisfactory, you can tune the hyperparameters of the algorithm or try different algorithms to improve its performance.

Deploy the model: Once you are satisfied with the model's performance, you can deploy it in production to classify new text documents.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is the purpose of using a custom preprocessor in a text classification pipeline?

# Which analyzer is used in the given scenario?

# "Writing a text classification pipeline using a custom preprocessor and CharNGramAnalyzer using data from Wikipedia articles as a training set."

# What is the purpose of evaluating the performance on held-out test sets in text classification?

# 

# Suggested Reference:

# "Building a Text Classification Pipeline with Python" by Dipanjan Sarkar: This article provides a step-by-step guide on how to build a text classification pipeline using Python and scikit-learn library. It covers preprocessing techniques, feature extraction, model selection, and evaluation.

# "Text Classification with NLTK and Scikit-Learn" by Ahmed Besbes: This tutorial provides a detailed guide on how to perform text classification using Python and two popular libraries, NLTK and scikit-learn. It covers data preprocessing, feature extraction, and model training and evaluation.

# "Using Wikipedia Articles for Text Classification" by Nikolay Krylov: This article demonstrates how to use Wikipedia articles as a training set for text classification. It covers data collection, preprocessing, feature extraction using TF-IDF and CharNGramAnalyzer, model training, and evaluation.

# "Text Classification with Python and Scikit-Learn" by Sebastian Raschka: This book chapter provides a comprehensive guide on how to perform text classification using Python and scikit-learn. It covers data preprocessing, feature extraction, model training, and evaluation, as well as advanced topics such as model selection and parameter tuning.

# "A Complete Tutorial on Text Classification using Naive Bayes Algorithm" by Divya Gupta: This tutorial provides a detailed guide on how to perform text classification using Naive Bayes algorithm in Python. It covers data preprocessing, feature extraction, model training and evaluation, as well as parameter tuning.

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**

**Experiment No: 16**

**Write a text classification pipeline to classify movie reviews as either positive or negative.**

**Find a good set of parameters using grid search.**

**Evaluate the performance on a held out test set.**

**Date:**

**Competency and Practical Skills:**

**Competency skills:**

* Basic knowledge of computer systems, operating systems, and file systems.
* Familiarity with command-line interfaces (CLI) and graphical user interfaces (GUI).
* Understanding of programming languages, syntax, and logic.

**Practical skills:**

* Strong understanding of Natural Language Processing (NLP) concepts, such as tokenization, stemming, lemmatization, and feature extraction.
* Familiarity with popular NLP libraries such as NLTK, SpaCy, and Scikit-learn.
* Knowledge of machine learning algorithms, such as Naive Bayes, Support Vector Machines, and Neural Networks.
* Ability to preprocess text data, including removing stop words, cleaning text, and performing feature engineering.
* Experience with data exploration and visualization tools, such as Pandas and Matplotlib.
* Familiarity with Python programming language and its data science ecosystem, including NumPy, SciPy, and Pandas.
* Ability to evaluate the performance of a classification model using appropriate metrics such as accuracy, precision, recall, and F1-score.
* Knowledge of different hyperparameter tuning techniques and cross-validation methods to optimize the model's performance..

**Relevant CO: CO3, CO4, CO5**

**Objectives:** (a) To create an accurate and reliable model that can automatically classify movie reviews as positive or negative, which can be useful for analyzing large volumes of reviews quickly and efficiently, as well as for providing recommendations to users based on their preferences..

**Equipment/Instruments:** Personal Computer, Internet, Python

**Theory:**

The theory behind writing a text classification pipeline to classify movie reviews as either positive or negative involves several key steps:

Data preprocessing: This step involves cleaning and preparing the raw text data by removing stop words, converting text to lowercase, and performing stemming or lemmatization.

Feature extraction: This step involves converting the preprocessed text data into a numerical representation that can be used as input to a machine learning algorithm. Common techniques include Bag-of-Words, TF-IDF, and Word Embeddings.

Model selection and training: This step involves selecting an appropriate machine learning algorithm and training it on the preprocessed and transformed data. Popular algorithms include Naive Bayes, Support Vector Machines, and Neural Networks.

Hyperparameter tuning: This step involves selecting the optimal hyperparameters for the chosen machine learning algorithm. This can be done using techniques such as grid search or random search.

Evaluation: This step involves evaluating the performance of the trained model on a held-out test set. This can be done using metrics such as accuracy, precision, recall, and F1-score.

Deployment: This step involves deploying the trained model in a production environment, where it can be used to classify new movie reviews.

Grid search is a hyperparameter tuning technique that involves searching for the optimal set of hyperparameters for a given machine learning algorithm by exhaustively trying all possible combinations of hyperparameter values. This can be done by training and evaluating the model with different combinations of hyperparameters on a validation set, and selecting the combination that yields the best performance.

Evaluating the performance of the trained model on a held-out test set is important to ensure that the model generalizes well to new, unseen data. This helps to avoid overfitting, where the model performs well on the training data but poorly on new data.

Overall, the theory behind writing a text classification pipeline to classify movie reviews as either positive or negative involves a combination of data preprocessing, feature extraction, model selection and training, hyperparameter tuning, evaluation, and deployment.

**Safety and necessary Precautions:**

1. Data preprocessing
2. Feature extraction
3. Model selection
4. Hyper parameter tuning
5. evaluation

**Procedure:**

1. Preprocess the data: Preprocess the movie review data by cleaning the text, removing stop words, and performing stemming or lemmatization to reduce the dimensionality of the feature space.
2. Split the data: Split the preprocessed data into training, validation, and test sets. The training set will be used to train the model, the validation set will be used to tune the hyperparameters, and the test set will be used to evaluate the final performance of the model.
3. Extract features: Extract features from the preprocessed text using techniques such as Bag-of-Words, TF-IDF, or Word Embeddings. This will convert the text data into a numerical representation that can be used as input to a machine learning algorithm.
4. Select a model: Choose a suitable machine learning algorithm, such as Naive Bayes, Support Vector Machines, or Neural Networks, and train it on the preprocessed and transformed data.
5. Hyperparameter tuning: Use grid search to find the best set of hyperparameters for the chosen machine learning algorithm. This involves training and evaluating the model with different combinations of hyperparameters on the validation set, and selecting the combination that yields the best performance.
6. Evaluate the model: Evaluate the performance of the trained model on the held-out test set using metrics such as accuracy, precision, recall, and F1-score.
7. Deploy the model: Deploy the trained model in a production environment, where it can be used to classify new movie reviews.

**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

# What is the first step you should take when developing a text classification pipeline?

# What are some techniques for feature extraction in text classification?

# Which of the following algorithms is not suitable for text classification?

# What is grid search used for in text classification?

# How do you evaluate the performance of a text classification model?

# What is the purpose of a held-out test set?

# Suggested Reference:

# "Introduction to Machine Learning with Python" by Andreas C. Müller and Sarah Guido - This book provides a comprehensive introduction to machine learning and includes a section on text classification. It covers topics such as preprocessing text data, feature extraction, and model evaluation.

# "Natural Language Processing with Python" by Steven Bird, Ewan Klein, and Edward Loper - This book provides an introduction to natural language processing and includes a section on text classification. It covers topics such as feature selection, training classifiers, and evaluation metrics.

# "Scikit-learn documentation" - Scikit-learn is a popular machine learning library in Python. The documentation includes a section on text classification, which covers preprocessing text data, feature extraction, and model selection. It also provides examples of how to use grid search to find the best set of hyperparameters for a model.

# "Text Classification in Python using spaCy" by Dipanjan Sarkar - This tutorial provides an introduction to text classification using spaCy, a popular NLP library in Python. It covers topics such as preprocessing text data, feature extraction, model selection, and hyperparameter tuning.

# "Sentiment Analysis on Movie Reviews" Kaggle competition - This Kaggle competition provides a dataset of movie reviews labeled as positive or negative. It includes notebooks from participants that demonstrate how to preprocess the data, extract features, and train models.

# References used by the students:

# Python.com

# GeeksofGeeks

# W3school

# Rubric wise marks obtained:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rubrics | 1 | 2 | 3 | 4 | 5 | Total |
| Marks |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Knowledge of subject (2) | | Programming Skill | | Team work (2) | | Communication Skill (2) | | Ethics(2) | |
| Good (2) | Average (1) | Good (2) | Average (1) | Good (2) | Satisfactory (1) | Good (2) | Satisfactory (1) | Good (2) | Average (1) |

**Sign of Faculty member**