A Mini Project Report

Submitted *by*

Mr. Abhijeet Bhaskar (Roll No: 11)

Mr. Atul Thete (Roll No.: 49)

Miss. Pranjal Thorat (Roll No.: 50)

Miss. Aditi Bhavsar (Roll No.: 54)

Miss. Aditi Shinde (Roll No.: 72)

submitted in partial fulfillment of the requirements for

the award of the degree of

Bachelor

in

COMPUTER ENGINEERING

For Academic Year 2022-2023

Under the guidance of

Prof. Archana Banait

DEPARTMENT OF COMPUTER ENGINEERING

MET's Institute of Engineering Bhujbal Knowledge City

Adgaon, Nashik-422003

THE MET LEAGUE OF COLLEGES
MET
AS SHARP AS YOU CAN GET

Bhujbal Knowledge City

Institute of Engineering

Department of Computer Engineering

Certificate

This is to Certify that

Mr. Abhijeet Bhaskar (Roll No: 11)

has completed the necessary DATA SCIENCE AND BIG DATA ANALYSIS Mini Project and prepared the report

in satisfactory manner as a fulfillment of the requirement of the award of degree of Bachelor of Computer Engineering in the Academic year 2022-2023

Seminar Guide	H.O.D	Principal
Prof. Archana Banait	Dr. M. U. Kharat.	Dr. V. P. Wani

THE MET LEAGUE OF COLLEGES
MET
AS SHARP AS YOU CAN GET

Bhujbal Knowledge City

Institute of Engineering

Department of Computer Engineering

Certificate

This is to Certify that

Mr. Atul Thete (Roll No.: 49)

has completed the necessary DATA SCIENCE AND BIG DATA ANALYSIS Mini Project and prepared the report

in satisfactory manner as a fulfillment of the requirement of the award of degree of Bachelor of Computer Engineering in the Academic year 2022-2023

Seminar Guide	H.O.D	Principal
Prof. Archana Banait	Dr. M. U. Kharat.	Dr. V. P. Wani

THE MET LEAGUE OF COLLEGES
MET
AS SHARP AS YOU CAN GET

Bhujbal Knowledge City

Institute of Engineering

Department of Computer Engineering

Certificate

This is to Certify that

Miss. Pranjal Thorat (Roll No.: 50)

has completed the necessary DATA SCIENCE AND BIG DATA ANALYSIS Mini Project and prepared the report

in satisfactory manner as a fulfillment of the requirement of the award of degree of Bachelor of Computer Engineering in the Academic year 2022-2023

Seminar Guide	H.O.D	Principal
Prof. Archana Banait	**Dr. M. U. Kharat.**	**Dr. V. P. Wani**

THE MET LEAGUE OF COLLEGES
MET
AS SHARP AS YOU CAN GET

Bhujbal Knowledge City

Institute of Engineering

Department of Computer Engineering

Certificate

This is to Certify that

Miss. Aditi Bhavsar (Roll No.: 54)

has completed the necessary DATA SCIENCE AND BIG DATA ANALYSIS Mini Project and prepared the report

in satisfactory manner as a fulfillment of the requirement of the award of degree of Bachelor of Computer Engineering in the Academic year 2022-2023

Seminar Guide	H.O.D	Principal
Prof. Archana Banait	**Dr. M. U. Kharat.**	**Dr. V. P. Wani**

THE MET LEAGUE OF COLLEGES
MET
AS SHARP AS YOU CAN GET

Bhujbal Knowledge City

Institute of Engineering
Department of Computer Engineering

Certificate

This is to Certify that

Miss. Aditi Shinde (Roll No.: 72)

has completed the necessary DATA SCIENCE AND BIG DATA ANALYSIS Mini Project and prepared the report

in satisfactory manner as a fulfillment of the requirement of the award of degree of Bachelor of Computer Engineering in the Academic year 2022-2023

Seminar Guide	H.O.D	Principal
Prof. Archana Banait	Dr. M. U. Kharat.	Dr. V. P. Wani

Acknowledgements

Every work is source which requires support from many people and areas. It gives us proud privilege to complete the Data Science And Big Data Analysis Mini Project Report under valuable guidance and encouragement of our guide **Prof. Archana Banait.**

We are also extremely grateful to our respected **H.O.D. Dr. M. U. Kharat** for providing all facilities and every help for smooth progress of our Mini Project.

At last we would like to thank all the staff members and our students who directly or indirectly supported me without which the Mini Project work would not have been completed successfully.

by

Mr. Abhijeet Bhaskar (Roll No: 11)

Mr. Atul Thete (Roll No.: 49)

Miss. Pranjal Thorat (Roll No.: 50)

Miss. Aditi Bhavsar (Roll No.: 54)

Miss. Aditi Shinde (Roll No.: 72)

Movie Recommendation System

Contents

1. PROBLEM STATEMENT

Develop a movie recommendation model using the scikit-learn library in python.

2. OBJECTIVE

The objective of this recommendation system is to provide satisfactory movie recommendations to users while keeping the system user friendly i.e. by taking minimum input from users. It recommends the movies based on metadata of the movies and past user ratings.

3. TECHNOLOGY USED

3.1 Machine Learning Library:

- Pandas
- numpy
- difflib
- AST
- scikit-learn

Requirements:

- Python 3.6

4. What is scikit-learn?

Scikit-Learn is a free machine learning library for Python. It supports both supervised and unsupervised machine learning, providing diverse algorithms for classification, regression, clustering, and dimensionality reduction. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis

Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.
The vision for the library is a level of robustness and support required for use in production systems. This means a deep focus on concerns such as easy of use, code quality, collaboration, documentation and performance.

Although the interface is Python, c-libraries are leverage for performance such as numpy for arrays and matrix operations.
It was originally called **scikits.learn** and was initially developed by David Cournapeau as a Google summer of code project in 2007. Later, in 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, and Vincent Michel, from FIRCA (French Institute for Research in Computer Science and Automation), took this project at another level and made the first public release (v0.1 beta) on 1st Feb. 2010.

FEATURES:

The library is focused on modelling data. It is not focused on loading, manipulating and summarizing data. For these features, refer to NumPy and Pandas. Some popular groups of models provided by scikit-learn include:

- **Clustering**: for grouping un labelled data such as K Means.
- **Cross Validation**: for estimating the performance of supervised models on unseen data.
- **Datasets**: for test datasets and for generating datasets with specific properties for investigating model behaviour.

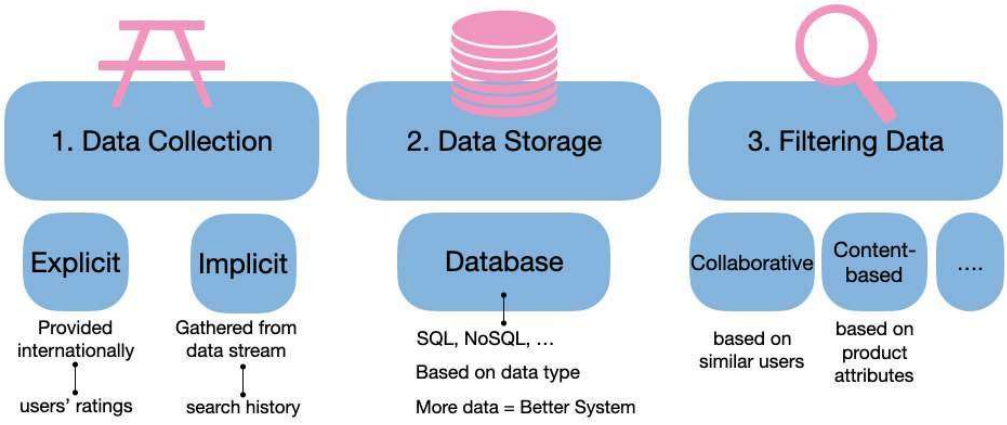
- **Dimensionality Reduction**: for reducing the number of attributes in data for summarization, visualization and feature selection such as Principal component analysis.
- **Ensemble methods**: for combining the predictions of multiple supervised models.
- **Feature extraction**: for defining attributes in image and text data.
- **Feature selection**: for identifying meaningful attributes from which to create supervised models.
- **Parameter Tuning**: for getting the most out of supervised models.
- **Manifold Learning**: For summarizing and depicting complex multi-dimensional data.
- **Supervised Models**: a vast array not limited to generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees

5. What is a Recommendation System?

Simply put a Recommendation System is a filtration program whose prime goal is to predict the "rating" or "preference" of a user towards a domain-specific item or item. In our case, this domain-specific item is a movie, therefore the main focus of our recommendation system is to filter and predict only those movies which a user would prefer given some data about the user him or herself.

Recommendation System Mechanism:

The engine of the recommendation system filters the data via different machine learning algorithms, and based on that filtering, it can predicts the most relevant entities to be recommended. After studying the previous behaviours of the users, it recommends products/services that the used may be interested on.



The engine's working of a recommendation is classified in these 3 steps:

❖ Data Collection

The techniques that can be used to collect data are:

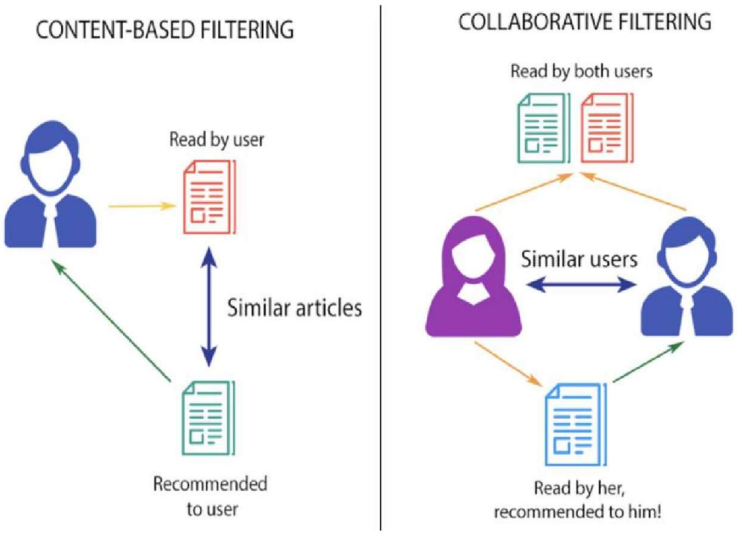
1. Explicit, where data are provided intentionally as an information (e.g. user's input such as movies rating)

2. Implicit, where data are provided intentionally but gathered from available data stream (e.g. search history, clicks, order history, etc…)

Data Storage

It can be stored in a cloud storage such as SQL database, NoSQL database, or some other kind of object storage. However, it depends on the data type and amount as well. The more data that the storage can have for the model, the better recommendation system can be.

6. What are the different filtration strategies?



Content-based Filtering:

This filtration strategy is based on the data provided about the items. The Algorithm recommends products that are similar to the ones that a user has liked in the past. This similarity (generally cosine similarity) is computed from the data we have about the items as well as the user's past preferences.

For example, if a user likes movies such as 'The Prestige' then we can recommend him the movies of 'Christian Bale' or movies with the genre 'Thriller' or maybe even movies directed by 'Christopher Nolan'. So what happens here the recommendation system checks the past preferences of the user and find the film "The Prestige", then tries to find similar movies to that using the information available in the database such as the lead actors, the director, genre of the film, production house, etc and based on this information find movies similar to "The Prestige".

Disadvantages:

1.Different products do not get much exposure to the user.

2.Businesses cannot be expanded as the user does not try different types of products.

Collaborative Filtering:

This filtration strategy is based on the combination of the user's behaviour and comparing and contrasting that with other users' behaviour in the database. The history of all users plays an important role in this algorithm. The main difference between content-based filtering and collaborative filtering that in the latter, the interaction of all users with the items influences the recommendation algorithm while for content-based filtering only the concerned user's data is taken into account. There are multiple ways to implement collaborative filtering but the main concept to be grasped is that in collaborative filtering multiple user's data influences the outcome of the recommendation. and doesn't depend on only one user's data for modelling.

There are 2 types of collaborative filtering algorithms:

❖ User-based Collaborative filtering:

The basic idea here is to find users that have similar past preference patterns as the user 'A' has had and then recommending him or her items liked by those similar users which 'A' has not encountered yet. This is achieved by making a matrix of items each user has rated/viewed/liked/clicked depending upon the task at hand, and then computing the similarity score between the users and finally recommending items that the concerned user isn't aware of but users similar to him/her are and liked it. For example, if the user 'A' likes 'Batman Begins', 'Justice League' and 'The Avengers' while the user 'B' likes 'Batman Begins', 'Justice League' and 'Thor' then they have similar interests because we know that these movies belong to the super-hero genre. So, there is a high probability that the user 'A' would like 'Thor' and the user 'B' would like The Avengers'.

Disadvantages:

1. People are fickle-minded i.e their taste change from time to time and as this algorithm is based on user similarity it may pick up initial similarity patterns between 2 users who after a while may have completely different preferences.
2. There are many more users than items therefore it becomes very difficult to maintain such large matrices and therefore needs to be recomputed very regularly.

3. This algorithm is very susceptible to shilling attacks where fake users profiles consisting of biased preference patterns are used to manipulate key decisions.

❖ Item-based Collaborative Filtering:

The concept in this case is to find similar movies instead of similar users and then recommending similar movies to that 'A' has had in his/her past preferences. This is executed by finding every pair of items that were rated/viewed/liked/clicked by the same user, then measuring the similarity of those rated/viewed/liked/clicked across all user who rated/viewed/liked/clicked both, and finally recommending them based on similarity scores.

Here, for example, we take 2 movies 'A' and 'B' and check their ratings by all users who have rated both the movies and based on the similarity of these ratings, and based on this rating similarity by users who have rated both we find similar movies. So if most common users have rated 'A' and 'B' both similarly and it is highly probable that 'A' and 'B' are similar, therefore if someone has watched and liked 'A' they should be recommended 'B' and vice versa.

Advantages over User-based Collaborative Filtering :

1. Unlike people's taste, movies don't change.
2. There are usually a lot fewer items than people, therefore easier to maintain and compute the matrices.
3. Shilling attacks are much harder because items cannot be faked.

7. Data Description:

A data set (or dataset) is a collection of data. In the case of tabular data, a data set corresponds to one or more database tables, where every column of a table represents a particular variable, and each row corresponds to a given record of the data set in question. The data set lists values for each of the variables, such as for example height and weight of an object, for each member of the data set. Data sets can also consist of a collection of documents or files. In the open data discipline, data set is the unit to measure the information released in a public open data repository. The European Open Data portal aggregates more than half a million data

sets.[2] Some other issues (real-time data sources,[3] non-relational data sets, etc.) increases the difficulty to reach a consensus about it.[

This dataset contain 26 million ratings from 270,000 users for all 45,000 movies listed in the Full Movie Lens Dataset. The dataset consists of movies released on or before July 2017. Data points include cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages.

8. Building a Movie Recommendation System:

The approach to build the movie recommendation engine consists of the following:

1. Perform Exploratory Data Analysis (EDA) on the data.
2. Build the recommendation system.
3. Get recommendations.

- After downloading the dataset, we need to import all the required libraries and then read the csv file using read_csv() method.
- If you visualize the dataset, you will see that it has many extra info about a movie. We don't need all of them. So, we choose keywords, cast, genres and director column to use as our feature set(the so called "content" of the movie).
- If you visualize the dataset, you will see that it has many extra info about a movie. We don't need all of them. So, we choose keywords, cast, genres and director column to use as our feature set(the so called "content" of the movie).
- Now, we need to call this function over each row of our dataframe. But, before doing that, we need to clean and pre-process the data for our use.
- We will fill all the NaN values with blank string in the dataframe. Now that we have obtained the combined strings, we can now feed these strings to a CountVectorizer() object for getting the count matrix.
- At this point, 60% work is done. Now, we need to obtain the cosine similarity matrix from the count matrix.
- Now, we will define two helper functions to get movie title from movie index and vice-versa.
- Our next step is to get the title of the movie that the user currently likes. Then we will find the index of that movie.

- After that, we will access the row corresponding to this movie in the similarity matrix.
- Thus, we will get the similarity scores of all other movies from the current movie. Then we will enumerate through all the similarity scores of that movie to make a tuple of movie index and similarity score.
- This will convert a row of similarity scores like this- [1 0.5 0.2 0.9] to this- [(0, 1) (1, 0.5) (2, 0.2) (3, 0.9)] . Here, each item is in this form- (movie index, similarity score). Now comes the most vital point.
- We will sort the list similar_movies according to similarity scores in descending order. Since the most similar movie to a given movie will be itself, we will discard the first element after sorting the movies.
- Now, we will run a loop to print first 5 entries from sorted_similar_movies list.

9. Results

In [1]:
```
from sklearn.metrics.pairwise import cosine_similarity
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

df = pd.read_csv("movie_dataset.csv")
```

In [2]:
```
df.head()
```

Out[2]:

	index	budget	genres	homepage	id	keywords	origi
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995	culture clash future space war space colony so...	
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	ocean drug abuse exotic island east india trad...	
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy based on novel secret agent sequel mi6	
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	49026	dc comics crime fighter terrorist secret ident...	
4	4	260000000	Action Adventure Science Fiction	http://movies.disney.com/john-carter	49529	based on novel mars medallion space travel pri...	

5 rows × 24 columns

In [3]:
```
features = ['keywords','cast','genres','director']
```

In [4]:
```
def combine_features(row):
    return row['keywords']+" "+row['cast']+" "+row['genres']+" "+row['director
```

In [5]:
```
for feature in features:
    df[feature] = df[feature].fillna('')

df["combined_features"] = df.apply(combine_features,axis=1)
```

In [6]:
```
cv = CountVectorizer()
count_matrix = cv.fit_transform(df["combined_features"])
```

In [7]:
```
cosine_sim = cosine_similarity(count_matrix)
```

In [8]:
```
def get_title_from_index(index):
    return df[df.index == index]["title"].values[0]
def get_index_from_title(title):
    return df[df.title == title]["index"].values[0]
```

In [9]:
```
movie_user_likes = "Avatar"
movie_index = get_index_from_title(movie_user_likes)
similar_movies = list(enumerate(cosine_sim[movie_index]))
```

In [10]:
```
sorted_similar_movies = sorted(similar_movies,key=lambda x:x[1],reverse=True)[
```

In [11]:
```
i=0
print("Top 5 similar movies to "+movie_user_likes+" are:\n")
for element in sorted_similar_movies:
    print(get_title_from_index(element[0]))
    i=i+1
    if i>5:
        break
```

```
Top 5 similar movies to Avatar are:

Guardians of the Galaxy
Aliens
Star Wars: Clone Wars: Volume 1
Star Trek Into Darkness
Star Trek Beyond
Alien
```

In [ ]:

10.CONCLUSION

Recommendation systems have become an important part of everyone's lives. With the enormous number of movies releasing worldwide every year, people often miss out on some amazing work of arts due to the lack of correct suggestion. Putting machine learning based Recommendation systems into work is thus very important to get the right recommendations. The content-based recommendation systems that although may not seem very effective on its own, but when combined with collaborative techniques can solve the cold start problems that collaborative filtering methods face when run independently.

Covid Vaccine State-wise

Contents

1. Introduction

During the current coronavirus pandemic, monitoring the evolution of COVID- 19 cases is of utmost importance for the authorities to make informed policy decisions (e.g., lock-downs), and to raise awareness in the general public for taking appropriate public health measures.

For this reason, given the rapid progression of the pandemic, in some cases health authorities are forced to make important decisions based on sub-optimal data. For this reason, its important to analyze and visualize the data, to better understand the progress of a pandemic. So we decided to perform data analysis using the COVID 19 dataset.

2. Problem Statement

Use the following covid_vaccine_statewise.csv dataset and perform following analytics on the given dataset- https://www.kaggle.com/sudalairajkumar/covid19- in-india?select=covid vaccine_statewise.csv a. Describe the dataset

b. Number of persons state wise vaccinated for first dose in India
c. Number of persons state wise vaccinated for second dose in India
d. Number of Males vaccinated

3. Objectives and Scope

1. One will be able to understand the COVID vaccination data and perform analytics on it.
2. One will be able to visuallize the COVID vaccination data
3. Will be able to describe the COVID database.

4. Methodological Details

1. Importing required libraries: For analyzing data, we need some libraries. In this section, we have imported all the required libraries like pandas, NumPy, matplotlib, plotly, seaborn that were required for data analysis.
2. Loading the dataset: Read the CSV file using pandas read_csv() function and show the output using head() function.

3. Getting Basic Information of the dataset: Identify the dimensions of the dataset, identifying the columns in the dataset.

4. Data Preparation and Cleaning:

   (a) Get an information about the dataset.

   (b) Identify datatypes of the columns and found that the Updated on column is having the object datatype which we to change to the datetime.

   (c) count the number of missing values in each column and found the tech- nique to avoid the missing values that we filled them with the value 0.

   (d) Found out all the unique values for all the states for which this dataset is created.

   (e) In the listed states, India is also present which is not the state. May be it contains the sum of values in all states. But it will change the values in our analysis. So we will will drop the rows containing India as State by using 'drop()' function.

Here, after that our index has changed in the dataframe so resetted index using 'reset index' method.

5. Exploratory Analysis and Visualization- In this section, we have explored relationships between columns by doing visualization using matplotlib and seaborn libraries of python.

5. Modern engineering tools used

5.1 TECHNOLOGIES USED

Python for data preprocessing , visualization etc.

Libraries Used :

1. **Pandas** : Working with data files

   • Python Pandas is defined as an open-source library that provides high performance data manipulation in Python. This tutorial is designed for both beginners and professionals.

   • It is used for data analysis in Python and developed by Wes McKinney in 2008. Our Tutorial provides all the basic and advanced concepts of Python Pandas, such as Numpy, Data operation and Time Series

   • Pandas is built on top of the Numpy package, means Numpy is required for operating the Pandas.

   • Pandas is defined as an open-source library that provides high performance data manipulation in Python. The name of Pandas is derived from the word Panel Data, which means an Econometrics from Multidimensional data. It is used for data analysis in Python and developed by Wes Mckinney in 2008.

2. **Numpy** : For Scientific Calculation

• NumPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements.

• Travis Oliphant created NumPy package in 2005 by injecting the features of the ancestor module Numeric into another module Numarray.

• It is an extension module of Python which is mostly written in C. It provides various functions which are capable of performing the numeric computations with a high speed.

• NumPy provides various powerful data structures, implementing multidimensional arrays and matrices. These data structures are used for the optimal computations regarding arrays and matrices

• To install Numpy use : pip install numpy

• To import numpy use: import numpy

3. **Seaborn** : For Data Visualization

• Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

• Seaborn offers the following functionalities:

• Dataset oriented API to determine the relationship between variables.

• Automatic estimation and plotting of linear regression plots.

• It supports high-level abstractions for multi-plot grids.

• Visualizing univariate and bivariate distribution. k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Syntax of Seaborn:

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.distplot([0, 1, 2, 3, 4, 5]) plt.show()
```

4. **Matplotlib** : Basic Visualization
   • Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.
   • Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.
   • One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc

5.2 DATA PREPROCESSING

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing.

5.3 DATA EXPLORATION

5.3.1 TYPES OF VISUALIZATION USED

1. PIE CHART : To show vaccine status based on gender.

BAR PLOT : To show state wise vaccination for dose 1 and dose

6. Observations

1. Most of the people from Uttar Pradesh were vaccinated for the first dose of vaccine in this time interval.
2. Most of the people from Maharashtra were vaccinated for the second dose of the vaccine in this time interval.
3. Least Vaccinated People were from the Lakshadweep, Ladakh and Andaman and Nicobar.
4. Among the vaccinated people 46.98% were females and 53% were males and other contains 0.02%.

Results:-

In [1]:
```
# Importing the required libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
# Reading the csv file
data = pd.read_csv("covid_vaccine_statewise.csv")
```

In [3]:
```
# Top five rows
print("The top five rows are: ")
data.head()
```

```
The top five rows are:
```

Out[3]:

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	18-44 Years (Doses Administered)	45-60 Year (Dose Administered
0	16/01/2021	India	48276.0	3455.0	2957.0	48276.0	0.0	NaN	NaN	NaN	...	NaN	Nal
1	17/01/2021	India	58604.0	8532.0	4954.0	58604.0	0.0	NaN	NaN	NaN	...	NaN	Nal
2	18/01/2021	India	99449.0	13611.0	6583.0	99449.0	0.0	NaN	NaN	NaN	...	NaN	Nal
3	19/01/2021	India	195525.0	17855.0	7951.0	195525.0	0.0	NaN	NaN	NaN	...	NaN	Nal
4	20/01/2021	India	251280.0	25472.0	10504.0	251280.0	0.0	NaN	NaN	NaN	...	NaN	Nal

5 rows × 24 columns

In [4]:
```
# Last five rows
print("The last five rows are: ")
data.tail()
```

```
The last five rows are:
```

Out[4]:

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	18-44 Years (Doses Administered)	45-60 Ye (Do Administe
7840	11/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	N
7841	12/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	N
7842	13/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	N
7843	14/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	N
7844	15/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	N

5 rows × 24 columns

In [5]:
```
# Shape of the dataset in the format of (rows, columns)
print("The shape is: ")
data.shape
```

```
The shape is:
```

Out[5]: (7845, 24)

In [6]:
```
# Names of columns
print("The columns present in the dataset are: ")
data.columns
```

```
The columns present in the dataset are:
```

Out[6]:
```
Index(['Updated On', 'State', 'Total Doses Administered', 'Sessions',
       ' Sites ', 'First Dose Administered', 'Second Dose Administered',
       'Male (Doses Administered)', 'Female (Doses Administered)',
       'Transgender (Doses Administered)', ' Covaxin (Doses Administered)',
       'CoviShield (Doses Administered)', 'Sputnik V (Doses Administered)',
       'AEFI', '18-44 Years (Doses Administered)',
       '45-60 Years (Doses Administered)', '60+ Years (Doses Administered)',
       '18-44 Years(Individuals Vaccinated)',
       '45-60 Years(Individuals Vaccinated)',
       '60+ Years(Individuals Vaccinated)', 'Male(Individuals Vaccinated)',
       'Female(Individuals Vaccinated)', 'Transgender(Individuals Vaccinated)',
       'Total Individuals Vaccinated'],
      dtype='object')
```

In [7]:
```
data.describe()
```

Out[7]:

	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	Covaxin (Doses Administered)	CoviShield (Doses Administered)	...	A
count	7.621000e+03	7.621000e+03	7621.000000	7.621000e+03	7.621000e+03	7.461000e+03	7.461000e+03	7461.000000	7.621000e+03	7.621000e+03	...	1
mean	9.188171e+06	4.792358e+05	2282.872064	7.414415e+06	1.773755e+06	3.620156e+06	3.168416e+06	1162.978019	1.044669e+06	8.126553e+06	...	8
std	3.746180e+07	1.911511e+06	7275.973730	2.995209e+07	7.570382e+06	1.737938e+07	1.515310e+07	5931.353995	4.452259e+06	3.298414e+07	...	2
min	7.000000e+00	0.000000e+00	0.000000	7.000000e+00	0.000000e+00	0.000000e+00	2.000000e+00	0.000000	0.000000e+00	7.000000e+00	...	2
25%	1.356570e+05	6.004000e+03	69.000000	1.166320e+05	1.283100e+04	5.655500e+04	5.210700e+04	8.000000	0.000000e+00	1.331340e+05	...	4

In [8]:
```
data.describe(include='object')
```

Out[8]:

	Updated On	State
count	7845	7845
unique	213	37
top	16/01/2021	Delhi
freq	37	213

In [9]:
```
# Information about the dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7845 entries, 0 to 7844
Data columns (total 24 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   Updated On                        7845 non-null   object
 1   State                             7845 non-null   object
 2   Total Doses Administered          7621 non-null   float64
 3   Sessions                          7621 non-null   float64
 4    Sites                            7621 non-null   float64
 5   First Dose Administered           7621 non-null   float64
 6   Second Dose Administered          7621 non-null   float64
 7   Male (Doses Administered)         7461 non-null   float64
 8   Female (Doses Administered)       7461 non-null   float64
 9   Transgender (Doses Administered)  7461 non-null   float64
 10   Covaxin (Doses Administered)     7621 non-null   float64
 11  CoviShield (Doses Administered)   7621 non-null   float64
 12  Sputnik V (Doses Administered)    2995 non-null   float64
 13  AEFI                              5438 non-null   float64
 14  18-44 Years (Doses Administered)  1702 non-null   float64
```

```
 15  45-60 Years (Doses Administered)     1702 non-null   float64
 16  60+ Years (Doses Administered)       1702 non-null   float64
 17  18-44 Years(Individuals Vaccinated)  3733 non-null   float64
 18  45-60 Years(Individuals Vaccinated)  3734 non-null   float64
 19  60+ Years(Individuals Vaccinated)    3734 non-null   float64
 20  Male(Individuals Vaccinated)         160 non-null    float64
 21  Female(Individuals Vaccinated)       160 non-null    float64
 22  Transgender(Individuals Vaccinated)  160 non-null    float64
 23  Total Individuals Vaccinated         5919 non-null   float64
dtypes: float64(22), object(2)
memory usage: 1.4+ MB
```

In [10]:
```
data.isnull().sum()
```

Out[10]:
```
Updated On                            0
State                                 0
Total Doses Administered            224
Sessions                            224
 Sites                              224
First Dose Administered             224
Second Dose Administered            224
Male (Doses Administered)           384
Female (Doses Administered)         384
Transgender (Doses Administered)    384
 Covaxin (Doses Administered)       224
CoviShield (Doses Administered)     224
Sputnik V (Doses Administered)     4850
AEFI                               2407
18-44 Years (Doses Administered)    6143
45-60 Years (Doses Administered)    6143
60+ Years (Doses Administered)      6143
18-44 Years(Individuals Vaccinated) 4112
45-60 Years(Individuals Vaccinated) 4111
60+ Years(Individuals Vaccinated)   4111
Male(Individuals Vaccinated)        7685
Female(Individuals Vaccinated)      7685
Transgender(Individuals Vaccinated) 7685
Total Individuals Vaccinated        1926
dtype: int64
```

In [11]:
```
# Average of First Dose Administered
avg_firstdose = data["First Dose Administered"].astype("float").mean(axis = 0)
print("Average of First Dose:", avg_firstdose)
```

```
Average of First Dose: 7414415.300354284
```

In [12]:
```
# Replacing First Dose Administered
data["First Dose Administered"].fillna(value = avg_firstdose, inplace=True)
data
```

Out[12]:

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	18-44 Years (Doses Administered)	45-60 (Administ
0	16/01/2021	India	48276.0	3455.0	2957.0	4.827600e+04	0.0	NaN	NaN	NaN	...	NaN	
1	17/01/2021	India	58604.0	8532.0	4954.0	5.860400e+04	0.0	NaN	NaN	NaN	...	NaN	
2	18/01/2021	India	99449.0	13611.0	6583.0	9.944900e+04	0.0	NaN	NaN	NaN	...	NaN	
3	19/01/2021	India	195525.0	17855.0	7951.0	1.955250e+05	0.0	NaN	NaN	NaN	...	NaN	
4	20/01/2021	India	251280.0	25472.0	10504.0	2.512800e+05	0.0	NaN	NaN	NaN	...	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	

7840	11/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	NaN	NaN	NaN	NaN ...	NaN
7841	12/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	NaN	NaN	NaN	NaN ...	NaN
7842	13/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	NaN	NaN	NaN	NaN ...	NaN
7843	14/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	NaN	NaN	NaN	NaN ...	NaN
7844	15/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	NaN	NaN	NaN	NaN ...	NaN

7845 rows × 24 columns

In [13]:
```
# Average of Second Dose Administered
avg_seconddose = data["Second Dose Administered"].astype("float").mean(axis = 0)
print("Average of Second Dose:", avg_seconddose)
```

```
Average of Second Dose: 1773755.2436688098
```

In [14]:
```
# Replacing Second Dose Administered
data["Second Dose Administered"].fillna(value = avg_seconddose, inplace = True)
data
```

Out[14]:

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	18-44 Years (Doses Administered)	45-60 (Adminis
0	16/01/2021	India	48276.0	3455.0	2957.0	4.827600e+04	0.000000e+00	NaN	NaN	NaN	...	NaN	
1	17/01/2021	India	58604.0	8532.0	4954.0	5.860400e+04	0.000000e+00	NaN	NaN	NaN	...	NaN	
2	18/01/2021	India	99449.0	13611.0	6583.0	9.944900e+04	0.000000e+00	NaN	NaN	NaN	...	NaN	
7840	11/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	1.773755e+06	NaN	NaN	NaN	...	NaN	
7841	12/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	1.773755e+06	NaN	NaN	NaN	...	NaN	
7842	13/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	1.773755e+06	NaN	NaN	NaN	...	NaN	
7843	14/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	1.773755e+06	NaN	NaN	NaN	...	NaN	
7844	15/08/2021	West Bengal	NaN	NaN	NaN	7.414415e+06	1.773755e+06	NaN	NaN	NaN	...	NaN	

7845 rows × 24 columns

In [15]:
```
first_dose = data.groupby('State')[['First Dose Administered']].sum()
first_dose
```

Out[15]:

State	First Dose Administered
Andaman and Nicobar Islands	6.091235e+07
Andhra Pradesh	1.277347e+09
Arunachal Pradesh	9.349147e+07
Assam	6.300867e+08
Bihar	1.514989e+09
Chandigarh	8.918960e+07
Chhattisgarh	8.404894e+08
Dadra and Nagar Haveli and Daman and Diu	8.549597e+07

Chhattisgarh	8.404894e+08
Dadra and Nagar Haveli and Daman and Diu	8.549597e+07
Delhi	6.762404e+08
Goa	1.204779e+08
Gujarat	2.176133e+09
Haryana	8.002848e+08
Himachal Pradesh	3.607805e+08
India	2.830663e+10
Jammu and Kashmir	4.545883e+08
Jharkhand	6.481602e+08
Karnataka	1.917816e+09
Kerala	1.238332e+09
Ladakh	6.229574e+07
Lakshadweep	4.885015e+07
Madhya Pradesh	1.841091e+09
Maharashtra	2.828851e+09
Manipur	1.118961e+08
Meghalaya	1.071025e+08
Mizoram	9.235957e+07
Nagaland	8.689726e+07
Odisha	1.077120e+09
Puducherry	8.583335e+07
Punjab	6.288331e+08
Rajasthan	2.245531e+09
Sikkim	8.146742e+07
Tamil Nadu	1.333019e+09
Telangana	9.248071e+08
Tripura	2.371762e+08
Uttar Pradesh	2.832898e+09
Uttarakhand	4.076779e+08
West Bengal	1.840936e+09

In [16]:
```
first_dose = data.groupby('State')[['Second Dose Administered']].sum()
first_dose
```

Out[16]:

State	Second Dose Administered
Andaman and Nicobar Islands	1.476109e+07
Andhra Pradesh	3.694601e+08
Arunachal Pradesh	2.257485e+07
Assam	1.414313e+08
Bihar	2.814331e+08
Chandigarh	2.223627e+07
Chhattisgarh	1.827629e+08

Dadra and Nagar Haveli and Daman and Diu	1.701070e+07
Delhi	2.006352e+08
Goa	2.684071e+07
Gujarat	6.110609e+08
Haryana	1.692986e+08
Himachal Pradesh	8.448111e+07
India	6.770264e+09
Jammu and Kashmir	9.659418e+07
Jharkhand	1.327636e+08
Karnataka	4.378297e+08
Kerala	3.746913e+08
Ladakh	1.609629e+07
Lakshadweep	1.169898e+07
Madhya Pradesh	3.275755e+08
Maharashtra	7.235236e+08
Manipur	2.250068e+07
Meghalaya	2.280916e+07
Mizoram	2.064095e+07
Nagaland	1.984717e+07
Odisha	2.619453e+08
Puducherry	1.925139e+07
Punjab	1.317635e+08
Rajasthan	5.023455e+08

Rajasthan	5.023455e+08
Sikkim	2.036617e+07
Tamil Nadu	3.013132e+08
Telangana	2.087955e+08
Tripura	7.591267e+07
Uttar Pradesh	5.650776e+08
Uttarakhand	1.107276e+08
West Bengal	5.967894e+08

In [17]:
```
male = data["Male(Individuals Vaccinated)"].sum()
print("The total number of male individuals vaccinated are", int(male))
```

```
The total number of male individuals vaccinated are 7138698858
```

In [18]:
```
female = data["Female(Individuals Vaccinated)"].sum()
print("The total number of female individuals vaccinated are", int(female))
```

```
The total number of female individuals vaccinated are 6321628736
```

In [ ]:

7 Conclusion

COVID-19 Data Analysis plays an important role because depending on the data received and analysed from various sources, government officials can take further step for well-being of people.

Gender information of most of the patients is nor released by the government but whatever data is available shows number of males vaccinated is more than that of females.

It used various libraries such as pandas, seaborn , matplotlib and so on for data analysis and visualization.