EXPERIMENT – 05

AIM: Implement basic operations of Pandas, Numpy and Matplotlib Libraries in Machine Learning.

DESCRIPTION:

Pandas

- **Purpose**: Data manipulation and analysis.
- **Key Features:**
  - **DataFrames**: Two-dimensional, size-mutable, potentially heterogeneous tabular data.
  - **Data Cleaning**: Handling missing values, filtering, and transforming data.
  - **Data Aggregation**: Grouping data and performing operations like sum, mean, etc.
  - **File I/O**: Easily read/write data from/to CSV, Excel, SQL databases, etc.

NumPy

- **Purpose**: Numerical computing with support for large, multi-dimensional arrays and matrices.
- **Key Features**:
  - **Arrays**: N-dimensional arrays for efficient storage and computation.
  - **Mathematical Functions**: Operations on arrays (e.g., addition, multiplication) and statistical functions (e.g., mean, median).
  - **Linear Algebra**: Functions for matrix operations and transformations.
  - **Random Sampling**: Tools for generating random numbers and samples.

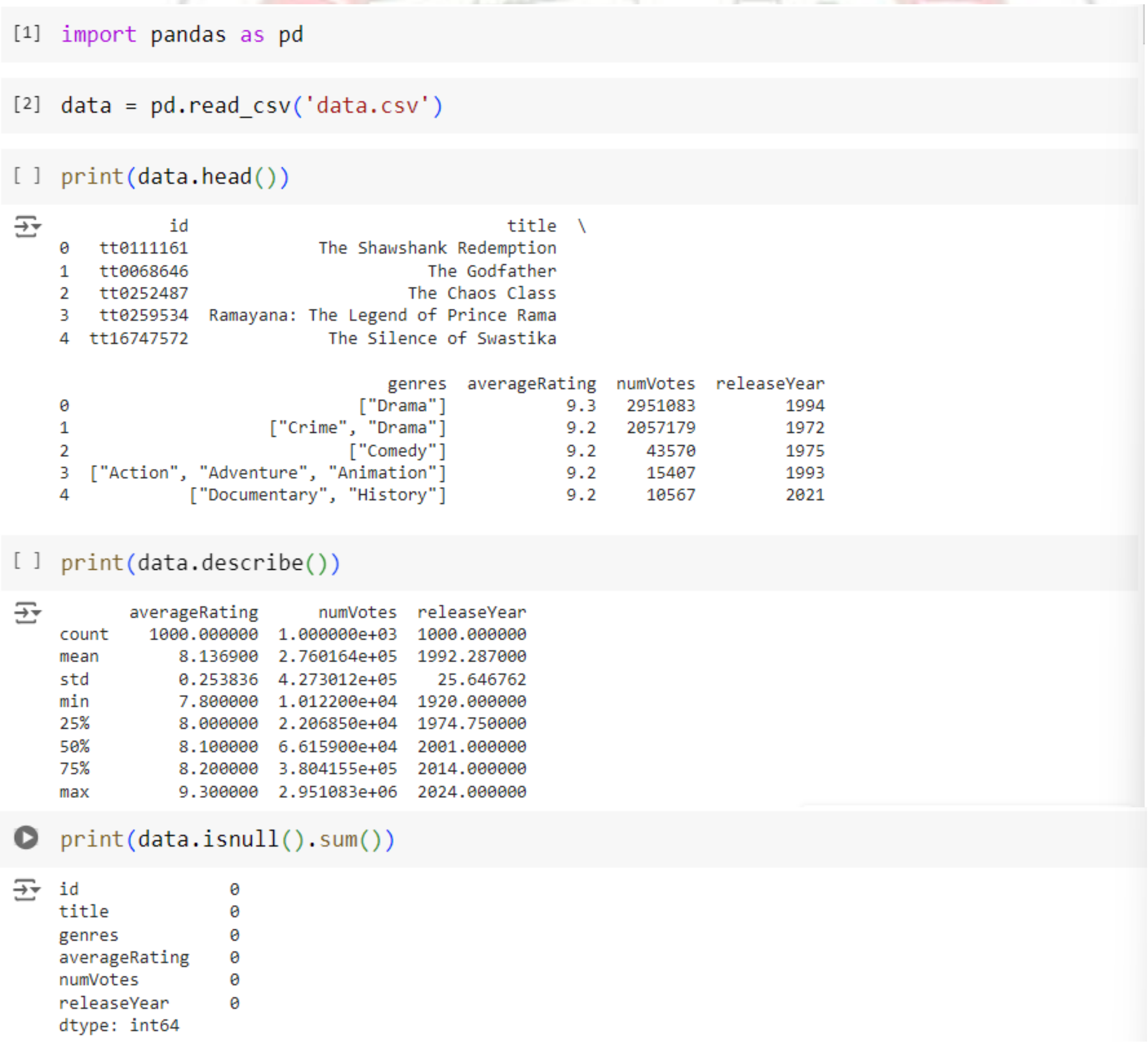
Matplotlib

- **Purpose**: Data visualization.
- **Key Features**:
  - **2D Plotting**: Create a variety of static, animated, and interactive plots.
  - **Customization**: Control over plot aesthetics (colors, labels, titles).
  - **Multiple Plot Types**: Line plots, scatter plots, histograms, bar charts, etc.
  - **Integration**: Works well with Pandas and NumPy for visualizing data directly from those libraries.

Machine Learning Workflow

- **Data Loading**: Use Pandas to read datasets into DataFrames.
- **Data Preprocessing**: Clean and manipulate data with Pandas (e.g., handling missing values).
- **Numerical Operations**: Use NumPy for computations and transformations on data arrays.
- **Model Training**: Apply machine learning algorithms (e.g., from libraries like scikit-learn) using the cleaned data.
- **Predictions**: Make predictions with the trained model.
- **Data Visualization**: Use Matplotlib to visualize the results and insights (e.g., comparing actual vs. predicted values).

CODE:



```
[1] import pandas as pd
```

```
[2] data = pd.read_csv('data.csv')
```

```
[ ] print(data.head())
```

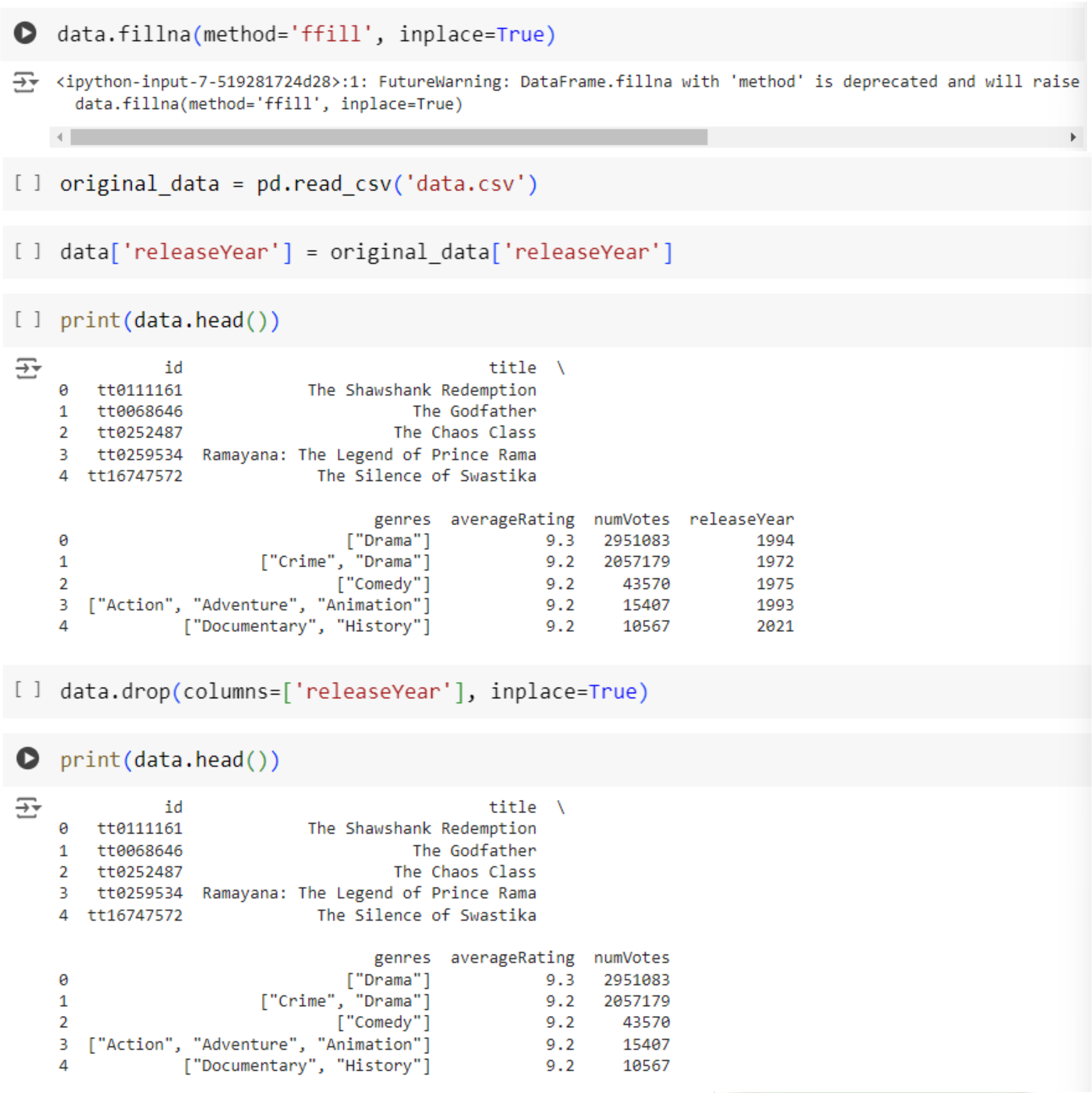
```
          id                            title  \
0   tt0111161          The Shawshank Redemption
1   tt0068646                     The Godfather
2   tt0252487                   The Chaos Class
3   tt0259534  Ramayana: The Legend of Prince Rama
4  tt16747572          The Silence of Swastika

                                  genres  averageRating  numVotes  releaseYear
0                              ["Drama"]            9.3   2951083         1994
1                     ["Crime", "Drama"]            9.2   2057179         1972
2                             ["Comedy"]            9.2     43570         1975
3  ["Action", "Adventure", "Animation"]            9.2     15407         1993
4              ["Documentary", "History"]            9.2     10567         2021
```

```
[ ] print(data.describe())
```

```
       averageRating      numVotes  releaseYear
count    1000.000000  1.000000e+03  1000.000000
mean        8.136900  2.760164e+05  1992.287000
std         0.253836  4.273012e+05    25.646762
min         7.800000  1.012200e+04  1920.000000
25%         8.000000  2.206850e+04  1974.750000
50%         8.100000  6.615900e+04  2001.000000
75%         8.200000  3.804155e+05  2014.000000
max         9.300000  2.951083e+06  2024.000000
```

```
print(data.isnull().sum())
```

```
id               0
title            0
genres           0
averageRating    0
numVotes         0
releaseYear      0
dtype: int64
```



```
data.fillna(method='ffill', inplace=True)
```

```
<ipython-input-7-519281724d28>:1: FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise
  data.fillna(method='ffill', inplace=True)
```

```
[ ] original_data = pd.read_csv('data.csv')
```

```
[ ] data['releaseYear'] = original_data['releaseYear']
```

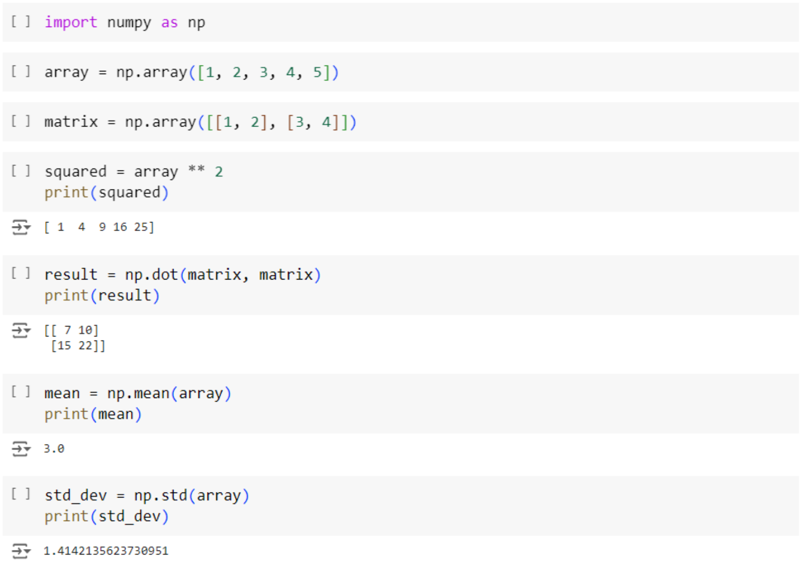
```
[ ] print(data.head())
```

```
          id                              title  \
0  tt0111161           The Shawshank Redemption
1  tt0068646                      The Godfather
2  tt0252487                    The Chaos Class
3  tt0259534  Ramayana: The Legend of Prince Rama
4 tt16747572             The Silence of Swastika

                             genres  averageRating  numVotes  releaseYear
0                         ["Drama"]            9.3   2951083         1994
1                ["Crime", "Drama"]            9.2   2057179         1972
2                        ["Comedy"]            9.2     43570         1975
3 ["Action", "Adventure", "Animation"]          9.2     15407         1993
4          ["Documentary", "History"]          9.2     10567         2021
```

```
[ ] data.drop(columns=['releaseYear'], inplace=True)
```

```
print(data.head())
```

```
          id                              title  \
0  tt0111161           The Shawshank Redemption
1  tt0068646                      The Godfather
2  tt0252487                    The Chaos Class
3  tt0259534  Ramayana: The Legend of Prince Rama
4 tt16747572             The Silence of Swastika

                             genres  averageRating  numVotes
0                         ["Drama"]            9.3   2951083
1                ["Crime", "Drama"]            9.2   2057179
2                        ["Comedy"]            9.2     43570
3 ["Action", "Adventure", "Animation"]          9.2     15407
4          ["Documentary", "History"]          9.2     10567
```



```
[ ] import numpy as np
```

```
[ ] array = np.array([1, 2, 3, 4, 5])
```

```
[ ] matrix = np.array([[1, 2], [3, 4]])
```

```
[ ] squared = array ** 2
    print(squared)
```

```
[ 1  4  9 16 25]
```

```
[ ] result = np.dot(matrix, matrix)
    print(result)
```

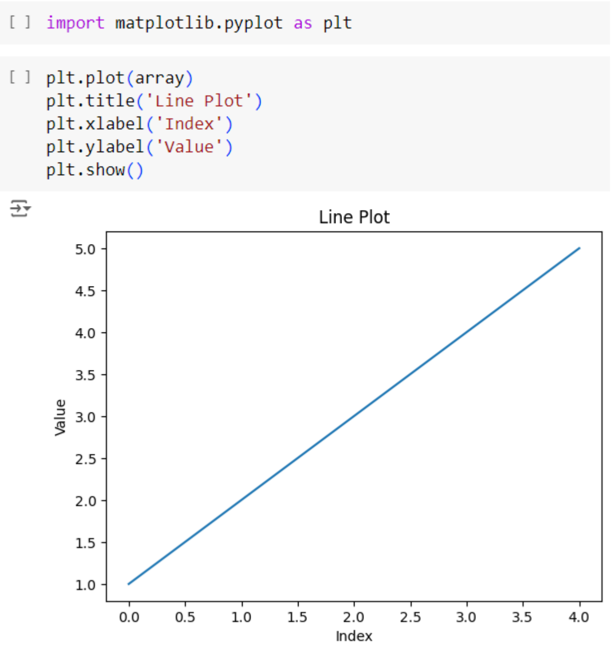
```
[[ 7 10]
 [15 22]]
```

```
[ ] mean = np.mean(array)
    print(mean)
```

```
3.0
```

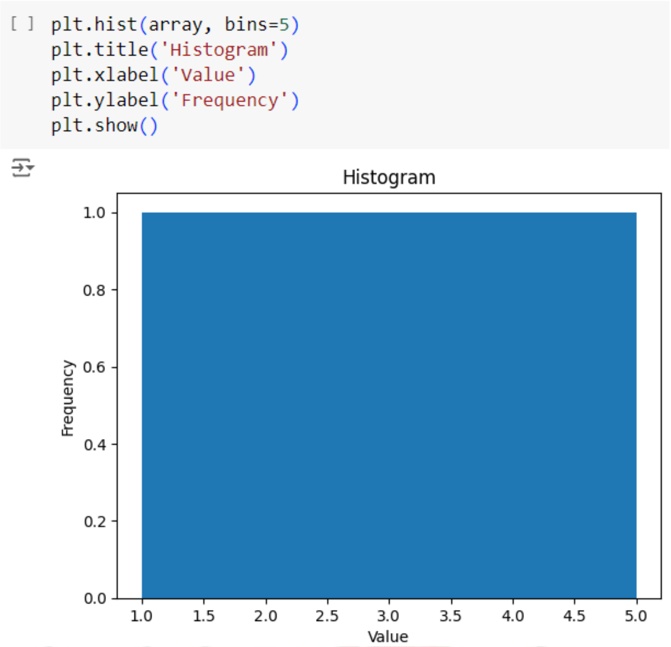
```
[ ] std_dev = np.std(array)
    print(std_dev)
```

```
1.4142135623730951
```

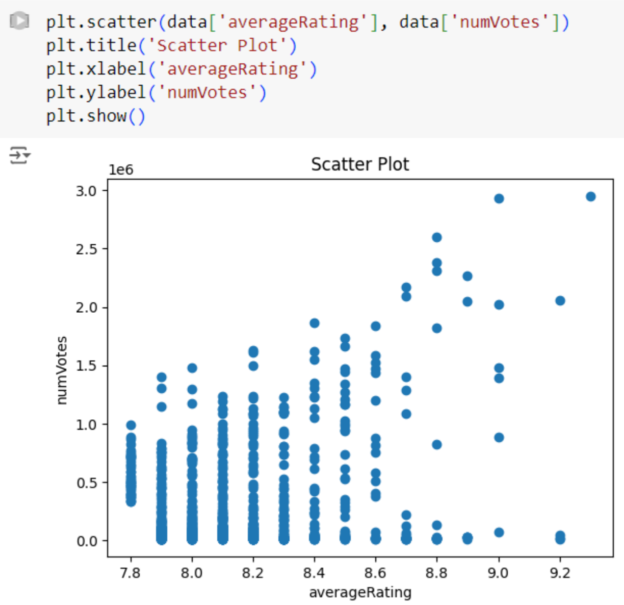


```
[ ] import matplotlib.pyplot as plt
```

```
[ ] plt.plot(array)
    plt.title('Line Plot')
    plt.xlabel('Index')
    plt.ylabel('Value')
    plt.show()
```



```
[ ] plt.hist(array, bins=5)
    plt.title('Histogram')
    plt.xlabel('Value')
    plt.ylabel('Frequency')
    plt.show()
```



```
plt.scatter(data['averageRating'], data['numVotes'])
plt.title('Scatter Plot')
plt.xlabel('averageRating')
plt.ylabel('numVotes')
plt.show()
```

OUTPUT ANALYSIS:

The output displays a scatter plot comparing actual vs. predicted values from the regression model. The closer the points are to the red line, the better the model's predictions. This visualization helps assess the model's performance, highlighting areas of over- or under-prediction.