L&T Mid-Term Assignment

Statistics and Probability Refresher, and Python Practice

Student Information

Name: Aniruddha Purushottam Somani

Course: Python Programming for Data Handling and Preprocessing

Branch: Computer Science Engineering (AIML)

Year & Semester: 1st Year & IInd Semester

College: Sir Padampat Singhania University, Udaipur

1. Basic Array Operations

Project Outlines:

- Create NumPy arrays from lists and initialize arrays with different data types.
- Reshape arrays into different dimensions for efficient computation.
- Perform element-wise operations such as addition, subtraction, and multiplication.

Expected Outcome:

- Students will understand how to create and manipulate NumPy arrays, perform arithmetic operations, and reshape data structures for better handling in computational tasks.

```
import numpy as np

list_data = [1, 2, 3, 4, 5]
array_from_list = np.array(list_data, dtype=float)
print("Array from list:", array_from_list)

zeros_array = np.zeros((2, 3))
ones_array = np.ones((3, 2))
range_array = np.arange(1, 10, 2)
print("Zeros array:\n", zeros_array)
print("Ones array:\n", ones_array)
print("Range array:", range_array)

reshaped_array = range_array.reshape((3, 1))
print("Reshaped array:\n", reshaped_array)

array_a = np.array([1, 2, 3])
array_b = np.array([4, 5, 6])

sum_array = array_a + array_b
diff_array = array_b - array_a
product_array = array_a * array_b
print("Sum:", sum_array)
print("Difference:", diff_array)
print("Product:", product_array)
```

Answer 1

```
Array from list: [1. 2. 3. 4. 5.]
Zeros array:
 [[0. 0. 0.]
 [0. 0. 0.]]
Ones array:
 [[1. 1.]
 [1. 1.]
 [1. 1.]]
Range array: [1 3 5 7 9]
Reshaped array:
 [[1]
 [3]
 [5]
 [7]
 [9]]
Sum: [5 7 9]
Difference: [3 3 3]
Product: [ 4 10 18]
```

2. Matrix Operations

Project Outlines:
- Implement fundamental matrix operations, including addition and multiplication.
- Compute the transpose of a matrix for data transformation.
- Calculate the inverse of a matrix and understand its applications.

Expected Outcome:
- Students will gain a solid understanding of matrix operations, which are essential in data science, computer graphics, and engineering applications.

```
import numpy as np

matrix_a = np.array([[1, 2], [3, 4]])
matrix_b = np.array([[5, 6], [7, 8]])

matrix_sum = matrix_a + matrix_b
print("Matrix Addition:\n", matrix_sum)

elementwise_product = matrix_a * matrix_b
matrix_product = np.dot(matrix_a, matrix_b)
print("Element-wise Product:\n", elementwise_product)
print("Matrix Product:\n", matrix_product)

transpose_matrix = matrix_a.T
print("Transpose of Matrix A:\n", transpose_matrix)

matrix_c = np.array([[4, 7], [2, 6]])
inverse_matrix = np.linalg.inv(matrix_c)
print("Inverse of Matrix C:\n", inverse_matrix)
```

Answer 2

```
Matrix Addition:
 [[ 6  8]
 [10 12]]
Element-wise Product:
 [[ 5 12]
 [21 32]]
Matrix Product:
 [[19 22]
 [43 50]]
Transpose of Matrix A:
 [[1 3]
 [2 4]]
Inverse of Matrix C:
 [[ 0.6 -0.7]
 [-0.2  0.4]]
```

3. Statistical Analysis

Project Outlines:

• Compute key statistical measures such as mean, median, variance, and standard deviation.

• Analyze datasets to understand distributions and variability.

• Apply statistical functions in real-world scenarios like financial or experimental data analysis.

Expected Outcome:

• Students will be able to analyze data distributions, interpret statistical measures, and apply them in research, business, and machine learning tasks.

Answer 3

```
import numpy as np

data = [15, 20, 21, 19, 22, 24, 17, 30, 25, 22]
mean = np.mean(data)
median = np.median(data)
variance = np.var(data)
std_dev = np.std(data)

print("Mean:", mean)
print("Median:", median)
print("Variance:", variance)
print("Standard Deviation:", std_dev)

min_val, max_val = min(data), max(data)
num_bins = 5
bin_size = (max_val - min_val) // num_bins + 1
bins = [min_val + i * bin_size for i in range(num_bins + 1)]
histogram = [0] * num_bins

for value in data:
    for i in range(len(bins) - 1):
        if bins[i] <= value < bins[i + 1]:
            histogram[i] += 1
            break

print("Data Distribution (Histogram):")
for i in range(len(histogram)):
    print(f"{bins[i]}-{bins[i + 1] - 1}: {'*' * histogram[i]}")

stock_prices = [120, 125, 130, 128, 127, 132, 135]
price_mean = np.mean(stock_prices)
price_variance = np.var(stock_prices)
print("Stock Price Mean:", price_mean)
print("Stock Price Variance:", price_variance)
```

```
Mean: 21.5
Median: 21.5
Variance: 16.25
Standard Deviation: 4.031128874149275
Mean: 21.5
Median: 21.5
Variance: 16.25
Standard Deviation: 4.031128874149275
Data Distribution (Histogram):
15-18: **
19-22: *****
23-26: **
27-30: *
31-34:
Stock Price Mean: 128.14285714285714
Stock Price Variance: 20.40816326530612
Data Distribution (Histogram):
15-18: **
19-22: ******
23-26: **
27-30: *
31-34:
Stock Price Mean: 128.14285714285714
Stock Price Variance: 20.40816326530612
```

4. Random Number Generator

Project Outlines:
• Generate random numbers using NumPy’s random module.
• Analyze distributions such as uniform and normal.
• Apply random number generation in simulations and probability experiments.
Expected Outcome:
• Students will understand how random number generation works, how to simulate data for statistical testing, and how probability distributions are used in machine learning and finance.

Answer 4

```
import numpy as np

random_uniform = np.random.uniform(0, 1, 10)
random_normal = np.random.normal(0, 1, 10)

print("Uniform Distribution:")
print(random_uniform)

print("\nNormal Distribution:")
print(random_normal)

num_tosses = 1000
coin_toss = np.random.choice(['Heads', 'Tails'], size=num_tosses)
heads_count = np.sum(coin_toss == 'Heads')
tails_count = np.sum(coin_toss == 'Tails')

print("\nCoin Toss Simulation:")
print(f"Heads: {heads_count}")
print(f"Tails: {tails_count}")

num_points = 10000

x = np.random.uniform(-1, 1, num_points)
y = np.random.uniform(-1, 1, num_points)

inside_circle = x**2 + y**2 <= 1

pi_estimate = 4 * np.sum(inside_circle) / num_points
print(f"Estimated value of Pi using Monte Carlo simulation: {pi_estimate:.4f}")
```

```
Uniform Distribution:
[0.77649368 0.79717804 0.83624784 0.79069417 0.00486274 0.64068657
 0.49283877 0.65428074 0.21805659 0.96586906]

Normal Distribution:
[-0.06857172 -0.03744472  1.52413744  0.86321552 -0.92669716 -0.69652297
 -0.70415875  1.6269318  -1.12256807  0.08168761]

Coin Toss Simulation:
Heads: 486
Tails: 514
Estimated value of Pi using Monte Carlo simulation: 3.1268
```

5. Basic DataFrame Operations

Project Outlines:

- **Create** DataFrames from lists and dictionaries using Pandas.
- Modify, filter, and sort data within a DataFrame.
- Perform indexing and slicing operations for efficient data retrieval.

Expected Outcome:

- Students will gain hands-on experience with Pandas and learn to manage structured datasets efficiently, preparing them for further data analysis.

Answer 5

```
import pandas as pd

data_dict = {'Name': ['Amit', 'Sumit', 'Rahul'], 'Age': [25, 30, 35], 'City': ['Delhi', 'Mumbai', 'Banglore']}
df_from_dict = pd.DataFrame(data_dict)
print("DataFrame from dictionary:\n", df_from_dict)

data_list = [['Virat', 28, 'Udaipur'], ['Rohit', 22, 'Agra']]
df_from_list = pd.DataFrame(data_list, columns=['Name', 'Age', 'City'])
print("\nDataFrame from list:\n", df_from_list)

df_from_dict['Salary'] = [70000, 80000, 90000]
print("\nDataFrame after adding column:\n", df_from_dict)

filtered_df = df_from_dict[df_from_dict['Age'] > 28]
print("\nFiltered DataFrame (Age > 28):\n", filtered_df)

sorted_df = df_from_dict.sort_values(by='Salary', ascending=False)
print("\nSorted DataFrame by Salary:\n", sorted_df)

selected_row = df_from_dict.loc[1]
print("\nSelected row (index 1):\n", selected_row)

subset = df_from_dict.iloc[0:2, 1:3]
print("\nSubset of DataFrame (iloc):\n", subset)
```

```
DataFrame from dictionary:
     Name  Age      City
0   Amit   25     Delhi
1  Sumit   30    Mumbai
2  Rahul   35  Banglore

DataFrame from list:
     Name  Age     City
0  Virat   28  Udaipur
1  Rohit   22     Agra

DataFrame after adding column:
     Name  Age      City  Salary
0   Amit   25     Delhi   70000
1  Sumit   30    Mumbai   80000
2  Rahul   35  Banglore   90000

Filtered DataFrame (Age > 28):
     Name  Age      City  Salary
1  Sumit   30    Mumbai   80000
2  Rahul   35  Banglore   90000

Sorted DataFrame by Salary:
     Name  Age      City  Salary
2  Rahul   35  Banglore   90000
1  Sumit   30    Mumbai   80000
0   Amit   25     Delhi   70000

Selected row (index 1):
 Name       Sumit
Age           30
City      Mumbai
Salary     80000
Name: 1, dtype: object

Subset of DataFrame (iloc):
    Age    City
0   25   Delhi
1   30  Mumbai
```

6. CSV File Handling

Project Outlines:
- Read data from CSV files into Pandas DataFrames.
- Modify and manipulate CSV data, including filtering and aggregations.
- Save updated DataFrames back to CSV for data persistence.

Expected Outcome:
- Students will develop the skills to handle structured data files, making them proficient in real-world data handling and preprocessing tasks.

Answer 6

```
import pandas as pd

df = pd.read_csv('example.csv')
print("Original DataFrame:\n", df)

df['New_Column'] = df['Column1'] * 2
print("\nDataFrame after adding new column:\n", df)

filtered_df = df[df['Column2'] > 50]
print("\nFiltered DataFrame (Column2 > 50):\n", filtered_df)

aggregated_df = df.groupby('Category')['Column3'].mean().reset_index()
print("\nAggregated DataFrame (Mean of Column3):\n", aggregated_df)

filtered_df.to_csv('filtered_data.csv', index=False)
print("\nFiltered DataFrame saved to 'filtered_data.csv'")
```

7. Exploring a Dataset

Project Outlines:
• Load real-world datasets (e.g., Titanic dataset) into Pandas.
• Perform basic exploratory data analysis (EDA), including summary statistics and data visualization.
• Identify trends and patterns in the dataset.
Expected Outcome:
• Students will be able to explore and summarize datasets effectively, gaining insights that help in data-driven decision-making.

Answer 7

```
import pandas as pd

df = pd.read_csv('titanic.csv')
print("First 5 Rows:\n", df.head())
print("\nData Summary:\n", df.info())

print("\nSummary Statistics:\n", df.describe())

missing_values = df.isnull().sum()
print("\nMissing Values:\n", missing_values)

df['Age'].fillna(df['Age'].mean(), inplace=True)

gender_survival = df.groupby('Sex')['Survived'].mean()
print("\nSurvival Rates by Gender:\n", gender_survival)

class_survival = df.groupby('Pclass')['Survived'].mean()
print("\nSurvival Rates by Passenger Class:\n", class_survival)

combined_trends = df.groupby(['Pclass', 'Sex'])['Survived'].mean().unstack()
print("\nSurvival Rates by Gender and Class:\n", combined_trends)
```

8. Data Cleaning

Project Outlines:

- Detect and handle missing values in datasets.
- Remove duplicate entries to ensure data consistency.
- Correct incorrect or inconsistent data entries for accurate analysis.

Expected Outcome:

- Students will be able to clean and preprocess raw data, ensuring that datasets are ready for analysis and machine learning applications.

Answer 8

```
import pandas as pd

data ={
   'Name': ['Alice', 'Bob', 'Charlie', 'Alice', 'Eve'],
   'Age': [25, 30, None, 25, 22],
   'Gender': ['Female', 'Male', 'male', 'Female', 'female'],
   'City': ['New York', 'Paris', 'London', 'New York', None]
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

print("\nMissing Values:\n", df.isnull().sum())
df['Age'].fillna(df['Age'].mean(), inplace=True)
df.dropna(subset=['City'], inplace=True)
print("\nDataFrame after handling missing values:\n", df)

df.drop_duplicates(inplace=True)
print("\nDataFrame after removing duplicates:\n", df)

df['Gender'] = df['Gender'].str.lower()
df['Gender'].replace({'female': 'Female', 'male': 'Male'}, inplace=True)
print("\nDataFrame after correcting inconsistent entries:\n", df)
```

9. Feature Encoding

(Student Performance Dataset)

Project Outlines:

- Convert categorical variables (e.g., gender, parental education) into numerical form.
- Apply one-hot encoding and label encoding techniques.
- Prepare datasets for machine learning models.

Expected Outcome:

- Students will understand how to transform categorical data into a machine-learning-friendly format, an essential step in predictive modeling.

Answer 9

```
import pandas as pd

data = {
   'Name': ['Alice', 'Bob', 'Charlie', 'David'],
   'Gender': ['Female', 'Male', 'Male', 'Female'],
   'Parental Education': ['High School', 'Bachelor', 'Master', 'High School']
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

gender_mapping = {'Female': 0, 'Male': 1}
df['Gender_Numeric'] = df['Gender'].map(gender_mapping)
print("\nDataFrame after mapping Gender to numeric:\n", df)

education_labels = {level: idx for idx, level in enumerate(df['Parental Education'].unique())}
df['Parental_Education_Label'] = df['Parental Education'].map(education_labels)
print("\nDataFrame after manually Label Encoding 'Parental Education':\n", df)

one_hot_encoded = pd.get_dummies(df['Parental Education'], prefix='Education')
df = pd.concat([df, one_hot_encoded], axis=1)
print("\nDataFrame after One-Hot Encoding 'Parental Education':\n", df)

df.drop(['Parental Education', 'Name'], axis=1, inplace=True)
print("\nFinal Preprocessed Dataset:\n", df)
```

10. Feature Scaling

(Boston House Prices Dataset)

Project Outlines:

- Normalize and standardize numerical features for better model performance.
- Apply Min-Max scaling and StandardScaler from Scikit-Learn.
- Understand the impact of scaling on different machine learning algorithms.

Expected Outcome:

- Students will learn how to scale data appropriately, improving the accuracy and performance of predictive models.

Answer 10

```
import pandas as pd

data = {
   'Feature1': [10, 20, 30],
   'Feature2': [5, 15, 25]
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

df['Feature1_MinMax'] = (df['Feature1'] - df['Feature1'].min()) / (df['Feature1'].max() - df['Feature1'].min())
df['Feature2_MinMax'] = (df['Feature2'] - df['Feature2'].min()) / (df['Feature2'].max() - df['Feature2'].min())

print("\nDataFrame after Min-Max Scaling:\n", df[['Feature1_MinMax', 'Feature2_MinMax']])

df['Feature1_Standard'] = (df['Feature1'] - df['Feature1'].mean()) / df['Feature1'].std()
df['Feature2_Standard'] = (df['Feature2'] - df['Feature2'].mean()) / df['Feature2'].std()

print("\nDataFrame after Standard Scaling:\n", df[['Feature1_Standard', 'Feature2_Standard']])
```

11. Sales Data Analysis

Project Outlines:
• Compute total revenue and analyze sales performance.
• Identify top-selling products and customer purchasing trends.
• Use aggregation functions to derive business insights.
Expected Outcome:
• Students will develop business intelligence skills by analyzing sales data and extracting valuable insights for decision-making.

Answer 11

```
import pandas as pd

data = {
   'Product': ['X', 'Y', 'X', 'Z'],
   'Price': [50, 100, 50, 150],
   'Quantity': [3, 2, 1, 4]
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

df['Revenue'] = df['Price'] * df['Quantity']
total_revenue = df['Revenue'].sum()
print("\nTotal Revenue:", total_revenue)

top_product = df.groupby('Product')['Revenue'].sum().sort_values(ascending=False)
print("\nRevenue by Product:\n", top_product)
```

12. Weather Data Analysis

Project Outlines:

- Load and analyze temperature and climate data.
- Visualize temperature variations using plots and graphs.
- Identify seasonal trends and anomalies in the dataset.

Expected Outcome:

- Students will gain experience in working with time-series data, identifying weather patterns, and applying statistical techniques to real-world datasets.

Answer 12

```
import pandas as pd

data ={
   'Date': ['2023-01-01', '2023-02-01', '2023-03-01', '2023-04-01'],
   'Temperature': [15, 18, 22, 30]
}

df = pd.DataFrame(data)
df['Date'] = pd.to_datetime(df['Date'])  # Convert to datetime format
df.set_index('Date', inplace=True)
print("Original DataFrame:\n", df)

print("\nSummary Statistics:\n", df.describe())

df['Rolling_Avg'] = df['Temperature'].rolling(window=2).mean()
print("\nTemperature with Rolling Average:\n", df)

anomalies = df[df['Temperature'] > 25]
print("\nAnomalies:\n", anomalies)

df['Month'] = df.index.month
monthly_avg = df.groupby('Month')['Temperature'].mean()
print("\nMonthly Average Temperatures:\n", monthly_avg)
```

Thank You!

Name: Aniruddha Purushottam Somani

Course: Python Programming for Data Handling and Preprocessing

Branch: Computer Science Engineering (AIML)

Year & Semester: 1st Year & IInd Semester

College: Sir Padampat Singhania University, Udaipur