Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Spring, Year:2025), B.Sc. in CSE (Day)

CLP NO: 03

Course Title: Data Mining Lab

Course Code: CSE 436 Section: 213-D3

Student Details

Name	ID
Md. Shakil Hossain	213002042

3.3.5 Lab Task

Problem 1: Download and load a dataset. Now, write a Python program to impute null values (if any) using the average value of it's previous and next value.

Dataset: https://www.kaggle.com/datasets/yasserh/titanic-dataset/data

Code:

```
import pandas as pd
import numpy as np
dataset_name = 'Titanic-Dataset.csv'
df = pd.read_csv(dataset_name)
print(f'Orginal Dataset:\n${df.head()}\n\n')

# Count missing values per column
missing_values = df.isnull().sum()

# Print missing values for each column
print("Missing values in each column:\n", missing_values)

# Function to impute missing values using the average of previous and next values
def inpute_missing_values(df):
    for column in df.select_dtypes(include=[np.number]):  # Only numeric columns
        df[column] = df[column].interpolate(method='linear', limit_direction='both')
    return df

# call function Impute Missing Values
df = inpute_missing_values(df)

print("\n\n\nProcessed Dataset:")
print(df.head())
```

Output 1:

```
Orginal Dataset:
$   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

Fig:2.1.1 Original dataset

```
Missing values in each column:
 PassengerId      0
Survived          0
Pclass            0
Name              0
Sex               0
Age             177
SibSp             0
Parch             0
Ticket            0
Fare              0
Cabin           687
Embarked          2
dtype: int64
```

Fig:2.1.2 Missing value in each column

```
Processed Dataset:
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

Fig:2.1.1 Processed dataset after impute null values

Problem 2: Using one-hot encoding, convert categorical values into numerical values.

Code:

```
import pandas as pd
import numpy as np
dataset_name = 'Titanic-Dataset.csv'
df = pd.read_csv(dataset_name)

# Function to impute missing values using the average of previous and next values
def inpute_missing_values(df):
    for column in df.select_dtypes(include=[np.number]):  # Only numeric columns
        df[column] = df[column].interpolate(method='linear', limit_direction='both')
    return df

# Function to apply one-hot encoding to categorical columns
def one_hot_encode(df):
    categorical_columns = df.select_dtypes(include=['object']).columns  # Identify categorical columns
    df = pd.get_dummies(df, columns=categorical_columns, drop_first=True)
    return df

# Step 1: Impute Missing Values
df = inpute_missing_values(df)
```

```
# Step 2: One-Hot Encoding
df = one_hot_encode(df)

print("\nProcessed Dataset:")
print(df.head())

# # Save the processed file
df.to_csv("Processed_" + dataset_name, index=False)
print(f"\nProcessed dataset saved as Processed_{dataset_name}")
```

Output 2:

```
Processed Dataset:
   PassengerId  Survived  Pclass   Age  SibSp  Parch     Fare  \
0            1         0       3  22.0      1      0   7.2500
1            2         1       1  38.0      1      0  71.2833
2            3         1       3  26.0      0      0   7.9250
3            4         1       1  35.0      1      0  53.1000
4            5         0       3  35.0      0      0   8.0500

   Name_Abbott, Mr. Rossmore Edward  Name_Abbott, Mrs. Stanton (Rosa Hunt)  \
0                             False                                  False
1                             False                                  False
2                             False                                  False
3                             False                                  False
4                             False                                  False

   Name_Abelson, Mr. Samuel  ...  Cabin_F G63  Cabin_F G73  Cabin_F2  \
0                     False  ...        False        False     False
1                     False  ...        False        False     False
2                     False  ...        False        False     False
3                     False  ...        False        False     False
4                     False  ...        False        False     False

   Cabin_F33  Cabin_F38  Cabin_F4  Cabin_G6  Cabin_T  Embarked_Q  Embarked_S
0      False      False     False     False    False       False        True
1      False      False     False     False    False       False       False
2      False      False     False     False    False       False        True
3      False      False     False     False    False       False        True
4      False      False     False     False    False       False        True

[5 rows x 1726 columns]

Processed dataset saved as Processed_Titanic-Dataset.csv
```

Fig:2.2.1 Processed dataset after Using one-hot encoding, convert categorical values into numerical values.

Binning program:

```
import pandas as pd
import numpy as np

# Load dataset (Change filename if needed)
df = pd.read_csv("/content/Titanic-Dataset.csv")

# Select numeric column for binning (e.g., 'Age')
column_name = 'Age'

# Drop NaN values to avoid errors in sorting
df = df.dropna(subset=['Age'])

# Step 1: Sort the column
df = df.sort_values(by=column_name).reset_index(drop=True)

# Step 2: Define number of bins (e.g., 4 bins)
num_bins = 4

# Create bin labels
bin_labels = [f'Bin_{i+1}' for i in range(num_bins)]

# Apply equal-depth binning using qcut (approximately same samples per bin)
df['Binned_Age'] = pd.qcut(df[column_name], q=num_bins, labels=bin_labels)

# Step 3: Compute statistics for each bin
bin_means = df.groupby('Binned_Age')[column_name].mean()
bin_medians = df.groupby('Binned_Age')[column_name].median()
bin_boundaries = df.groupby('Binned_Age')[column_name].agg([min, max])

# Print results
print("\nBinned Data:\n", df[['Age', 'Binned_Age']].head(10))
print("\nMean of each bin:\n", bin_means)
print("\nMedian of each bin:\n", bin_medians)
print("\nBoundaries of each bin:\n", bin_boundaries)

# Save processed dataset
df.to_csv("Processed_Titanic_Binned.csv", index=False)
print("\nProcessed dataset saved as 'Processed_Titanic_Binned.csv'")
```

Output:

```
Binned Data:
     Age Binned_Age
0  0.42      Bin_1
1  0.67      Bin_1
2  0.75      Bin_1
3  0.75      Bin_1
4  0.83      Bin_1
5  0.83      Bin_1
6  0.92      Bin_1
7  1.00      Bin_1
8  1.00      Bin_1
9  1.00      Bin_1

Mean of each bin:
 Binned_Age
Bin_1    12.651788
Bin_2    24.374317
Bin_3    32.880000
Bin_4    49.299435
Name: Age, dtype: float64

Median of each bin:
 Binned_Age
Bin_1    16.0
Bin_2    24.0
Bin_3    33.0
Bin_4    48.0
Name: Age, dtype: float64

Boundaries of each bin:
              min   max
Binned_Age
Bin_1        0.42  20.0
Bin_2       20.50  28.0
Bin_3       28.50  38.0
Bin_4       39.00  80.0

Processed dataset saved as 'Processed_Titanic_Binned.csv'
```