Programming for Bioinformatics (SECB3203)

PROJECT PROPOSAL

Title: Diabetic Prediction using Machine Learning

Group 18

Group Members	Matric Number
HARCHANA A/P ARULAPPAN	**A21EC0028**
MALAVIKA A/P BASKARAN	**B22EC0069**

Table of Contents

1.0 Importing Dataset

Our dataset from kaggle : https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

We import the diabetes.csv file from the local directory in Python and display the result to the first 5 rows.

Progress2.py

D: > Python > Progress2.py > ...

```python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file

# Importing dataset
path = 'D:\\Python\\diabetes.csv'
df = pd.read_csv(path)

# Display first few rows of DataFrame
print(df.head())
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Malavika> & D:/Python/python.exe d:/Python/Progress2.py
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  DiabetesPedigreeFunction  Age  Outcome
0            6      148             72             35        0  33.6                     0.627   50        1
1            1       85             66             29        0  26.6                     0.351   31        0
2            8      183             64              0        0  23.3                     0.672   32        1
3            1       89             66             23       94  28.1                     0.167   21        0
4            0      137             40             35      168  43.1                     2.288   33        1
PS C:\Users\Malavika>
```

2.0 Check for Missing Values

Next, we checked missing values in our dataset and found that there is no missing values.

Progress2.py

D: > Python > Progress2.py > ...

```python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file

# Importing dataset
path = 'D:\\Python\\diabetes.csv'
df = pd.read_csv(path)

# Display first few rows of DataFrame
print(df.head())

# Check for missing values
print(df.isnull())
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin    BMI  DiabetesPedigreeFunction    Age  Outcome
0          False    False          False          False    False  False                     False  False    False
1          False    False          False          False    False  False                     False  False    False
2          False    False          False          False    False  False                     False  False    False
3          False    False          False          False    False  False                     False  False    False
4          False    False          False          False    False  False                     False  False    False
..           ...      ...            ...            ...      ...    ...                       ...    ...      ...
763        False    False          False          False    False  False                     False  False    False
764        False    False          False          False    False  False                     False  False    False
765        False    False          False          False    False  False                     False  False    False
766        False    False          False          False    False  False                     False  False    False
767        False    False          False          False    False  False                     False  False    False

[768 rows x 9 columns]
PS C:\Users\Malavika>
```

3.0 Check missing values in each column

Then, we checked missing values in each column.

```
Progress2.py ×
D: > Python > Progress2.py > ...
13
14    # Check missing values in each columns
15    print(df.isnull().sum())
16
17
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

4.0 Check for duplicate rows

Then, we checked for any duplicate rows in the dataset and we found that there is no duplicate rows.

```
Progress2.py
D: > Python > Progress2.py > ...
# Check for duplicate rows
duplicate_rows = df[df.duplicated()]
print(duplicate_rows)
```

```
PS C:\Users\Malavika> & D:/Python/python.exe d:/Python/Progress2.py
Empty DataFrame
Columns: [Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age, Outcome]
Index: []
```

5.0 Data Formatting

Display data types in each column.

```
Progress2.py
D: > Python > Progress2.py > ...
# Display data types
print(df.dtypes)
```

```
PS C:\Users\Malavika> & D:/Python/python.exe d:/Python/Progress2.py
Pregnancies                   int64
Glucose                       int64
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                         float64
DiabetesPedigreeFunction    float64
Age                           int64
Outcome                       int64
dtype: object
```

6.0 Data Normalization

For data normalization, we use Z-score to measure how many standard deviations a data point is from the mean of a dataset.

Progress2.py

D: > Python > Progress2.py > ...

```
numeric_data = df.select_dtypes(include=['float64', 'int64'])

# Data Normalization (Z-Score)
scaler = StandardScaler()
df_normalized = pd.DataFrame(scaler.fit_transform(numeric_data), columns = df.columns)

print("Normalized DataFrame: ")
print(df_normalized)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Malavika> & D:/Python/python.exe d:/Python/Progress2.py
Normalized DataFrame:
     Pregnancies   Glucose  BloodPressure  SkinThickness   Insulin       BMI  DiabetesPedigreeFunction       Age   Outcome
0       0.639947  0.848324       0.149641       0.907270 -0.692891  0.204013                  0.468492  1.425995  1.365896
1      -0.844885 -1.123396      -0.160546       0.530902 -0.692891 -0.684422                 -0.365061 -0.190672 -0.732120
2       1.233880  1.943724      -0.263941      -1.288212 -0.692891 -1.103255                  0.604397 -0.105584  1.365896
3      -0.844885 -0.998208      -0.160546       0.154533  0.123302 -0.494043                 -0.920763 -1.041549 -0.732120
4      -1.141852  0.504055      -1.504687       0.907270  0.765836  1.409746                  5.484909 -0.020496  1.365896
..           ...       ...            ...            ...       ...       ...                       ...       ...       ...
763     1.827813 -0.622642       0.356432       1.722735  0.870031  0.115169                 -0.908682  2.532136 -0.732120
764    -0.547919  0.034598       0.046245       0.405445 -0.692891  0.610154                 -0.398282 -0.531023 -0.732120
765     0.342981  0.003301       0.149641       0.154533  0.279594 -0.735190                 -0.685193 -0.275760 -0.732120
766    -0.844885  0.159787      -0.470732      -1.288212 -0.692891 -0.240205                 -0.371101  1.170732  1.365896
767    -0.844885 -0.873019       0.046245       0.656358 -0.692891 -0.202129                 -0.473785 -0.871374 -0.732120

[768 rows x 9 columns]
PS C:\Users\Malavika>
```

7.0 Data Binning

We use age group for data bining.

```python
# Binning the 'age' columns
bins = [20, 30, 40, 50, 60, 70, 80, 90]
labels = ['20-30', '31-40', '41-50', '51-60', '61-70', '71-80', '81-90']
df['age_grouop'] = pd.cut(df['Age'], bins=bins, labels=labels, right=False)
print(df['age_grouop'])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Malavika> & D:/Python/python.exe d:/Python/Progress2.py
0      51-60
1      31-40
2      31-40
3      20-30
4      31-40
       ...
763    61-70
764    20-30
765    31-40
766    41-50
767    20-30
Name: age_grouop, Length: 768, dtype: category
Categories (7, object): ['20-30' < '31-40' < '41-50' < '51-60' < '61-70' < '71-80' < '81-90']
PS C:\Users\Malavika>
```