Vivekanand Education Society's
Institute of Technology
Approved by AICTE & Affiliated to University of Mumbai

Artificial Intelligence and Data Science Department

Machine Learning Lab/Even Sem 2023-23/Experiment 1b

Name :	Class/Roll No. :	Grade:
Dyotak Kachare	D11AD/26	

Title of Experiment :

Introduction to scikit learn, matplotlib, seaborn library

Objective of Experiment :

To introduce platforms such as Anaconda, COLAB suitable to Machine learning.

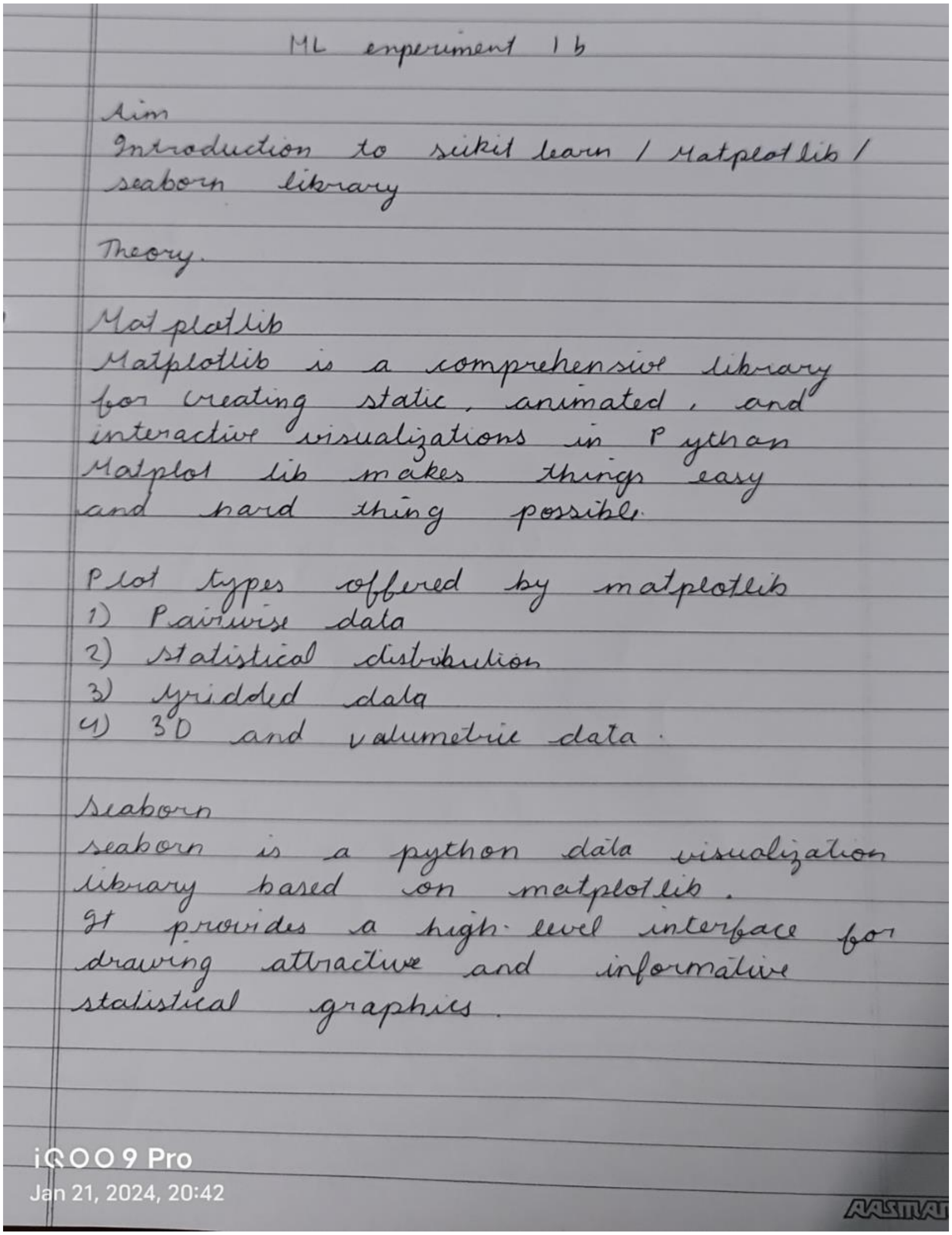
Outcome of Experiment :

Implement various Machine learning models

Problem Statement :

Introduction to scikit learn, matplotlib, seaborn library

Description / Theory :



ML enperiment 1b

Aim

Introduction to sikit learn / Matplotlib / seaborn library

Theory.

Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python
Matplot lib makes things easy and hard thing possible.

Plot types offered by matplotlib
1) Pairwise data
2) statistical distribution
3) gridded data
4) 3D and valumetric data.

Seaborn

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

1 Introduction to Scikit Learn

[1]:

```
# Importing required libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn
import seaborn as sns
```

[2]:

```
# Reading data

heart_disease = pd.read_csv('../data/cleaned/heart.csv')
heart_disease.head()
```

[2]:

```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak □
 ↪slope  \
0   52    1   0       125   212    0        1      168      0      1.0    □
 ↪2
1   53    1   0       140   203    1        0      155      1      3.1    □
 ↪0
2   70    1   0       145   174    0        1      125      1      2.6    □
 ↪0
3   61    1   0       148   203    0        1      161      0      0.0    □
 ↪2
4   62    0   0       138   294    1        1      106      0      1.9    □
 ↪1

   ca  thal  target
0   2     3       0
1   0     3       0
2   0     3       0
3   1     3       0
4   3     2       0
```

[3]:

```
# Create X (all the feature columns)
X = heart_disease.drop("target", axis=1)

# Create y (the target column)
y = heart_disease["target"]

# Check the head of the features DataFrame
X.head()
```

[3]:

```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak □
 ↪slope  \
0   52    1   0       125   212   0        1      168      0      1.0
  ↪2
1   53    1   0       140   203   1        0      155      1      3.1
  ↪0
2   70    1   0       145   174   0        1      125      1      2.6
  ↪0
3   61    1   0       148   203   0        1      161      0      0.0
  ↪2
4   62    0   0       138   294   1        1      106      0      1.9
  ↪1

   ca  thal
0   2     3
1   0     3
2   0     3
3   1     3
4   3     2
```

[4]:

```
# Check the head and the value counts of the labels
y.head(), y.value_counts()
```

[4]:

```
(0    0
 1    0
 2    0
 3    0
 4    0
 Name: target, dtype: int64,
 target
 1    526
 0    499
 Name: count, dtype: int64)
```

[5]:

```
# Split the data into training and test sets
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=0.25) # by
  ↪default train_test_split uses 25% of the data for the test set

X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

[5]:

```
((768, 13), (257, 13), (768,), (257,))
```

[6]:

```
from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier()
```

```
[7]: clf.get_params()
```

```
[7]: {'bootstrap': True,
      'ccp_alpha': 0.0,
      'class_weight': None,
      'criterion': 'gini',
      'max_depth': None,
      'max_features': 'sqrt',
      'max_leaf_nodes': None,
      'max_samples': None,
      'min_impurity_decrease': 0.0,
      'min_samples_leaf': 1,
      'min_samples_split': 2,
      'min_weight_fraction_leaf': 0.0,
      'n_estimators': 100,
      'n_jobs': None,
      'oob_score': False,
      'random_state': None,
      'verbose': 0,
      'warm_start': False}
```

```
[8]: clf.fit(X=X_train, y=y_train)
```

```
[8]: RandomForestClassifier()
```

```
[9]: y_preds = clf.predict(X=X_test)
     y_preds
```

```
[9]: array([1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1,
            0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0,
            1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0,
            0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
            1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1,
            0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0,
            0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0,
            1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0,
            1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1,
            1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1,
            0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1,
            1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0], dtype=int64)
```

```
[10]: from sklearn.metrics import classification_report, confusion_matrix,
        ↪accuracy_score

      # Create a classification report
      print(classification_report(y_test, y_preds))
```

```
              precision    recall  f1-score   support
```

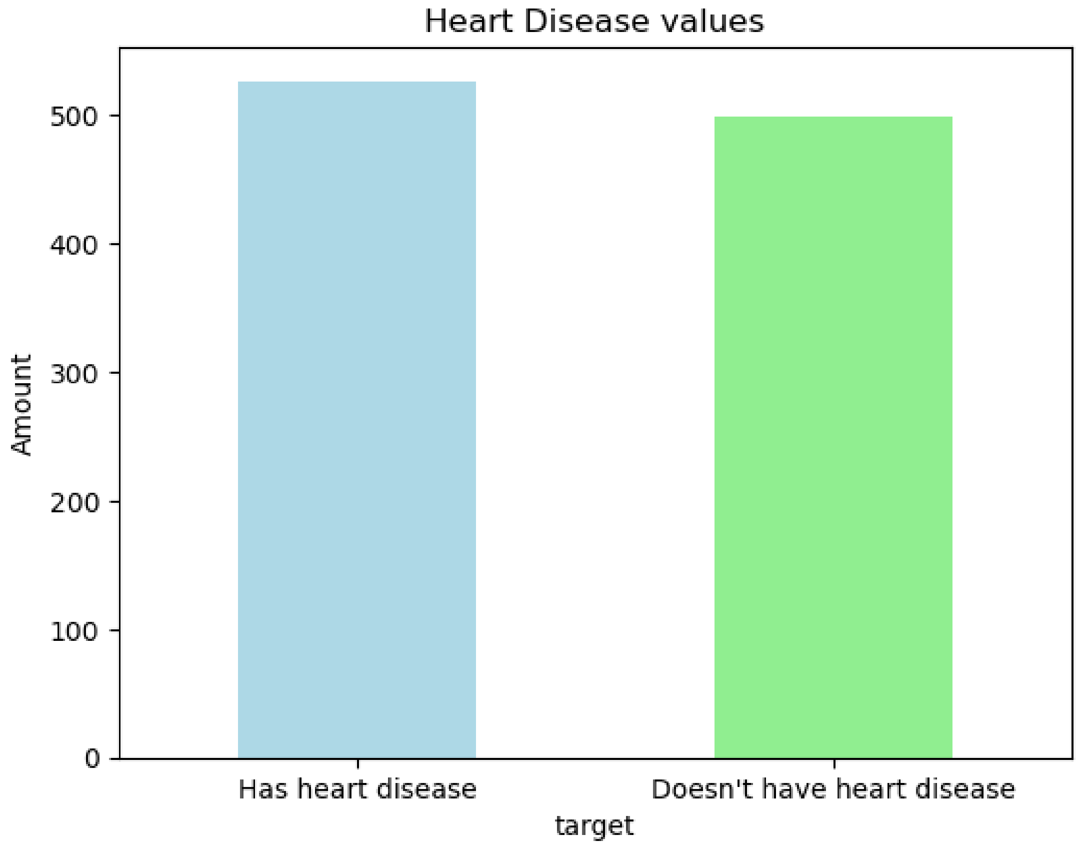
```
           0       1.00      1.00      1.00       124
           1       1.00      1.00      1.00       133

    accuracy                           1.00       257
   macro avg       1.00      1.00      1.00       257
weighted avg       1.00      1.00      1.00       257
```

[11]:
```
# Create a confusion matrix
conf_mat = confusion_matrix(y_test, y_preds)
conf_mat
```

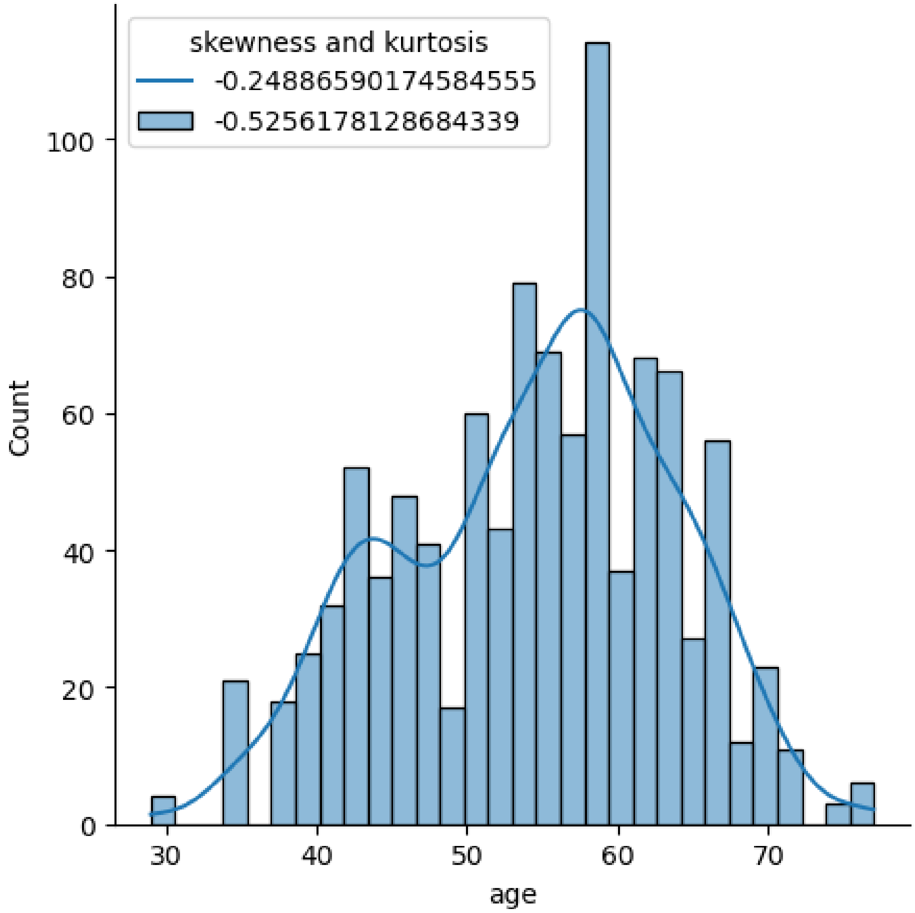
[11]:
```
array([[124,   0],
       [  0, 133]], dtype=int64)
```

[12]:
```
fig = heart_disease.target.value_counts().plot(kind = 'bar',
  ↪color=["lightblue", 'lightgreen'])
fig.set_xticklabels(labels=['Has heart disease', "Doesn't have heart
  ↪disease"], rotation=0);
plt.title("Heart Disease values")
plt.ylabel("Amount");
```



[13]:

```
#create a distribution plot with normal distribution curve
sns.displot( x = 'age', data = heart_disease, bins = 30, kde = True)
skewness=str(heart_disease["age"].skew())
kurtosis=str(heart_disease["age"].kurt())
plt.legend([skewness,kurtosis],title=("skewness and kurtosis"))
plt.show();
```



Result and Discussion :

Thus we have successfully used the python libraries.