Department: ENCS - Computer Systems Engineering - هندسة أنظمة الحاسوب

ENCS3340 – Artificial Intelligence

Give Life: Predict Blood Donations

1200388 – Mohammad Sabobeh

1200342 – Joud Hijaz

Instructor: Dr. Adnan Yahya

Section: 1

Date: 27 – 01 – 2024

Abstract

Our machine learning project predicts when people might donate blood, focusing on their donation history and total blood given. We use Decision Trees, Neural Networks, and Naïve Bias to make a helpful guide for blood banks to keep a steady supply for healthcare.

We will learn how to use machine learning tools (Python) to test these several algorithms for categorization tasks in this project. The data consists of several attributes;
ID, Recency (months), Frequency (times), Monetary (c.c. blood), Time (months),"whether he/she donated blood in March 2007.

The Dataset From : https://www.kaggle.com/code/mercury9181/blood-donations-prediction-using-ensemble-model/input

Full Code: https://drive.google.com/drive/folders/1X-jrFmgxj4Qbp9o8CinEF3FTxVhFKA3-?usp=sharing

Table of Contents

Implementation

Everything was implemented using python and mainly 'sklearn' library as it has lots of prediction models classifiers such as (Decision trees, Naive Bayes and ANN). Firstly we split the data into train and test provided with propotion approximatlly equal to 0.8 train and 0.2 test. And then we split each part into parameters and target variable renaming them to (x_data and y_data) we fit each model on the train (x_data, y_data) and finally ask it to classify y_test_data given x_test_data predciting our variable. We implemented each model giving it default parameters and then trying to optimize it trying a bunch of different hyper parameters using something called 'grid_search' to find the optimal parameter where the accuracy of the prediction is maximized. We implemented the interface using 'tkinter' library and plotted the diagrams and trees using 'matplotlib' library.

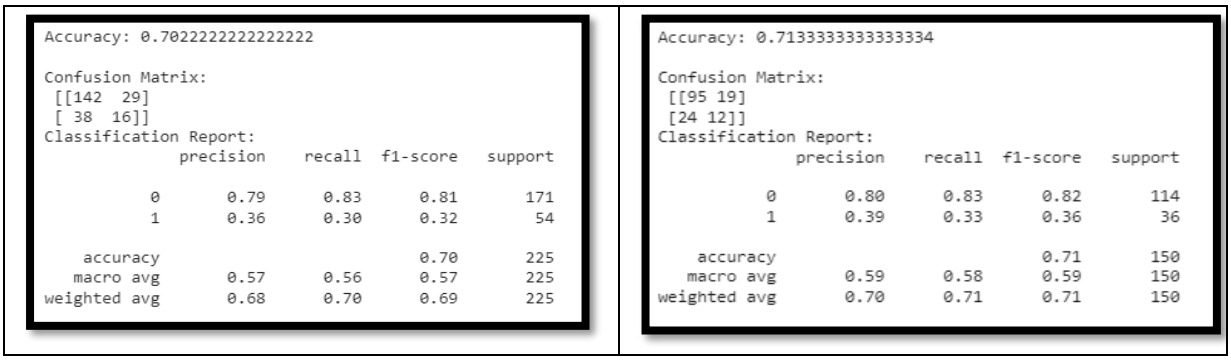
Model Algorithms

Before we model any algorithm, we use a scaler to make sure all the features have the same importance when the computer calculates distances and makes decisions. This can make our machine learning models work better and faster.

Decision Tree

A) Default Hyper parameters: (For the three test cases)

- We obtained the average weights of precision, recall and F-1 Score as shown in the figure below:



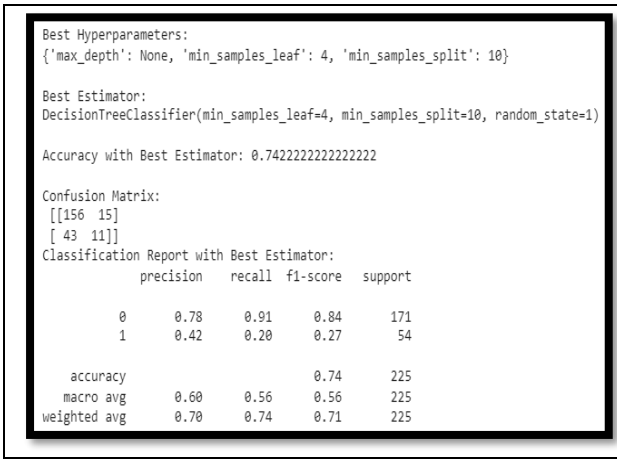
```
Accuracy: 0.7022222222222222

Confusion Matrix:
 [[142  29]
 [ 38  16]]
Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.83      0.81       171
           1       0.36      0.30      0.32        54

    accuracy                           0.70       225
   macro avg       0.57      0.56      0.57       225
weighted avg       0.68      0.70      0.69       225
```

```
Accuracy: 0.7133333333333334

Confusion Matrix:
 [[95 19]
 [24 12]]
Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.83      0.82       114
           1       0.39      0.33      0.36        36

    accuracy                           0.71       150
   macro avg       0.59      0.58      0.59       150
weighted avg       0.70      0.71      0.71       150
```

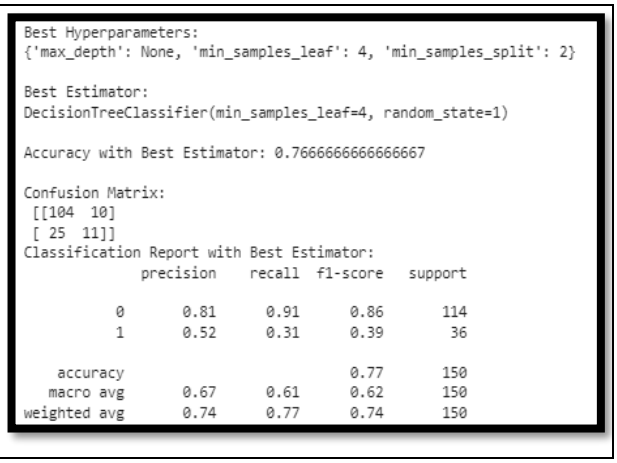
Case 1) (0.3, 0.7) **Case 2)** (0.2, 0.8)

B) New Hyper parameters: (For the first two test cases)

- We obtained the average weights of precision, recall and F-1 Score as shown in the figure below:



```
Best Hyperparameters:
{'max_depth': None, 'min_samples_leaf': 4, 'min_samples_split': 10}

Best Estimator:
DecisionTreeClassifier(min_samples_leaf=4, min_samples_split=10, random_state=1)

Accuracy with Best Estimator: 0.7422222222222222

Confusion Matrix:
 [[156  15]
 [ 43  11]]
Classification Report with Best Estimator:
              precision    recall  f1-score   support

           0       0.78      0.91      0.84       171
           1       0.42      0.20      0.27        54

    accuracy                           0.74       225
   macro avg       0.60      0.56      0.56       225
weighted avg       0.70      0.74      0.71       225
```



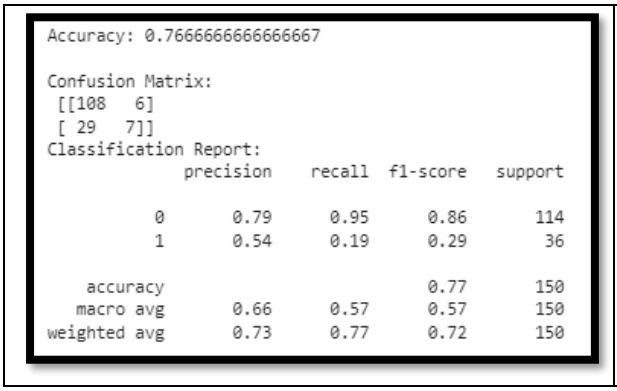
```
Best Hyperparameters:
{'max_depth': None, 'min_samples_leaf': 4, 'min_samples_split': 2}

Best Estimator:
DecisionTreeClassifier(min_samples_leaf=4, random_state=1)

Accuracy with Best Estimator: 0.7666666666666667

Confusion Matrix:
 [[104  10]
 [ 25  11]]
Classification Report with Best Estimator:
              precision    recall  f1-score   support

           0       0.81      0.91      0.86       114
           1       0.52      0.31      0.39        36

    accuracy                           0.77       150
   macro avg       0.67      0.61      0.62       150
weighted avg       0.74      0.77      0.74       150
```

Case 1) (0.3, 0.7) **Case 2)** (0.2, 0.8)

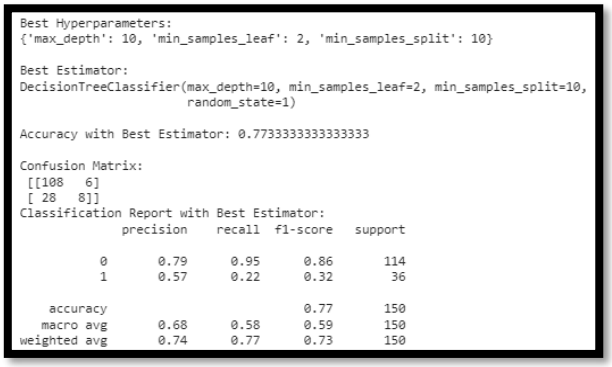
We can notice from the figure below that the accuracy, precision, recall, F-Measure and the confusion matrix changed, when the train data is bigger the accuracy is better though the model is better.

As we can see, the accuracy increased and the other values changed.

C) When we remove a specific feature “Time (months)” on Test Case 2 (0.2, 0.8)



```
Accuracy: 0.7666666666666667

Confusion Matrix:
 [[108   6]
 [ 29   7]]
Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.95      0.86       114
           1       0.54      0.19      0.29        36

    accuracy                           0.77       150
   macro avg       0.66      0.57      0.57       150
weighted avg       0.73      0.77      0.72       150
```



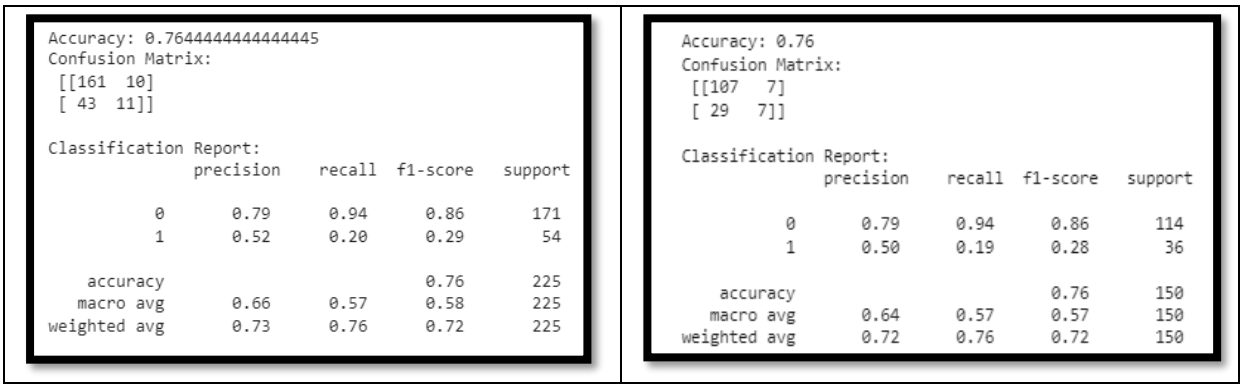
```
Best Hyperparameters:
{'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 10}

Best Estimator:
DecisionTreeClassifier(max_depth=10, min_samples_leaf=2, min_samples_split=10,
                       random_state=1)

Accuracy with Best Estimator: 0.7733333333333333

Confusion Matrix:
 [[108   6]
 [ 28   8]]
Classification Report with Best Estimator:
              precision    recall  f1-score   support

           0       0.79      0.95      0.86       114
           1       0.57      0.22      0.32        36

    accuracy                           0.77       150
   macro avg       0.68      0.58      0.59       150
weighted avg       0.74      0.77      0.73       150
```

Default new HyperParam

Naive Bayes

A) Default Hyper parameters: (For first two test cases)

- We obtained the average weights of precision, recall and F-1 Score as shown in the figure below:



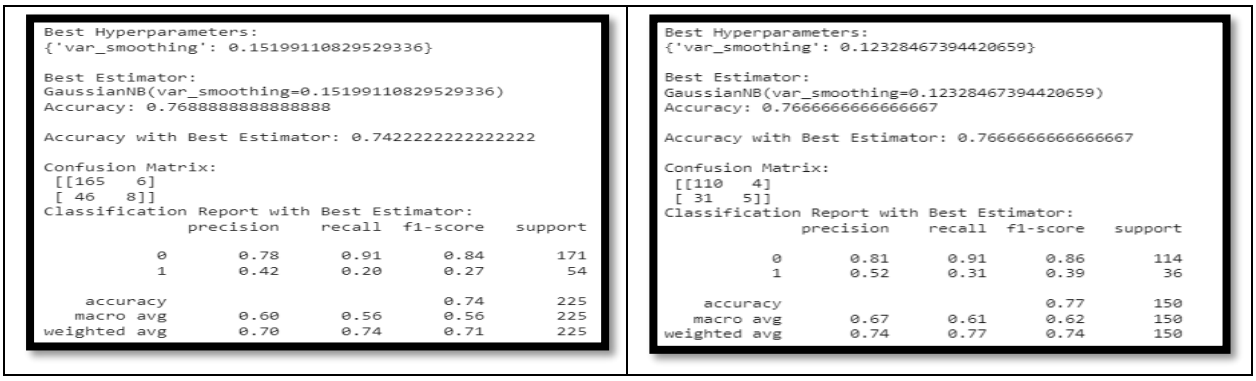
```
Accuracy: 0.7644444444444445
Confusion Matrix:
 [[161  10]
 [ 43  11]]

Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.94      0.86       171
           1       0.52      0.20      0.29        54

    accuracy                           0.76       225
   macro avg       0.66      0.57      0.58       225
weighted avg       0.73      0.76      0.72       225
```

```
Accuracy: 0.76
Confusion Matrix:
 [[107   7]
 [ 29   7]]

Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.94      0.86       114
           1       0.50      0.19      0.28        36

    accuracy                           0.76       150
   macro avg       0.64      0.57      0.57       150
weighted avg       0.72      0.76      0.72       150
```

Case 1) (0.3, 0.7) **Case 2)** (0.2, 0.8)

B) New Hyper parameters: (For the first two test cases)

- We obtained the average weights of precision, recall and F-1 Score as shown in the figure below:



```
Best Hyperparameters:
{'var_smoothing': 0.15199110829529336}

Best Estimator:
GaussianNB(var_smoothing=0.15199110829529336)
Accuracy: 0.7688888888888888

Accuracy with Best Estimator: 0.7422222222222222

Confusion Matrix:
 [[165   6]
 [ 46   8]]
Classification Report with Best Estimator:
              precision    recall  f1-score   support

           0       0.78      0.91      0.84       171
           1       0.42      0.20      0.27        54

    accuracy                           0.74       225
   macro avg       0.60      0.56      0.56       225
weighted avg       0.70      0.74      0.71       225
```

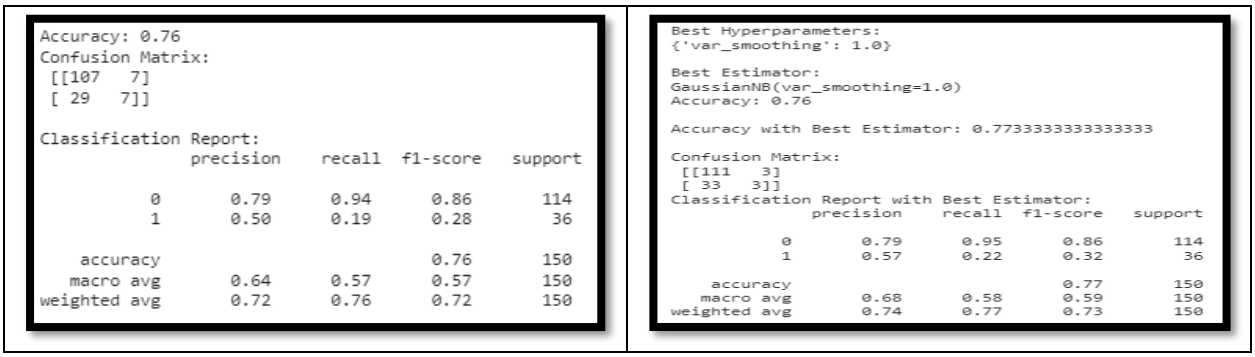
```
Best Hyperparameters:
{'var_smoothing': 0.12328467394420659}

Best Estimator:
GaussianNB(var_smoothing=0.12328467394420659)
Accuracy: 0.7666666666666667

Accuracy with Best Estimator: 0.7666666666666667

Confusion Matrix:
 [[110   4]
 [ 31   5]]
Classification Report with Best Estimator:
              precision    recall  f1-score   support

           0       0.81      0.91      0.86       114
           1       0.52      0.31      0.39        36

    accuracy                           0.77       150
   macro avg       0.67      0.61      0.62       150
weighted avg       0.74      0.77      0.74       150
```

Case 1) (0.3, 0.7) **Case 2)** (0.2, 0.8)

As we can see, the accuracy increased and the other values changed.

C) When we remove a specific feature “Time (months)” using test case 2

Test Case 2) (0.2, 0.8)



```
Accuracy: 0.76
Confusion Matrix:
 [[107   7]
 [ 29   7]]

Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.94      0.86       114
           1       0.50      0.19      0.28        36

    accuracy                           0.76       150
   macro avg       0.64      0.57      0.57       150
weighted avg       0.72      0.76      0.72       150
```

```
Best Hyperparameters:
{'var_smoothing': 1.0}

Best Estimator:
GaussianNB(var_smoothing=1.0)
Accuracy: 0.76

Accuracy with Best Estimator: 0.7733333333333333

Confusion Matrix:
 [[111   3]
 [ 33   3]]
Classification Report with Best Estimator:
              precision    recall  f1-score   support

           0       0.79      0.95      0.86       114
           1       0.57      0.22      0.32        36

    accuracy                           0.77       150
   macro avg       0.68      0.58      0.59       150
weighted avg       0.74      0.77      0.73       150
```

Default new Parameters

Artificial Neural Networks

A) Default Hyper parameters: (For first two test cases)

- We obtained the average weights of precision, recall and F-1 Score and the confusion matrix as shown in the figures below:

```
Accuracy: 0.7777777777777778
Confusion Matrix:
 [[167   4]
 [ 46   8]]
Classification Report:
              precision    recall  f1-score   support

           0       0.78      0.98      0.87       171
           1       0.67      0.15      0.24        54

    accuracy                           0.78       225
   macro avg       0.73      0.56      0.56       225
weighted avg       0.76      0.78      0.72       225
```

Case 1) (0.3, 0.7)

```
Accuracy: 0.7866666666666666
Confusion Matrix:
 [[111   3]
 [ 29   7]]
Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.97      0.87       114
           1       0.70      0.19      0.30        36

    accuracy                           0.79       150
   macro avg       0.75      0.58      0.59       150
weighted avg       0.77      0.79      0.74       150
```

Case 2) (0.2, 0.8)

B) New Hyper parameters: (For the first two test cases)

```
Accuracy: 0.8133333333333334
Confusion Matrix:
 [[164   7]
 [ 35  19]]
Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.96      0.89       171
           1       0.73      0.35      0.48        54

    accuracy                           0.81       225
   macro avg       0.78      0.66      0.68       225
weighted avg       0.80      0.81      0.79       225
```

Case 1) (0.3, 0.7)

```
Accuracy: 0.7933333333333333
Confusion Matrix:
 [[106   8]
 [ 23  13]]
Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.93      0.87       114
           1       0.62      0.36      0.46        36

    accuracy                           0.79       150
   macro avg       0.72      0.65      0.66       150
weighted avg       0.77      0.79      0.77       150
```

Case 2) (0.2, 0.8)

As we can see, the accuracy increased and the other values changed.

A) When we remove a specific feature “Time (months)” using test case 2

```
Accuracy: 0.76
Confusion Matrix:
 [[110   4]
 [ 32   4]]
Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.96      0.86       114
           1       0.50      0.11      0.18        36

    accuracy                           0.76       150
   macro avg       0.64      0.54      0.52       150
weighted avg       0.71      0.76      0.70       150
```

Default

```
Accuracy: 0.7733333333333333
Confusion Matrix:
 [[112   2]
 [ 32   4]]
Classification Report:
              precision    recall  f1-score   support

           0       0.78      0.98      0.87       114
           1       0.67      0.11      0.19        36

    accuracy                           0.77       150
   macro avg       0.72      0.55      0.53       150
weighted avg       0.75      0.77      0.71       150
```

new Parameters

Comparison between Models

Confusion matrices for each model (for the updated hyper parameters): (Case1, Case2)

	Model	Accuracy	Precision	Recall	F1-Score
0	Decision Tree	0.742222	0.423077	0.203704	0.275000
1	Naive Bayes	0.768889	0.571429	0.148148	0.235294
2	Artificial Nerual Network	0.813333	0.730769	0.351852	0.475000

	Model	Accuracy	Precision	Recall	F1-Score
0	Decision Tree	0.766667	0.523810	0.305556	0.385965
1	Naive Bayes	0.766667	0.555556	0.138889	0.222222
2	Artificial Nerual Network	0.793333	0.619048	0.361111	0.456140

Conclusion

Machine learning is very important to ease the job of humans by teaching the machines how to test the data and make correct decisions. In this project, we used PYTHON language to test the data set, and we used 3 different algorithms which are: Decision tree, Naïve Bayes and Artificial Neural Network. Each algorithm displayed different success rate (accuracy), all of them had a range of accuracy between 70% in Decision tree and 81% in Naïve Bayes. Each algorithm performs using a certain procedure and displays its results. Applying filters, changing the hyper parameters will affect the results we obtain, and that was shown for each algorithm.