Machine-Learning Experiments

#we use random_state = 42 , To avoid from randomnes that can affect our expirements

Model Decision Tree:

```
1. DecisionTreeClassifier(criterion = "log_loss",max_depth=
20,min_samples_split= 5 ,min_samples_leaf= 6 ,random_state = 42)
```

Criterion = “log loss” (default = gini), max_depth = 20 (default = None) min_sample split = 5 (default = 2) , min sample leaf = 6 (default = 1).

The Results are:

Accuracy Score : 0.9766081871345029
Precision Score : 0.9727272727272728
Recall Score: 0.9907407407407407
F1-Score: 0.981651376146789
Confuision Matrix: [[60 3]
[1 107]]

```
2. DecisionTreeClassifier(criterion="entropy",random_state=42)
```

Criterion = “entropy” (default = gini).

The Results are:

Accuracy Score : 0.9649122807017544
Confuision Matrix:
[[59 4]
[2 106]]
Precision Score : 0.9636363636363636
Recall Score: 0.9814814814814815
F1-Score: 0.9724770642201835

```
3.DecisionTreeClassifier(criterion="entropy",splitter="random",max_depth=1
5,max_features=5,random_state=42)
```

Criterion = entropy(default = gini) , spliter = random(default = best) ,max depth = 15 (default = None) , max features = 5 (default = None).

The Results are:

Accuracy Score : 0.9181286549707602
Confuision Matrix:
[[58 5]
[9 99]]
Precision Score : 0.9519230769230769
Recall Score: 0.9166666666666666
F1-Score: 0.9339622641509434

```
4.DecisionTreeClassifier(criterion="entropy",splitter="random",max_depth=40,max_features=40,max_leaf_nodes=20,min_samples_split=5,random_state=42)
```

Criterion = entropy(default = gini) , spliter = random(default = best), max depth = 40 (default = None) , max features = 40 (default = None) , max leaf node = 20 (default = None) ,min sample split = 5 (default = 1).

The Results are:

Accuracy Score : 0.9590643274853801
Confuision Matrix:
[[60 3]
 [4 104]]
Precision Score : 0.9719626168224299
Recall Score: 0.9629629629629629
F1-Score: 0.9674418604651163

```
5. DecisionTreeClassifier(criterion = "log_loss",splitter="random",max_depth=8,max_features=5,min_samples_leaf=3,min_impurity_decrease=0.1,random_state=42)
```

Criterion = log loss (default = gini) , spliter = random (default = best),max depth = 8 (default = None) , max features =5 (default = None) , min sample leaf = 3 (default = 1) , min impurity decrease = 0.1 (default = 0.0).

The Results are:

Accuracy Score : 0.8654970760233918
Confuision Matrix:
[[59 4]
 [19 89]]
Precision Score : 0.956989247311828
Recall Score: 0.8240740740740741
F1-Score: 0.8855721393034826

Conclusion: **because we choose the Recall Score as our decisive metric , expirement #1 give the best result.**

Model Random Forest:

1.
```
RandomForestClassifier(n_estimators=1000 , criterion="entropy",max_features=None,random_state=42)
```

N_estimators = 1000(default = 100) , critetion = entropy (default = gini) , max features = None (default = sqrt).

The Results are:

Accuracy Score : 0.9590643274853801

Confuision Matrix:

[[59 4]

 [3 105]]

Precision Score : 0.963302752293578

Recall Score: 0.9722222222222222

F1-Score: 0.967741935483871

2.
```
RandomForestClassifier(n_estimators=1000 , criterion="log_loss",max_features="log2" , min_samples_split= 4 , min_samples_leaf= 10 ,min_weight_fraction_leaf=0.2 , max_leaf_nodes=100,random_state=42)
```

N_estimators = 1000(default = 100) , critetion = log loss (default = gini) , max features = log2 (default = sqrt) , min sample split = 4 (default = 2) , min sample leaf = 10 (default = 1) , min weight fraction leaf = 0.2 (default = 0.0) , max leaf nodes = 100 (default = none) .

The Results are:

Accuracy Score : 0.9707602339181286

Confuision Matrix:

[[58 5]

 [0 108]]

Precision Score : 0.9557522123893806

Recall Score: 1.0

F1-Score: 0.9773755656108597

3.
```
RandomForestClassifier(n_estimators=2000 , criterion="log_loss", min_samples_split= 3 ,min_impurity_decrease=0.1 , min_samples_leaf= 5 ,min_weight_fraction_leaf=0.5 , max_leaf_nodes=50,random_state=42)
```

N_estimators = 2000(default = 100) , critetion = log loss (default = gini) , min sample split = 3 (default = 2) , min sample leaf = 5 (default = 1),min impurity decrease = 0.1 (default = 0.0) , min weight fraction leaf = 0.5 (default = 0.0) , max leaf nodes = 50 (default = none) .

The Results are:

Accuracy Score : 0.9532163742690059

Confuision Matrix:

[[60 3]

 [5 103]]

Precision Score : 0.9716981132075472

Recall Score: 0.9537037037037037

F1-Score: 0.9626168224299065

4
```
RandomForestClassifier(n_estimators=2000, min_samples_leaf= 10  ,
max_leaf_nodes=125,random_state=42)
```

N_estimators = 2000(default = 100), min sample leaf = 10 (default = 1), max leaf nodes = 125 (default = None).

The Results are:

Accuracy Score : 0.9707602339181286
Confuision Matrix:
[[59 4]
 [1 107]]
Precision Score : 0.963963963963964
Recall Score: 0.9907407407407407
F1-Score: 0.9771689497716894

5.
```
RandomForestClassifier(n_estimators=4000,min_weight_fraction_leaf=0.3 ,
max_features="log2"  , min_samples_leaf= 12  ,
max_leaf_nodes=125,random_state=42)
```

N_estimators = 4000(default = 100), min sample leaf = 12 (default = 1), max_feature = log2(default = sqrt) , min weight fraction leaf = 0.3 (default = 0.0) , max leaf nodes = 125 (default = None).

The Results are:

Accuracy Score : 0.9590643274853801
Confuision Matrix:
[[57 6]
 [1 107]]
Precision Score : 0.9469026548672567
Recall Score: 0.9907407407407407
F1-Score: 0.9683257918552036

Conclusion: **because we choose the Recall Score as our decisive metric , expirement #2 give the best result.**

Model AdaBoost :

1. `AdaBoostClassifier(n_estimators=100, learning_rate=2.0 , algorithm='SAMME',random_state=42)`

n-estimators = 100 (default = 50) , learning_rate = 2.0 (default = 1.0) algorithm = ‘SAMME’ (default = ‘SAMME.R’)

The Results are:

Accuracy Score : 0.9064327485380117
Confuision Matrix:
[[54 9]
 [7 101]]
Precision Score : 0.9181818181818182
Recall Score: 0.9351851851851852
F1-Score: 0.926605504587156

2. `AdaBoostClassifier(n_estimators=250, learning_rate=0.5 , algorithm='SAMME',random_state=42)`

n-estimators = 250 (default = 50) , learning_rate = 0.5 (default = 1.0) algorithm = ‘SAMME’ (default = ‘SAMME.R’)

The Results are:

Accuracy Score : 0.9766081871345029
Confuision Matrix:
[[62 1]
 [3 105]]
Precision Score : 0.9905660377358491
Recall Score: 0.9722222222222222
F1-Score: 0.9813084112149533

3. `AdaBoostClassifier(n_estimators=75 , learning_rate=0.5,random_state=42)`

n-estimators = 75 (default = 50) , learning_rate = 0.5 (default = 1.0)

The Result are:

Accuracy Score : 0.9590643274853801
Confuision Matrix:
[[60 3]
 [4 104]]
Precision Score : 0.9719626168224299
Recall Score: 0.9629629629629629
F1-Score: 0.9674418604651163

4.`AdaBoostClassifier(n_estimators=52,algorithm='SAMME',learning_rate=0.2,random_state=42)`

n-estimators = 52 (default = 50) , learning_rate = 0.2 (default = 1.0) , algorithm = SAMME (default = SAMME.R).

The Result are:

Accuracy Score : 0.9590643274853801
Confuision Matrix:
[[58 5]
 [2 106]]
Precision Score : 0.954954954954955
Recall Score: 0.9814814814814815
F1-Score: 0.9680365296803652

5. `AdaBoostClassifier(n_estimators=52,learning_rate=1.5,random_state=42)`

n-estimators = 52 (default = 50) , learning_rate = 1.5 (default = 1.0).
The Result are:
Accuracy Score : 0.9707602339181286
Confuision Matrix:
[[61 2]
 [3 105]]
Precision Score : 0.9813084112149533
Recall Score: 0.9722222222222222
F1-Score: 0.9767441860465116

Conclusion: **because we choose the Recall Score as our decisive metric , expirement #4 give the best result.**