Classification Assignment

1.) Identify your problem statement

- **Stage 1:** Number based which falls under Machine Learning
- **Stage 2:** Learning Selection would be Supervised learning both input and output is present.
- **Stage 3:** Since the output is in character's so it will falls under Classification.

2) Tell basic info about the dataset (Total number of rows, columns)

- **Total no. of Rows: 399**
- **Total no. of Columns: 25**

3) Mention the pre-processing method if you're doing any (like converting string to number – nominal data)

- Data's are in nominal data first we have to convert categorical column -> Nominal data one hot encoding.

4) Develop a good model with good evaluation metric. You can use any machine learning algorithm; you can create many models. Finally, you have to come up with final model.

1. Logistic Regression Model Performance:

[29]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(Y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter{}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'penalty': 'l2', 'solver': 'lbfgs'}: 0.9916844900066377
```

[27]:
```
from sklearn.metrics import classification_report
clf_report=classification_report(Y_test,grid_predictions)
```

[30]:
```
print("The consusion matrix:\n",cm)
```

```
The consusion matrix:
 [[45  0]
 [ 1 74]]
```

[31]:
```
print("The report:\n",clf_report)
```

```
The report:
              precision    recall  f1-score   support

       False       0.98      1.00      0.99        45
        True       1.00      0.99      0.99        75

    accuracy                           0.99       120
   macro avg       0.99      0.99      0.99       120
weighted avg       0.99      0.99      0.99       120
```

[32]:
```
from sklearn.metrics import roc_auc_score
roc_auc_score(Y_test,grid.predict_proba(X_test)[:,1])
```

[32]:
```
1.0
```

2. **Random Forest Classification**:

[19]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(Y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter{}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'criterion': 'log_loss', 'max_features': 'sqrt', 'n_estimators': 100}: 0.9833333333333333
```

[20]:
```
print("The consusion matrix:\n",cm)
```

```
The consusion matrix:
 [[44  1]
 [ 1 74]]
```

[21]:
```
print("The report:\n",clf_report)
```

```
The report:
              precision    recall  f1-score   support

       False       0.98      0.98      0.98        45
        True       0.99      0.99      0.99        75

    accuracy                           0.98       120
   macro avg       0.98      0.98      0.98       120
weighted avg       0.98      0.98      0.98       120
```

[22]:
```
from sklearn.metrics import roc_auc_score
roc_auc_score(Y_test,grid.predict_proba(X_test)[:,1])
```

[22]:
```
0.9997037037037036
```

3. **Decision Tree Classification**:

[15]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(Y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter{}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'criterion': 'entropy', 'max_features': 'log2', 'splitter': 'random'}: 0.9834018801410106
```

[16]:
```
print("The consusion matrix:\n",cm)
```

```
The consusion matrix:
 [[45  0]
 [ 2 73]]
```

[17]:
```
print("The report:\n",clf_report)
```

```
The report:
              precision    recall  f1-score   support

       False       0.96      1.00      0.98        45
        True       1.00      0.97      0.99        75

    accuracy                           0.98       120
   macro avg       0.98      0.99      0.98       120
weighted avg       0.98      0.98      0.98       120
```

[18]:
```
from sklearn.metrics import roc_auc_score
roc_auc_score(Y_test,grid.predict_proba(X_test)[:,1])
```

[18]:
```
0.9866666666666667
```

4. **SVM Classification**:

```
[15]: from sklearn.metrics import f1_score
      f1_macro=f1_score(Y_test,grid_predictions,average='weighted')
      print("The f1_macro value for best parameter{}:".format(grid.best_params_),f1_macro)

      The f1_macro value for best parameter{'criterion': 'entropy', 'max_features': 'log2', 'splitter': 'random'}: 0.9834018801410106

[16]: print("The consusion matrix:\n",cm)

      The consusion matrix:
       [[45  0]
       [ 2 73]]

[17]: print("The report:\n",clf_report)

      The report:
                    precision    recall  f1-score   support

             False       0.96      1.00      0.98        45
              True       1.00      0.97      0.99        75

          accuracy                           0.98       120
         macro avg       0.98      0.99      0.98       120
      weighted avg       0.98      0.98      0.98       120

[18]: from sklearn.metrics import roc_auc_score
      roc_auc_score(Y_test,grid.predict_proba(X_test)[:,1])

[18]: 0.9866666666666667
```

Out of all the models the best accuracy model is **Logistic Regression value is 0.99** when compared to other models