

Project Development Phase
Model Performance Test

Date - 19 November 2022
Team ID - PNT2022TMID53422
Project Name - Statistical Machine Learning Approaches To Liver Disease Prediction
Maximum Marks - 10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot																														
1.	Metrics	<p>Classification Model:</p> <p>Confusion Matrix - [[130 11] [43 9]]</p> <p>Accuracy Score- 72%</p> <p>Classification Report -</p> <table><tr><td></td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr><tr><td>0</td><td>0.75</td><td>0.92</td><td>0.83</td><td>141</td></tr><tr><td>1</td><td>0.45</td><td>0.17</td><td>0.25</td><td>52</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.72</td><td>193</td></tr><tr><td>macro avg</td><td>0.60</td><td>0.55</td><td>0.54</td><td>193</td></tr><tr><td>weighted avg</td><td>0.67</td><td>0.72</td><td>0.67</td><td>193</td></tr></table>		precision	recall	f1-score	support	0	0.75	0.92	0.83	141	1	0.45	0.17	0.25	52	accuracy			0.72	193	macro avg	0.60	0.55	0.54	193	weighted avg	0.67	0.72	0.67	193	Attached below
	precision	recall	f1-score	support																													
0	0.75	0.92	0.83	141																													
1	0.45	0.17	0.25	52																													
accuracy			0.72	193																													
macro avg	0.60	0.55	0.54	193																													
weighted avg	0.67	0.72	0.67	193																													

2.	Tune the Model	Hyperparameter Tuning - GridSearchCV Validation Method - GridSearchCV, XGBClassifier Accuracy after Hyperparameter Tuning- 76%	Attached below
----	----------------	--	----------------

Screenshots:

Confusion Matrix and Classification Report:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression(random_state=50)
model.fit(X_train,y_train)
pred_y = model.predict(X_test)
from sklearn.metrics import classification_report
x = accuracy_score(y_test, pred_y)
print("Logistic Regression's Accuracy is: ", x*100)
print(classification_report(y_test,pred_y))
print(confusion_matrix(y_test,pred_y))
```

```
Logistic Regression's Accuracy is: 72.02072538860104
precision    recall  f1-score   support

      0       0.75      0.92      0.83       141
      1       0.45      0.17      0.25        52

 accuracy          0.72       193
 macro avg          0.60       193
weighted avg          0.67       193

[[130  11]
 [ 43   9]]
```

Accuracy Before Hyperparameter Tuning:

```
Logistic Regression's Accuracy is: 72.02072538860104
```

Hyperparameter Tuning:

```
from sklearn.ensemble import RandomForestClassifier
rf_clf = GridSearchCV(RandomForestClassifier(), {'n_estimators': [1, 5, 10, 20, 30, 40, 50, 60, 1000, 5000, 6000]}, cv=10, return_train_score=False)
rf_clf.fit(X_train, y_train)
j=rf_clf.predict(X_test)
f=accuracy_score(y_test,j)
rf_clf.cv_results_

{'mean_fit_time': array([5.74145317e-01, 1.35345127e-02, 2.10136338e-02, 5.11044002e-02,
 3.91498566e-02, 1.12613206e-01, 8.8992451e-02, 1.07155395e-01,
 1.70091519e+00, 8.54370084e+00, 1.02711471e+01]),
'mean_score_time': array([0.00236555, 0.00373943, 0.00393867, 0.00715004, 0.00921438,
 0.0093846 , 0.00751764, 0.00667506, 0.11005119, 0.56951221,
 0.74176738]),
'mean_test_score': array([0.63076023, 0.66410256, 0.68205120, 0.68461538, 0.69487179,
 0.67179487, 0.66410256, 0.66923077, 0.66923077, 0.66923077,
 0.67435897]),
'param_n_estimators': masked_array(data=[1, 5, 10, 20, 30, 40, 50, 60, 1000, 5000, 6000],
 mask=[False, False, False, False, False, False, False, False,
       False, False, False],
 fill_value='?',
 dtype=object),
'params': [{'n_estimators': 1},
 {'n_estimators': 5},
 {'n_estimators': 10},
 {'n_estimators': 20},
 {'n_estimators': 30},
 ...,
 {'n_estimators': 30},
 {'n_estimators': 40},
 {'n_estimators': 50},
 {'n_estimators': 60},
 {'n_estimators': 1000},
 {'n_estimators': 5000},
 {'n_estimators': 6000}],
'rank_test_score': array([11, 10, 3, 2, 1, 5, 9, 6, 6, 6, 4], dtype=int32),
'split0_test_score': array([0.51282051, 0.58974359, 0.61538462, 0.58974359, 0.58974359,
 0.56410256, 0.58974359, 0.56410256, 0.58974359, 0.58974359,
 0.58974359]),
'split1_test_score': array([0.66666667, 0.58974359, 0.53846154, 0.61538462, 0.71794872,
 0.61538462, 0.58974359, 0.58974359, 0.58974359, 0.58974359,
 0.58974359]),
'split2_test_score': array([0.64102564, 0.74358974, 0.64102564, 0.69230769, 0.66666667,
 0.64102564, 0.69230769, 0.71794872, 0.71794872, 0.71794872,
 0.71794872]),
'split3_test_score': array([0.64102564, 0.66666667, 0.76923077, 0.74358974, 0.84615385,
 0.69230769, 0.76923077, 0.71794872, 0.71794872, 0.71794872,
 0.71794872]),
'split4_test_score': array([0.56410256, 0.66666667, 0.71794872, 0.74358974, 0.71794872,
 0.71794872, 0.66666667, 0.69230769, 0.69230769, 0.69230769,
 0.69230769]),
'split5_test_score': array([0.71794872, 0.64102564, 0.69230769, 0.71794872, 0.58974359,
 0.66666667, 0.64102564, 0.58974359, 0.58974359, 0.58974359,
 0.61538462]),
'split6_test_score': array([0.53846154, 0.56410256, 0.61538462, 0.58974359, 0.51282051,
 0.56410256, 0.53846154, 0.61538462, 0.51282051, 0.51282051,
 0.53846154]),
'split7_test_score': array([0.71794872, 0.84615385, 0.74358974, 0.64102564, 0.79487179,
 0.66666667, 0.69230769, 0.71794872, 0.69230769, 0.69230769,
```

Accuracy After Hyperparameter Tuning :

```
'split7_test_score': array([0.71794872, 0.84615385, 0.74358974, 0.64102564, 0.79487179,  
0.66666667, 0.69230769, 0.71794872, 0.69230769, 0.69230769,  
0.69230769]),  
'split8_test_score': array([0.64102564, 0.71794872, 0.76923077, 0.76923077, 0.76923077,  
0.79487179, 0.79487179, 0.71794872, 0.82051282, 0.82051282,  
0.82051282]),  
'split9_test_score': array([0.66666667, 0.61538462, 0.71794872, 0.74358974, 0.74358974,  
0.79487179, 0.66666667, 0.76923077, 0.76923077, 0.76923077,  
0.76923077]),  
'std_fit_time': array([0.00157073, 0.00228115, 0.00260054, 0.01041249, 0.0059753 ,  
0.01008318, 0.00368796, 0.0040749 , 0.01746637, 0.26887338,
```

```
[ ] f*100
```

```
75.64766839378238
```