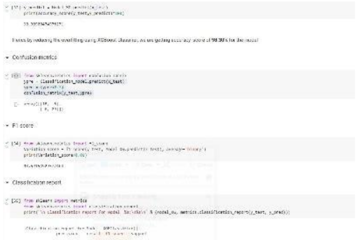
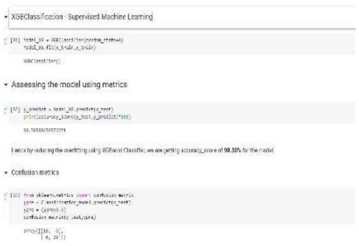
Project Development Phase
Model Performance Test

Date	17th November 2022
Team ID	PNT2022TMID28255
Project Name	Project – Detecting Parkinson’s Disease using Machine Learning
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	**Classification Model:** Confusion Matrix, F1 Score, Accuracy Score & Classification Report	
2.	Tune the Model	Data mining - XGBoost Classifier	

```
[37] y_predict = Model_XG.predict(x_test)
     print(accuracy_score(y_test,y_predict)*100)

     98.30508474576271
```

Hence by reducing the overfitting using XGBoost Classifier, we are getting accuracy score of 98.30% for the model

Confusion metrics

```
[33] from sklearn.metrics import confusion_matrix
     ypre = classification_model.predict(x_test)
     ypre = (ypre>0.5)
     confusion_matrix(y_test,ypre)

     array([[18, 5],
            [ 6, 29]])
```

F1 score

```
[34] from sklearn.metrics import f1_score
     variation_score = f1_score(y_test, model_xg.predict(x_test), average='binary')
     print(variation_score*100)

     [illegible]
```

Classification report

```
[36] from sklearn import metrics
     from sklearn.metrics import classification_report
     print('\n classification report for model %s:\n%s\n' % (model_xg, metrics.classification_report(y_test, y_pred)))

     Classification report for model : XGBClassifier():
                   precision    recall  f1-score   support
```

XGBClassification - Supervised Machine Learning

```
[31] model_XG = XGBClassifier(random_state=0)
     model_XG.fit(x_train,y_train)

     XGBClassifier()
```

Assessing the model using metrics

```
[32] y_predict = Model_XG.predict(x_test)
     print(accuracy_score(y_test,y_predict)*100)

     98.30508474576271
```

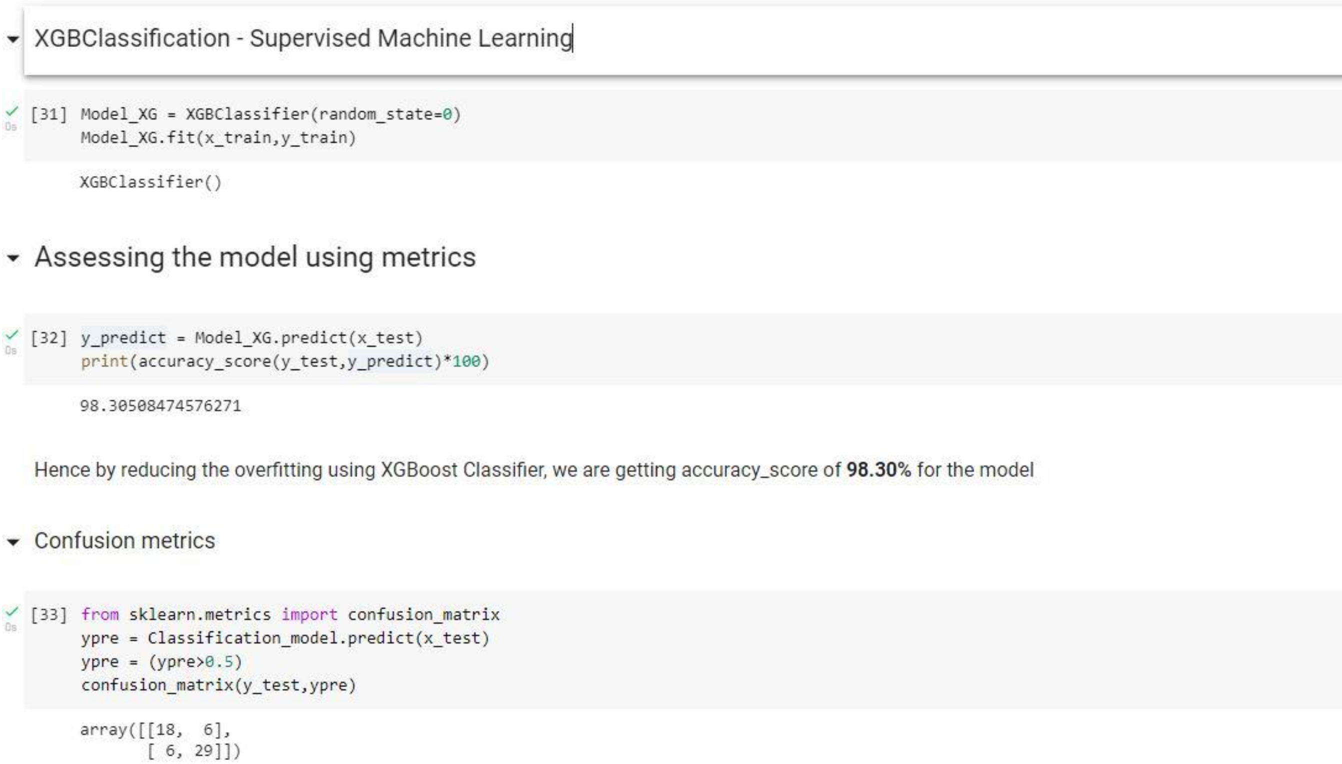
Hence by reducing the overfitting using XGBoost Classifier, we are getting accuracy_score of 98.30% for the model

Confusion metrics

```
[33] from sklearn.metrics import confusion_matrix
     ypre = classification_model.predict(x_test)
     ypre = (ypre>0.5)
     confusion_matrix(y_test,ypre)

     array([[18, 5],
            [ 6, 29]])
```

1) Metrics Parameter screenshot



XGBClassification - Supervised Machine Learning

```
[31] Model_XG = XGBClassifier(random_state=0)
     Model_XG.fit(x_train,y_train)
```

```
XGBClassifier()
```

Assessing the model using metrics

```
[32] y_predict = Model_XG.predict(x_test)
     print(accuracy_score(y_test,y_predict)*100)
```

```
98.30508474576271
```

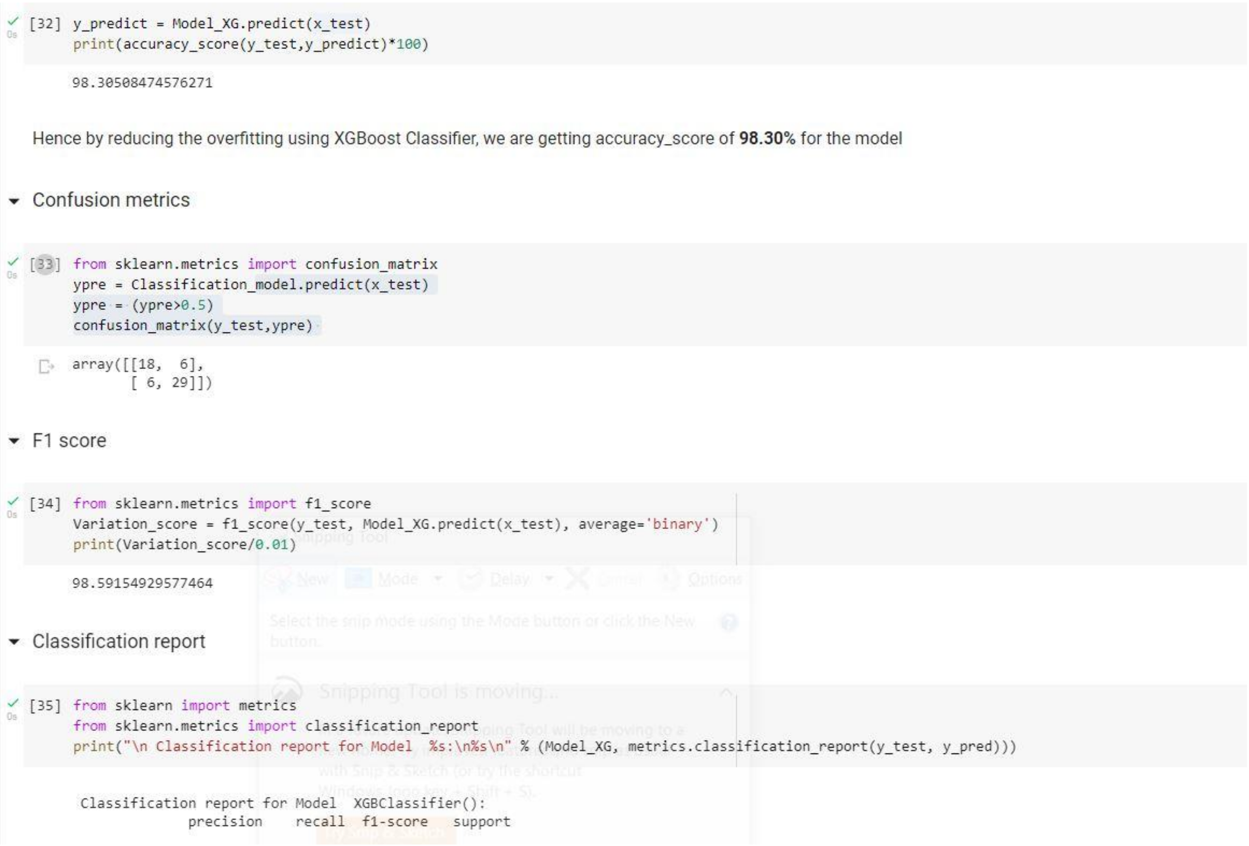
Hence by reducing the overfitting using XGBoost Classifier, we are getting accuracy_score of **98.30%** for the model

Confusion metrics

```
[33] from sklearn.metrics import confusion_matrix
     ypre = Classification_model.predict(x_test)
     ypre = (ypre>0.5)
     confusion_matrix(y_test,ypre)
```

```
array([[18,  6],
       [ 6, 29]])
```

2)Tune the model Parameter screenshot



```
[32] y_predict = Model_XG.predict(x_test)
     print(accuracy_score(y_test,y_predict)*100)
```

```
98.30508474576271
```

Hence by reducing the overfitting using XGBoost Classifier, we are getting accuracy_score of **98.30%** for the model

Confusion metrics

```
[33] from sklearn.metrics import confusion_matrix
     ypre = Classification_model.predict(x_test)
     ypre = (ypre>0.5)
     confusion_matrix(y_test,ypre)
```

```
array([[18,  6],
       [ 6, 29]])
```

F1 score

```
[34] from sklearn.metrics import f1_score
     Variation_score = f1_score(y_test, Model_XG.predict(x_test), average='binary')
     print(Variation_score/0.01)
```

```
98.59154929577464
```

Classification report

```
[35] from sklearn import metrics
     from sklearn.metrics import classification_report
     print("\n Classification report for Model  %s:\n%s\n" % (Model_XG, metrics.classification_report(y_test, y_pred)))
```

```
 Classification report for Model  XGBClassifier():
              precision    recall  f1-score   support
```