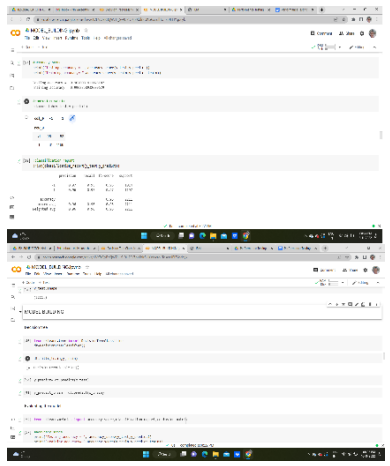
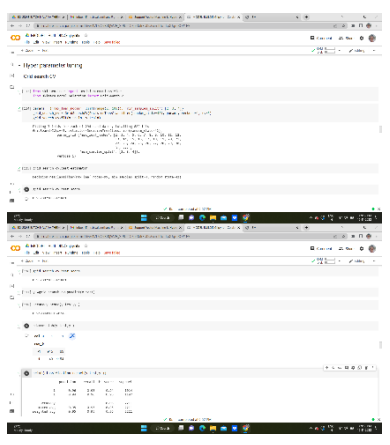


Project Development Phase Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID44074
Project Name	Project – WEB PHISHING DETECTION
Maximum Marks	10 Marks

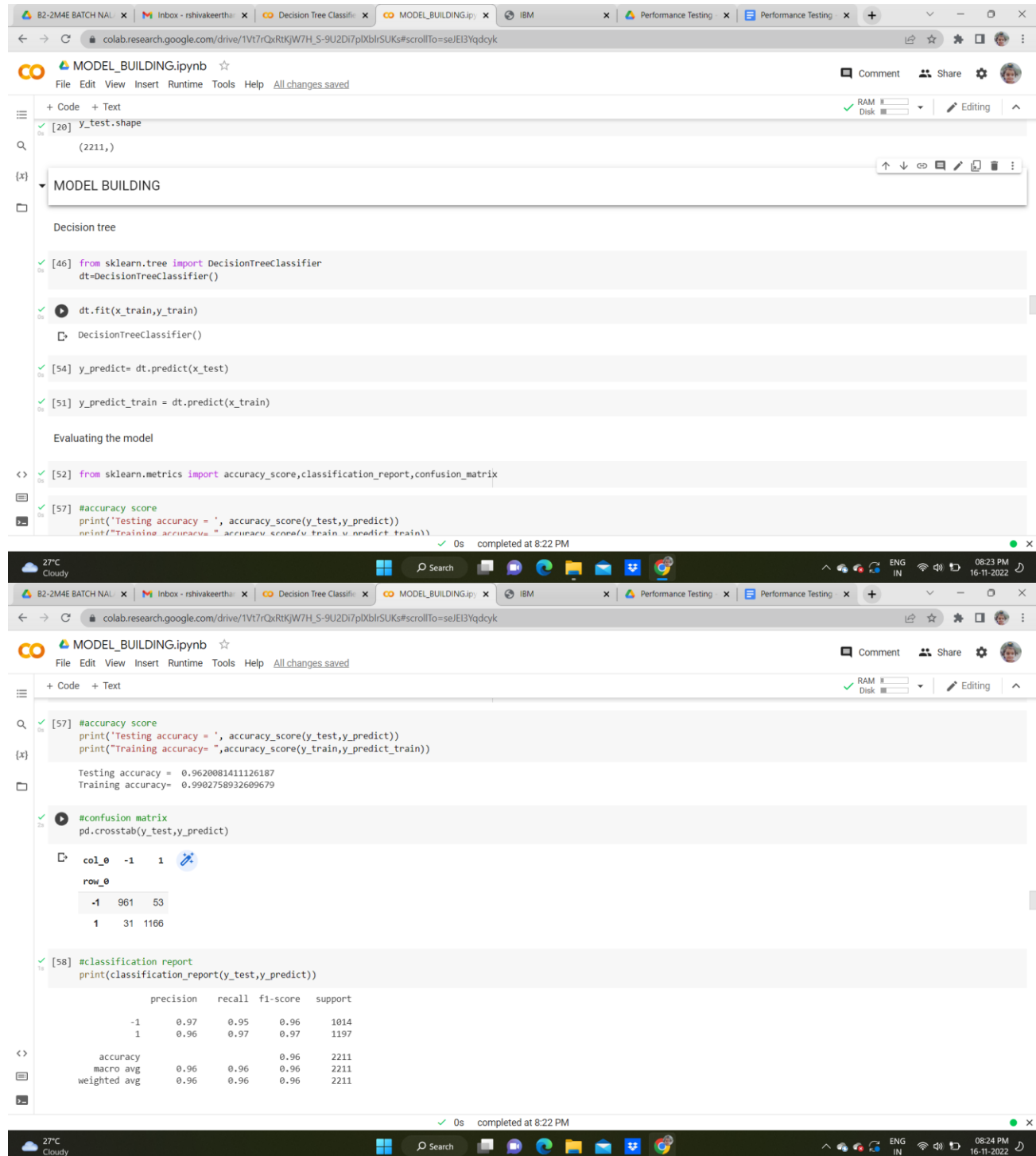
Model Performance Testing:

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

S.No	Parameter	Values	Screenshot
1.	Metrics	Classification Model: Confusion Matrix - , Accuracy Score- & Classification Report -	
2.	Tune the Model	Hyperparameter Tuning -	

1. Metrics

Classification Model: Confusion Matrix - , Accuracy Score- & Classification Report -



The following code is executed in the notebook:

```
[20] y_test.shape
(2211,)
```

MODEL BUILDING

Decision tree

```
[46] from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()

dt.fit(x_train,y_train)
DecisionTreeClassifier()

[54] y_predict= dt.predict(x_test)

[51] y_predict_train = dt.predict(x_train)
```

Evaluating the model

```
[52] from sklearn.metrics import accuracy_score,classification_report,confusion_matrix

[57] #accuracy score
print('Testing accuracy = ', accuracy_score(y_test,y_predict))
print('Training accuracy= ', accuracy_score(y_train,y_predict_train))
```

0s completed at 8:22 PM

The following code is executed in the notebook:

```
[57] #accuracy score
print('Testing accuracy = ', accuracy_score(y_test,y_predict))
print('Training accuracy= ', accuracy_score(y_train,y_predict_train))

Testing accuracy = 0.9620081411126187
Training accuracy= 0.9902758932609679

[58] #confusion matrix
pd.crosstab(y_test,y_predict)

col_0  -1    1
row_0
-1    961    53
 1     31  1166

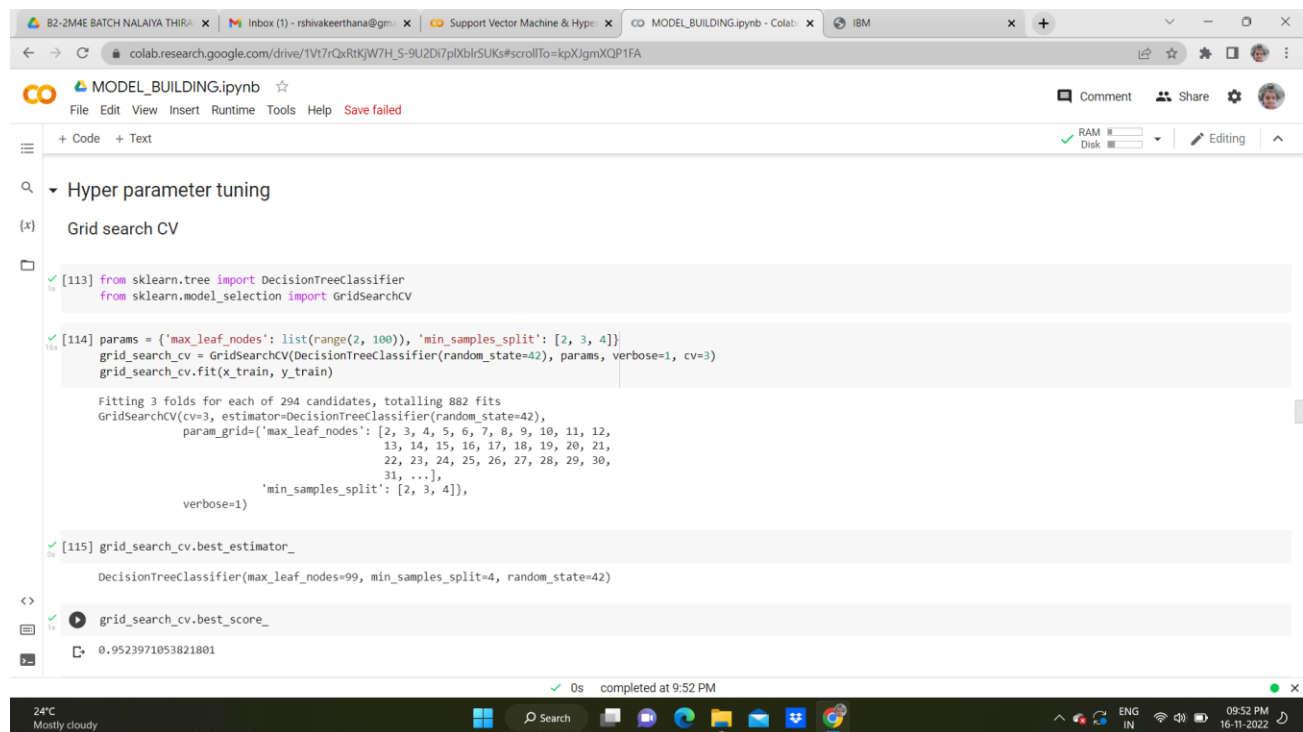
[58] #classification report
print(classification_report(y_test,y_predict))
```

	precision	recall	f1-score	support
-1	0.97	0.95	0.96	1014
1	0.96	0.97	0.97	1197
accuracy	0.96	0.96	0.96	2211
macro avg	0.96	0.96	0.96	2211
weighted avg	0.96	0.96	0.96	2211

0s completed at 8:22 PM

2. Tune the Model

Hyperparameter Tuning



The screenshot shows a Jupyter Notebook titled 'MODEL_BUILDING.ipynb' with the following code cells:

```
[113] from sklearn.tree import DecisionTreeClassifier
      from sklearn.model_selection import GridSearchCV

[114] params = {'max_leaf_nodes': list(range(2, 100)), 'min_samples_split': [2, 3, 4]}
      grid_search_cv = GridSearchCV(DecisionTreeClassifier(random_state=42), params, verbose=1, cv=3)
      grid_search_cv.fit(x_train, y_train)

      Fitting 3 folds for each of 294 candidates, totalling 882 fits
      GridSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=42),
        param_grid={'max_leaf_nodes': [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
          13, 14, 15, 16, 17, 18, 19, 20, 21,
          22, 23, 24, 25, 26, 27, 28, 29, 30,
          31, ...],
        'min_samples_split': [2, 3, 4]},
        verbose=1)

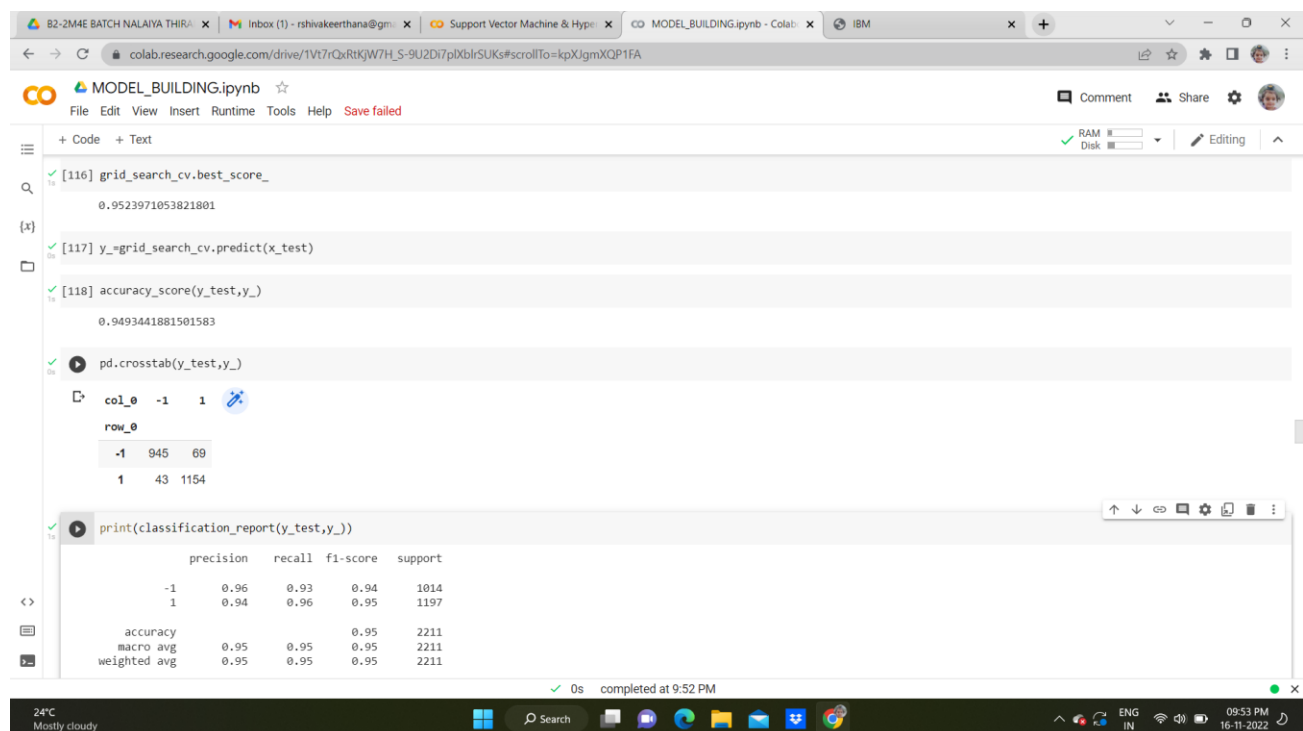
[115] grid_search_cv.best_estimator_

      DecisionTreeClassifier(max_leaf_nodes=99, min_samples_split=4, random_state=42)

grid_search_cv.best_score_

0.9523971053821801
```

The code is executed successfully, and the output shows the best score of 0.9523971053821801.



The screenshot shows the continuation of the Jupyter Notebook with the following code cells:

```
[116] grid_search_cv.best_score_

0.9523971053821801

[117] y_=grid_search_cv.predict(x_test)

[118] accuracy_score(y_test, y_)

0.9493441881501583

pd.crosstab(y_test, y_)

col_0    -1     1
row_0
-1    945    69
 1     43   1154

print(classification_report(y_test, y_))

              precision    recall  f1-score   support

-1               0.96         0.93         0.94         1014
 1               0.94         0.96         0.95         1197

accuracy          0.95
macro avg         0.95         0.95         0.95         2211
weighted avg      0.95         0.95         0.95         2211
```

The code is executed successfully, and the output shows the accuracy of the model and a classification report.

