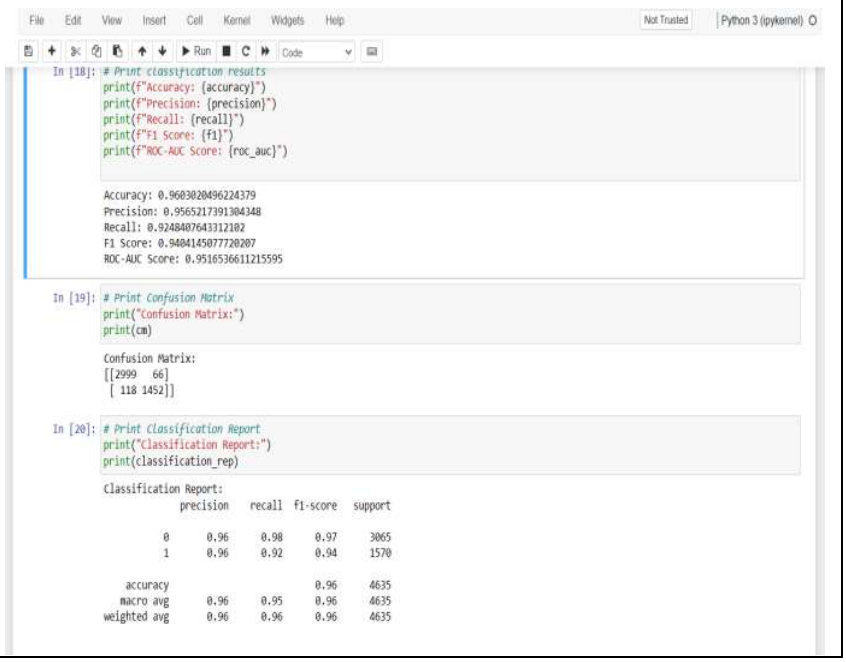
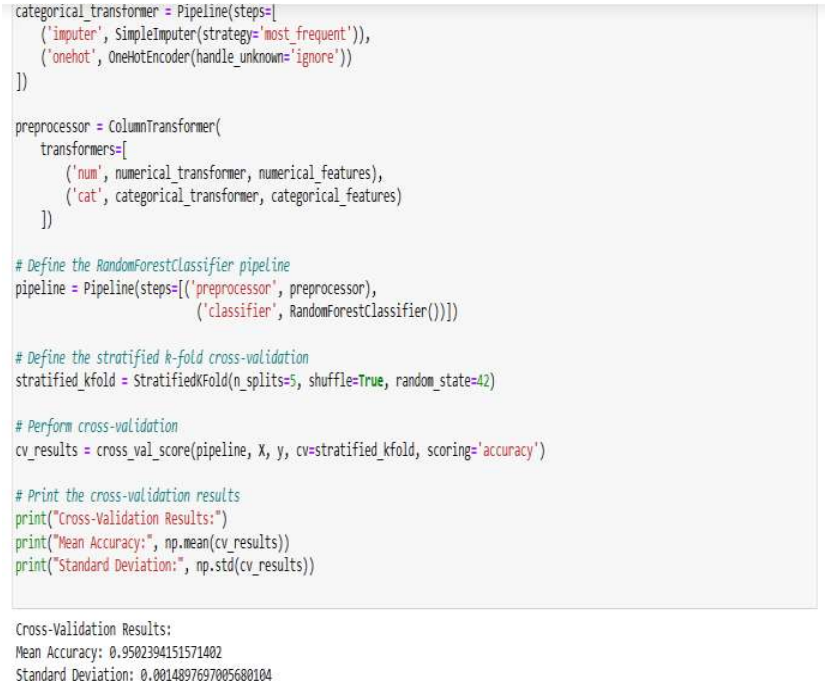
Project Development Phase
Model Performance Test

Date	9th November,2023
Team ID	Team-592973
Project Name	AIRLINE REVIEW CLASSIFICATION 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
	Metrics	**Regression Model:** MAE – 9798.736732205687, MSE – 96015241.547077, RMSE – 9798.736732205687, R2 score – 0.9258962347553819 **Classification Model:** Confusion Matrix – [[2999 66] [118 1452]] Accuracy Score- 0.9603020496224379 Classification Report – precision recall f1-score support 0 0.96 0.98 0.97 3065 1 0.96 0.92 0.94 1570 accuracy 0.96 4635 macro avg 0.96 0.95 0.96 4635 weighted avg 0.96 0.96 0.96 4635	 

	Tune the Model	Hyperparameter Tuning – Best Hyperparameters: {'classifier__n_estimators': 50, 'classifier__min_samples_split': 5, 'classifier__min_samples_leaf': 1, 'classifier__max_depth': None, 'classifier__bootstrap': False} Validation Method – Cross-Validation Results: Mean Accuracy: 0.9502394151571402 Standard Deviation: 0.0014897697005680104	 

1.

```
In [84]: #5.Evaluation of the model

In [85]: from sklearn import metrics

In [86]: # R- Square
         # evaluating testing accuracy
         #ACCURACY
         print(metrics.r2_score(y_test,y_pred))

0.9258962347553819

In [87]: #mean squared error
         print(metrics.mean_squared_error(y_test,y_pred))

96015241.547077

In [88]: # RMSE (Root Mean Square Error)
         print(np.sqrt(metrics.mean_squared_error(y_test,y_pred)))

9798.736732205687
```

2.

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
In [18]: # Print classification results
         print(f"Accuracy: {accuracy}")
         print(f"Precision: {precision}")
         print(f"Recall: {recall}")
         print(f"F1 Score: {f1}")
         print(f"ROC-AUC Score: {roc_auc}")

Accuracy: 0.9603020496224379
Precision: 0.9565217391304348
Recall: 0.9248407643312102
F1 Score: 0.9404145077720207
ROC-AUC Score: 0.9516536611215595

In [19]: # Print Confusion Matrix
         print("Confusion Matrix:")
         print(cm)

Confusion Matrix:
[[2999   66]
 [ 118 1452]]

In [20]: # Print Classification Report
         print("Classification Report:")
         print(classification_rep)

Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.98      0.97      3065
           1       0.96      0.92      0.94      1570

    accuracy                           0.96      4635
   macro avg       0.96      0.95      0.96      4635
weighted avg       0.96      0.96      0.96      4635
```

3.

```
# Perform RandomizedSearchCV
random_search = RandomizedSearchCV(pipeline, param_distributions=param_dist, n_iter=10, cv=5, scoring='accuracy', random_state=42
random_search.fit(X, y)

# Print the best hyperparameters
best_params = random_search.best_params_
print("Best Hyperparameters:")
print(best_params)
```

```
Best Hyperparameters:
{'classifier__n_estimators': 50, 'classifier__min_samples_split': 5, 'classifier__min_samples_leaf': 1, 'classifier__max_dept
h': None, 'classifier__bootstrap': False}
```

4.

```
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features)
    ])

# Define the RandomForestClassifier pipeline
pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                           ('classifier', RandomForestClassifier())])

# Define the stratified k-fold cross-validation
stratified_kfold = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

# Perform cross-validation
cv_results = cross_val_score(pipeline, X, y, cv=stratified_kfold, scoring='accuracy')

# Print the cross-validation results
print("Cross-Validation Results:")
print("Mean Accuracy:", np.mean(cv_results))
print("Standard Deviation:", np.std(cv_results))
```

```
Cross-Validation Results:
Mean Accuracy: 0.9502394151571402
Standard Deviation: 0.0014897697005680104
```