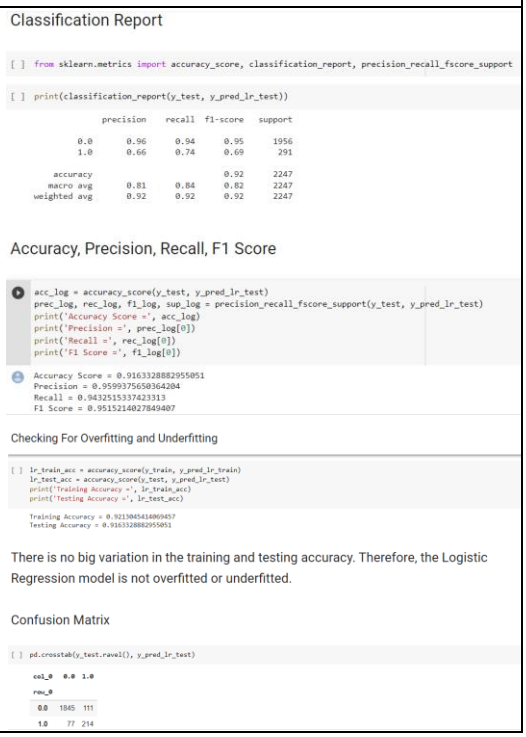
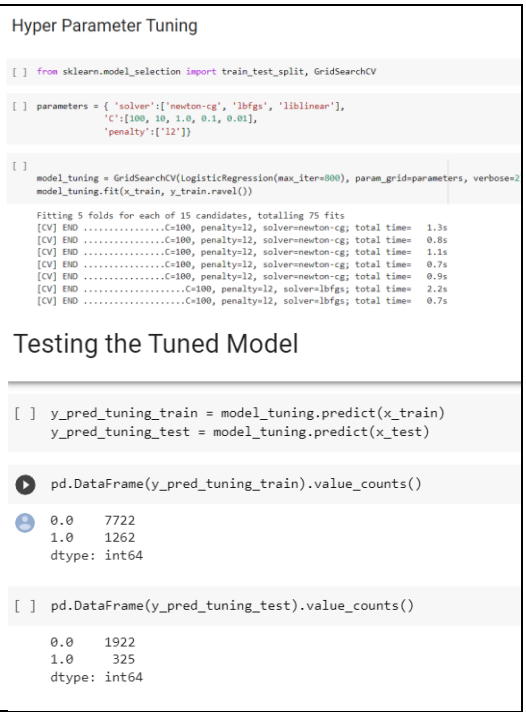
Project Development Phase
Model Performance Test

Date	04 NOVEMBER 2022
Team ID	PNT2022TMID32340
Project Name	DEVELOPING A FLIGHT DELAY PREDICTION MODEL USING MACHINE LEARNING ALGORITHM
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	**Regression Model:** MAE - , MSE - , RMSE - , R2 score - **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report -	

2.	Tune the Model	Hyperparameter Tuning - Validation Method -	

Hyper Parameter Tuning

```
[ ] from sklearn.model_selection import train_test_split, GridSearchCV
```

```
[ ] parameters = { 'solver':['newton-cg', 'lbfgs', 'liblinear'],
                   'C':[100, 10, 1.0, 0.1, 0.01],
                   'penalty':['l2']}
```

```
[ ]
    model_tuning = GridSearchCV(LogisticRegression(max_iter=800), param_grid=parameters, verbose=2
    model_tuning.fit(x_train, y_train.ravel())
```

```
Fitting 5 folds for each of 15 candidates, totalling 75 fits
[CV] END ..............C=100, penalty=l2, solver=newton-cg; total time=   1.3s
[CV] END ..............C=100, penalty=l2, solver=newton-cg; total time=   0.8s
[CV] END ..............C=100, penalty=l2, solver=newton-cg; total time=   1.1s
[CV] END ..............C=100, penalty=l2, solver=newton-cg; total time=   0.7s
[CV] END ..............C=100, penalty=l2, solver=newton-cg; total time=   0.9s
[CV] END ...................C=100, penalty=l2, solver=lbfgs; total time=   2.2s
[CV] END ...................C=100, penalty=l2, solver=lbfgs; total time=   0.7s
```

Testing the Tuned Model

```
[ ] y_pred_tuning_train = model_tuning.predict(x_train)
    y_pred_tuning_test = model_tuning.predict(x_test)
```

```
pd.DataFrame(y_pred_tuning_train).value_counts()
```

```
0.0    7722
1.0    1262
dtype: int64
```

```
[ ] pd.DataFrame(y_pred_tuning_test).value_counts()
```

```
0.0    1922
1.0     325
dtype: int64
```

			Evaluating the Tuned Model using Metrics
			Classification Report
			`[ ] print(classification_report(y_test, y_pred_tuning_test))`
			precision recall f1-score support
			0.0 0.96 0.94 0.95 1956
			1.0 0.66 0.74 0.69 291
			accuracy 0.92 2247
			macro avg 0.81 0.84 0.82 2247
			weighted avg 0.92 0.92 0.92 2247
			`[ ] #using F1SCORE`
			`acc_tuning = accuracy_score(y_test, y_pred_tuning_test)`
			`prec_tuning, rec_tuning, f1_tuning, sup_tuning = precision_recall_fscore_support(y_test, y_pred_tuning_te`
			`print('Accuracy Score =', acc_tuning)`
			`print('Precision =', prec_tuning[0])`
			`print('Recall =', rec_tuning[0])`
			`print('F1 Score =', f1_tuning[0])`
			Accuracy Score = 0.9163328882955051
			Precision = 0.9599375650364204
			Recall = 0.9432515337423313
			F1 Score = 0.9515214027849407
			Checking for Overfitting and Underfitting
			`tuning_train_acc = accuracy_score(y_train, y_pred_tuning_train)`
			`tuning_test_acc = accuracy_score(y_test, y_pred_tuning_test)`
			`print('Training Accuracy =', tuning_train_acc)`
			`print('Testing Accuracy =', tuning_test_acc)`
			Training Accuracy = 0.9214158504007124
			Testing Accuracy = 0.9163328882955051
			There is no big variation in the training and testing accuracy. Therefore, the Tune
			Logistic Regression model is not overfit or underfit.
			Confusion Matrix
			`[ ] pd.crosstab(y_test.ravel(), y_pred_tuning_test)`
			col_0 0.0 1.0
			row_0
			0.0 1845 111
			1.0 77 214