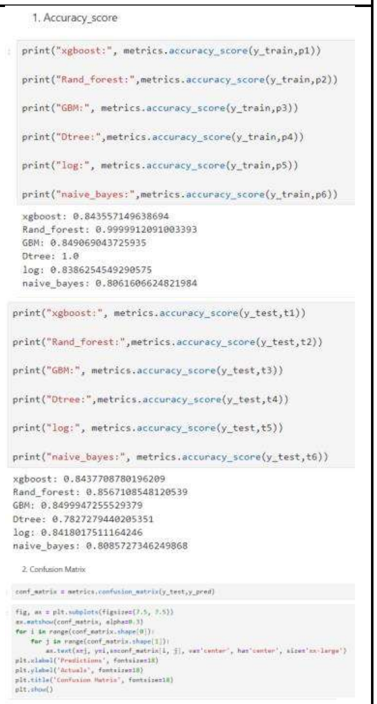
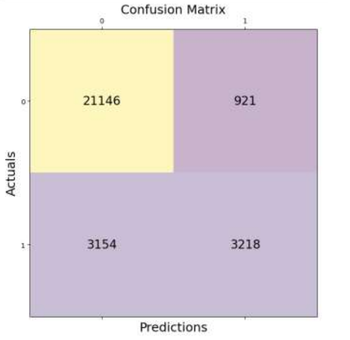
Project Development Phase
Model Performance Test

Date	20 NOvember 2022
Team ID	Team-591644
Project Name	Machine Learning Approach For Predicting The Rainfall
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 - , MSE - , RMSE - , R2 score - **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report -	 

1. Accuracy_score

```
print("xgboost:", metrics.accuracy_score(y_train,p1))

print("Rand_forest:",metrics.accuracy_score(y_train,p2))

print("GBM:", metrics.accuracy_score(y_train,p3))

print("Dtree:",metrics.accuracy_score(y_train,p4))

print("log:", metrics.accuracy_score(y_train,p5))

print("naive_bayes:",metrics.accuracy_score(y_train,p6))
```

```
xgboost: 0.843557149638694
Rand_forest: 0.9999912091003393
GBM: 0.849069043725935
Dtree: 1.0
log: 0.8386254549290575
naive_bayes: 0.8061606624821984
```

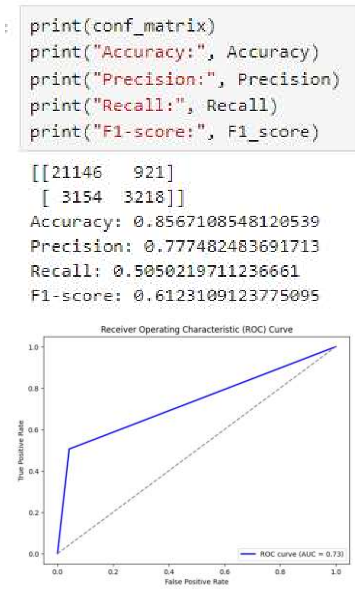
```
print("xgboost:", metrics.accuracy_score(y_test,t1))

print("Rand_forest:",metrics.accuracy_score(y_test,t2))

print("GBM:", metrics.accuracy_score(y_test,t3))

print("Dtree:",metrics.accuracy_score(y_test,t4))

print("log:", metrics.accuracy_score(y_test,t5))

print("naive_bayes:", metrics.accuracy_score(y_test,t6))
```

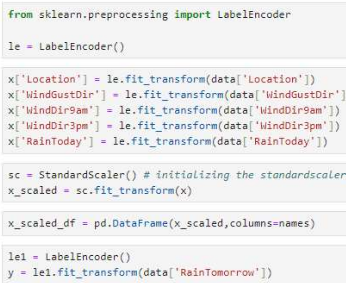
```
xgboost: 0.8437708780196209
Rand_forest: 0.8567108548120539
GBM: 0.849994725552937 9
Dtree: 0.7827279440205351
log: 0.8418017511164246
naive_bayes: 0.8085727346249868
```

2. Confusion Matrix

```
conf_matrix = metrics.confusion_matrix(y_test,y_pred)
```

```
fig, ax = plt.subplots(figsize=(7.5, 7.5))
ax.matshow(conf_matrix, alpha=0.3)
for i in range(conf_matrix.shape[0]):
    for j in range(conf_matrix.shape[1]):
        ax.text(x=j, y=i,s=conf_matrix[i, j], va='center', ha='center', size='xx-large')
plt.xlabel('Predictions', fontsize=18)
plt.ylabel('Actuals', fontsize=18)
plt.title('Confusion Matrix', fontsize=18)
plt.show()
```

			 print(conf_matrix) print("Accuracy:", Accuracy) print("Precision:", Precision) print("Recall:", Recall) print("F1-score:", F1_score) [[21146 921] [3154 3218]] Accuracy: 0.8567108548120539 Precision: 0.777482483691713 Recall: 0.5050219711236661 F1-score: 0.6123109123775095 Receiver Operating Characteristic (ROC) Curve True Positive Rate False Positive Rate 0.0 0.2 0.4 0.6 0.8 1.0 ROC curve (AUC = 0.73)

1.	Tune the Model	Hyperparameter Tuning -
		Validation Method -
			
			from sklearn.preprocessing import LabelEncoder
			
			le = LabelEncoder()
			
			x['Location'] = le.fit_transform(data['Location'])
			x['WindGustDir'] = le.fit_transform(data['WindGustDir']
			x['WindDir9am'] = le.fit_transform(data['WindDir9am'])
			x['WindDir3pm'] = le.fit_transform(data['WindDir3pm'])
			x['RainToday'] = le.fit_transform(data['RainToday'])
			
			sc = StandardScaler() # initializing the standardscaler
			x_scaled = sc.fit_transform(x)
			
			x_scaled_df = pd.DataFrame(x_scaled,columns=names)
			
			le1 = LabelEncoder()
			y = le1.fit_transform(data['RainTomorrow'])