

## Model Optimization and Tuning Phase Report


Date	15 July 2024
Team ID	739930
Project Title	Airline Review Classification
Maximum Marks	10 Marks

### Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

#### Hyperparameter Tuning Documentation (6 Marks):

# Logistic Regression



```
[ ] from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()

[ ] lr.fit(X_train,y_train)

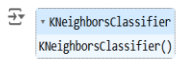
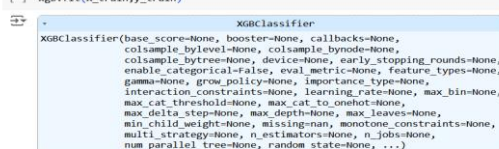
↳ - LogisticRegression
LogisticRegression()

[ ] pred_lr=lr.predict(X_test)
pred_lr


↳ array([1, 0, 0, ..., 1, 1, 0])

↳ from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
fpr_lr, tpr_lr, threshold_lr=roc_curve(y_test, pred_lr)
print(classification_report(y_test, pred_lr))
roc_auc_lr=auc(fpr_lr, tpr_lr)
print("roc_auc_lr:", roc_auc_lr)
cm_lr=confusion_matrix(y_test, pred_lr)
print("cm_lr:",cm_lr)
as_lr=accuracy_score(y_test, pred_lr)
print("as_lr:",as_lr)
```

	precision	recall	f1-score	support
0	0.93	0.92	0.92	3116
1	0.92	0.93	0.92	3030
accuracy			0.92	6146
macro avg	0.92	0.92	0.92	6146
weighted avg	0.92	0.92	0.92	6146

KNN	<pre>[ ] from sklearn.neighbors import KNeighborsClassifier knn=KNeighborsClassifier(n_neighbors=5)  [ ] knn.fit(X_train,y_train)</pre>  <pre>pred_knn=knn.predict(X_test) pred_knn  array([1, 0, 0, ..., 1, 1, 0])  [ ] from sklearn.metrics import classification_report, confusion_matrix, accuracy_score  fpr_knn, tpr_knn, threshold_knn = roc_curve(y_test, pred_knn) print(classification_report(y_test, pred_knn)) roc_auc_knn = auc(fpr_knn, tpr_knn) print("roc_auc_knn:", roc_auc_knn) cm_knn=confusion_matrix(y_test, pred_knn) print("cm_knn:",cm_knn) as_knn=accuracy_score(y_test, pred_knn) print("as_knn:",as_knn)</pre>	_____
XGB	<pre>[ ] from xgboost import XGBClassifier xgb=XGBClassifier()  [ ] xgb.fit(X_train,y_train)</pre>  <pre>pred_xgb=xgb.predict(X_test)</pre> <pre>from sklearn.metrics import classification_report, confusion_matrix, accuracy_score fpr_xgb, tpr_xgb, threshold_xgb=roc_curve(y_test, pred_xgb) print(classification_report(y_test, pred_xgb)) roc_auc_xgb=auc(fpr_xgb, tpr_xgb) print("roc_auc_xgb:", roc_auc_xgb) cm_xgb=confusion_matrix(y_test, pred_xgb) print("cm_xgb:",cm_xgb) as_xgb=accuracy_score(y_test, pred_xgb) print("as_xgb:",as_xgb)</pre>	_____

### Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric						
Decision Tree							
			precision	recall	f1-score	support	
		0	0.95	0.95	0.95	3116	
		1	0.95	0.95	0.95	3030	
		accuracy				0.95	6146
		macro avg	0.95	0.95	0.95	6146	
		weighted avg	0.95	0.95	0.95	6146	
		roc_auc_dt 0.9479143100446116					
		cm_dt: [[2958 158]					
		[ 162 2868]]					
as_dt: 0.9479336153595834							

Random Forest	<div><div><div></div></div><div><div></div></div></div> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>0.95</td><td>0.96</td><td>0.96</td><td>3116</td></tr><tr><td>1</td><td>0.96</td><td>0.95</td><td>0.96</td><td>3030</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.96</td><td>6146</td></tr><tr><td>macro avg</td><td>0.96</td><td>0.96</td><td>0.96</td><td>6146</td></tr><tr><td>weighted avg</td><td>0.96</td><td>0.96</td><td>0.96</td><td>6146</td></tr><tr><td colspan="5">roc_auc_rfc 0.9582704194681343</td></tr><tr><td colspan="5">cm_rfc: [[3003 113]</td></tr><tr><td colspan="5">[ 143 2887]]</td></tr><tr><td colspan="5">as_rfc: 0.9583468922876668</td></tr></tbody></table>		precision	recall	f1-score	support	0	0.95	0.96	0.96	3116	1	0.96	0.95	0.96	3030	accuracy			0.96	6146	macro avg	0.96	0.96	0.96	6146	weighted avg	0.96	0.96	0.96	6146	roc_auc_rfc 0.9582704194681343					cm_rfc: [[3003 113]					[ 143 2887]]					as_rfc: 0.9583468922876668				
	precision	recall	f1-score	support																																															
0	0.95	0.96	0.96	3116																																															
1	0.96	0.95	0.96	3030																																															
accuracy			0.96	6146																																															
macro avg	0.96	0.96	0.96	6146																																															
weighted avg	0.96	0.96	0.96	6146																																															
roc_auc_rfc 0.9582704194681343																																																			
cm_rfc: [[3003 113]																																																			
[ 143 2887]]																																																			
as_rfc: 0.9583468922876668																																																			
KNN	<div><div><div></div></div><div><div></div></div></div> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>0.96</td><td>0.93</td><td>0.95</td><td>3116</td></tr><tr><td>1</td><td>0.93</td><td>0.96</td><td>0.95</td><td>3030</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.95</td><td>6146</td></tr><tr><td>macro avg</td><td>0.95</td><td>0.95</td><td>0.95</td><td>6146</td></tr><tr><td>weighted avg</td><td>0.95</td><td>0.95</td><td>0.95</td><td>6146</td></tr><tr><td colspan="5">roc_auc_knn: 0.945478992700297</td></tr><tr><td colspan="5">cm_knn: [[2913 203]</td></tr><tr><td colspan="5">[ 133 2897]]</td></tr><tr><td colspan="5">as_knn: 0.9453302961275627</td></tr></tbody></table>		precision	recall	f1-score	support	0	0.96	0.93	0.95	3116	1	0.93	0.96	0.95	3030	accuracy			0.95	6146	macro avg	0.95	0.95	0.95	6146	weighted avg	0.95	0.95	0.95	6146	roc_auc_knn: 0.945478992700297					cm_knn: [[2913 203]					[ 133 2897]]					as_knn: 0.9453302961275627				
	precision	recall	f1-score	support																																															
0	0.96	0.93	0.95	3116																																															
1	0.93	0.96	0.95	3030																																															
accuracy			0.95	6146																																															
macro avg	0.95	0.95	0.95	6146																																															
weighted avg	0.95	0.95	0.95	6146																																															
roc_auc_knn: 0.945478992700297																																																			
cm_knn: [[2913 203]																																																			
[ 133 2897]]																																																			
as_knn: 0.9453302961275627																																																			
XGB	<div><div><div></div></div><div><div></div></div></div> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>0.96</td><td>0.97</td><td>0.96</td><td>3116</td></tr><tr><td>1</td><td>0.97</td><td>0.96</td><td>0.96</td><td>3030</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.96</td><td>6146</td></tr><tr><td>macro avg</td><td>0.96</td><td>0.96</td><td>0.96</td><td>6146</td></tr><tr><td>weighted avg</td><td>0.96</td><td>0.96</td><td>0.96</td><td>6146</td></tr><tr><td colspan="5">roc_auc_xgb: 0.9630194630502845</td></tr><tr><td colspan="5">cm_xgb: [[3011 105]</td></tr><tr><td colspan="5">[ 122 2908]]</td></tr><tr><td colspan="5">as_xgb: 0.9630654083957045</td></tr></tbody></table>		precision	recall	f1-score	support	0	0.96	0.97	0.96	3116	1	0.97	0.96	0.96	3030	accuracy			0.96	6146	macro avg	0.96	0.96	0.96	6146	weighted avg	0.96	0.96	0.96	6146	roc_auc_xgb: 0.9630194630502845					cm_xgb: [[3011 105]					[ 122 2908]]					as_xgb: 0.9630654083957045				
	precision	recall	f1-score	support																																															
0	0.96	0.97	0.96	3116																																															
1	0.97	0.96	0.96	3030																																															
accuracy			0.96	6146																																															
macro avg	0.96	0.96	0.96	6146																																															
weighted avg	0.96	0.96	0.96	6146																																															
roc_auc_xgb: 0.9630194630502845																																																			
cm_xgb: [[3011 105]																																																			
[ 122 2908]]																																																			
as_xgb: 0.9630654083957045																																																			

**Final Model Selection Justification (2 Marks):**

Final Model	Reasoning
Xextreme Gradient Boosting	The Xextreme Gradient Boosting model was selected for its superior performance, exhibiting high accuracy . Its ability to handle complex relationships, minimize overfitting, and optimize predictive accuracy aligns with project objectives, justifying its selection as the final model.