

Model Optimization and Tuning Phase Template

Date	03-10-2024
Team ID	LTVIP2024TMID24897
Project Title	Flight delay prediction using ML
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):

Model	Tuned Hyperparameters	Optimal Values
Decision tree	<pre>from sklearn.tree import DecisionTreeClassifier classifierDT = DecisionTreeClassifier(criterion = 'entropy', random_state = None) classifierDT.fit(X_train_sc, y_train)</pre>	<pre>DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=None, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best')</pre>
KNN	<pre>from sklearn.neighbors import KNeighborsClassifier objClassifier=KNeighborsClassifier(n_neighbors=10,metric='minkowski',p=2) objClassifier.fit(X_train_sc,y_train)</pre>	<pre>KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=None, n_neighbors=10, p=2, weights='uniform')</pre>
Logistic regression	<pre>from sklearn.linear_model import LogisticRegression classifier = LogisticRegression(random_state = 0) classifier.fit(X_train_sc, y_train)</pre>	<pre>LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, l1_ratio=None, max_iter=100, multi_class='warn', n_jobs=None, penalty='l2', random_state=0, solver='warn', tol=0.0001, verbose=0, warm_start=False)</pre>

Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric, Optimized Metrics
Decision tree	<pre>print("F1 score :",f1_score(y_test, y_pred, average="macro")) print("Precision Score : " , precision_score(y_test, y_pred, average="macro")) print("Recall Score : " , recall_score(y_test, y_pred, average="macro"))</pre> <p>F1 score : 0.2725298912293622 Precision Score : 0.6644134619299129 Recall Score : 0.5006117468892067</p>
KNN	<pre>print("F1 score :",f1_score(y_test, y_pred, average="macro")) print("Precision Score : " , precision_score(y_test, y_pred, average="macro")) print("Recall Score : " , recall_score(y_test, y_pred, average="macro"))</pre> <p>F1 score : 0.8696222002024336 Precision Score : 0.8825899500527832 Recall Score : 0.8609813308550668</p>

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Decision Tree	<pre># Predicting the Test set results y_pred = classifierDT.predict(X_test) # Making the Confusion Matrix cm = confusion_matrix(y_test, y_pred) score = classifierDT.score(X_test_sc,y_test)</pre> <p>cm</p> <pre>array([[1303, 982223], [59, 582189]])</pre> <p>score</p> <p>0.9826194584914554</p>

	<p>This model has been selected because it has the high accuracy and f1-score compared to the other model mentioned above.</p>
--	--