

Model Optimization and Tuning Phase Template

Date	15 July 2024
Team ID	740080
Project Title	Flight Delay Prediction using Machine Learning.
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
Random Forest Classifier	-----	<pre> from sklearn.ensemble import RandomForestClassifier from sklearn.metrics import classification_report, confusion_matrix rfc = RandomForestClassifier() rfc.fit(x_train,y_train) y_test_predict1 = rfc.predict(x_test) test_accuracy = accuracy_score(y_test,y_test_predict1) test_accuracy c:\Users\indhu\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\base.py:1473: return fit_method(estimator, *args, **kwargs) 0.9159561797752889 </pre>

Logistic Regression	-----	<pre> lr = LogisticRegression() lr.fit(x_train,y_train) y_test_predict2 = lr.predict(x_test) test_accuracy = accuracy_score(y_test,y_test_predict2) test_accuracy c:\Users\lindhu\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\utils_validation.py:1133: DataConversionWarning: A column-vector y was y = columnar_ufunc, warn=True) c:\Users\lindhu\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\linear_model_logistic.py:489: ConvergenceWarning: lbfgs failed to con STOP: TOTAL NO. of ITERATIONS REACHED LIMIT. Increase the number of iterations (max_iter) or scale the data as shown in: https://bit.ly/lars-ard/sharing/2020/04/04/ridge/ Please also refer to the documentation for alternative solver options: https://bit.ly/lars-ard/sharing/2020/04/04/ridge/linear_model.html#logistic-regression r_star_1 = _check_optimal_result(0.916382479598113 </pre>
Decision Tree Classifier	-----	<pre> dtc = DecisionTreeClassifier() dtc.fit(x_train,y_train) y_test_predict3 = dtc.predict(x_test) test_accuracy = accuracy_score(y_test,y_test_predict3) test_accuracy 0.8606741573033708 </pre>
Extra Tree Classifier	-----	<pre> etc = ExtraTreesClassifier() etc.fit(x_train,y_train) y_test_predict4 = etc.predict(x_test) test_accuracy = accuracy_score(y_test,y_test_predict4) test_accuracy c:\Users\lindhu\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\base.py:1478: return fit_method(estimator, *args, **kwargs) 0.8687640440433203 </pre>

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric
-------	------------------

Random Forest Classifier	<pre>print(classification_report(y_test,y_test_predict1))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0.0</td><td>0.94</td><td>0.96</td><td>0.95</td><td>1932</td></tr><tr><td>1.0</td><td>0.71</td><td>0.59</td><td>0.65</td><td>293</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.92</td><td>2225</td></tr><tr><td>macro avg</td><td>0.83</td><td>0.78</td><td>0.80</td><td>2225</td></tr><tr><td>weighted avg</td><td>0.91</td><td>0.92</td><td>0.91</td><td>2225</td></tr></tbody></table> <pre>confusion_matrix(y_test, y_test_predict1)</pre> <pre>array([[1863, 69], [120, 173]])</pre>		precision	recall	f1-score	support	0.0	0.94	0.96	0.95	1932	1.0	0.71	0.59	0.65	293	accuracy			0.92	2225	macro avg	0.83	0.78	0.80	2225	weighted avg	0.91	0.92	0.91	2225
	precision	recall	f1-score	support																											
0.0	0.94	0.96	0.95	1932																											
1.0	0.71	0.59	0.65	293																											
accuracy			0.92	2225																											
macro avg	0.83	0.78	0.80	2225																											
weighted avg	0.91	0.92	0.91	2225																											
Linear Regression	<pre>y_train_predict2 = rfc.predict(x_train) train_accuracy = accuracy_score(y_train,y_train_predict2) train_accuracy print(classification_report(y_test,y_test_predict2))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0.0</td><td>0.95</td><td>0.96</td><td>0.95</td><td>1932</td></tr><tr><td>1.0</td><td>0.70</td><td>0.65</td><td>0.68</td><td>293</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.92</td><td>2225</td></tr><tr><td>macro avg</td><td>0.83</td><td>0.81</td><td>0.82</td><td>2225</td></tr><tr><td>weighted avg</td><td>0.92</td><td>0.92</td><td>0.92</td><td>2225</td></tr></tbody></table> <pre>confusion_matrix(y_test, y_test_predict2)</pre> <pre>array([[1852, 80], [102, 191]])</pre>		precision	recall	f1-score	support	0.0	0.95	0.96	0.95	1932	1.0	0.70	0.65	0.68	293	accuracy			0.92	2225	macro avg	0.83	0.81	0.82	2225	weighted avg	0.92	0.92	0.92	2225
	precision	recall	f1-score	support																											
0.0	0.95	0.96	0.95	1932																											
1.0	0.70	0.65	0.68	293																											
accuracy			0.92	2225																											
macro avg	0.83	0.81	0.82	2225																											
weighted avg	0.92	0.92	0.92	2225																											
Decision Tree Classifier	<pre>y_train_predict3 = dtc.predict(x_train) train_accuracy = accuracy_score(y_train,y_train_predict3) train_accuracy print(classification_report(y_test,y_test_predict3))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0.0</td><td>0.92</td><td>0.92</td><td>0.92</td><td>1932</td></tr><tr><td>1.0</td><td>0.47</td><td>0.48</td><td>0.48</td><td>293</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.86</td><td>2225</td></tr><tr><td>macro avg</td><td>0.70</td><td>0.70</td><td>0.70</td><td>2225</td></tr><tr><td>weighted avg</td><td>0.86</td><td>0.86</td><td>0.86</td><td>2225</td></tr></tbody></table> <pre>confusion_matrix(y_test, y_test_predict3)</pre> <pre>array([[1773, 159], [151, 142]])</pre>		precision	recall	f1-score	support	0.0	0.92	0.92	0.92	1932	1.0	0.47	0.48	0.48	293	accuracy			0.86	2225	macro avg	0.70	0.70	0.70	2225	weighted avg	0.86	0.86	0.86	2225
	precision	recall	f1-score	support																											
0.0	0.92	0.92	0.92	1932																											
1.0	0.47	0.48	0.48	293																											
accuracy			0.86	2225																											
macro avg	0.70	0.70	0.70	2225																											
weighted avg	0.86	0.86	0.86	2225																											

Extra Tree Classifier

```
y_train_predict4 = etc.predict(x_train)
train_accuracy = accuracy_score(y_train,y_train_predict4)
train_accuracy
print(classification_report(y_test,y_test_predict4))
```

	precision	recall	f1-score	support
0.0	0.94	0.96	0.95	1932
1.0	0.69	0.56	0.62	293
accuracy			0.91	2225
macro avg	0.81	0.76	0.78	2225
weighted avg	0.90	0.91	0.90	2225

```
confusion_matrix(y_test, y_test_predict4)
```

```
array([[1857, 75],
       [ 128, 165]])
```

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random Forest Classifier	The Random Forest model was selected for its superior performance, exhibiting high accuracy during train and test. Its often more accurate than decision tree it builds multiple tree and averages their predictions, reducing the risk of overfitting. It can model non-linear relationships better than Linear Regression. Effective in detecting anomalies in datasets, useful in fraud detection and network security.