Model Optimization and Tuning Phase Template

Date	13 July 2024
Team ID	SWTID1720174957
Project Title	Human Resource Management: Predicting Employee Promotions 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

DecisionTree Classifier	`dt = DecisionTreeClassifier(criterion='entropy',max_depth=5,min_samples_split=10, min_samples_leaf=5,random_state=42)`
	`dt.fit(x_train,y_train)`
	`y_pred_dt=dt.predict(x_test)`
		`accuracy_score(y_test,y_pred_dt)`
		`0.7149980055843638`

RandomForest Classifier	`rf=RandomForestClassifier(n_estimators=100, max_depth=5, random_state=42)`
	`rf.fit(x_train,y_train)`
	`y_pred_rf=rf.predict(x_test)`
		`accuracy_score(y_test,y_pred_rf)`
		`0.7957718388512166`

Kneighbours Classifier	`knn=KNeighborsClassifier(n_neighbors=3, weights='uniform',algorithm='auto',leaf_size=10)`
	`knn.fit(x_train,y_train)`
	`y_pred_kn=knn.predict(x_test)`
		`accuracy_score(y_test,y_pred_kn)`
		`0.9032708416433984`

GradientBoosting Classifier	`xg=GradientBoostingClassifier(n_estimators=100, learning_rate=0.1, max_depth=3, random_state=42)`
	`xg.fit(x_train,y_train)`
	`y_pred_xg=xg.predict(x_test)`
		`accuracy_score(y_test,y_pred_xg)`
		`0.864379736737136`

Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric	Optimized Metric

DecisionTree Classifier	`classification_report(y_test,y_pred)`
	precision recall f1-score support
	0 0.95 0.93 0.94 10035
	1 0.93 0.95 0.94 10021
	accuracy 0.94 20056
	macro avg 0.94 0.94 0.94 20056
	weighted avg 0.94 0.94 0.94 20056
	`confusion_matrix(y_test,y_pred)`
	`array([[9289, 746],`
	`[ 510, 9511]], dtype=int64)`
		`y_pred_dt=decisionTree(x_train,x_test,y_train,y_test)`
		DecisionTreeClassifier
		Confusion matrix
		[[6474 3561]
		[2155 7866]]
		Classification report
		precision recall f1-score support
		0 0.75 0.65 0.69 10035
		1 0.69 0.78 0.73 10021
		accuracy 0.71 20056
		macro avg 0.72 0.72 0.71 20056
		weighted avg 0.72 0.71 0.71 20056

RandomForest Classifier	`classification_report(y_test,y_pred)`
	precision recall f1-score support
	0 0.95 0.95 0.95 10035
	1 0.95 0.95 0.95 10021
	accuracy 0.95 20056
	macro avg 0.95 0.95 0.95 20056
	weighted avg 0.95 0.95 0.95 20056
	`confusion_matrix(y_test,y_pred)`
	`array([[9498, 537],`
	`[ 477, 9544]], dtype=int64)`
		`y_pred_rf=randomForest(x_train,x_test,y_train,y_test)`
		RandomForestClassifier
		Confusion matrix
		[[7317 2718]
		[1378 8643]]
		Classification report
		precision recall f1-score support
		0 0.84 0.73 0.78 10035
		1 0.76 0.86 0.81 10021
		accuracy 0.80 20056
		macro avg 0.80 0.80 0.79 20056
		weighted avg 0.80 0.80 0.79 20056

<table>
<tr>
<td>Kneighbours Classifier</td>
<td>

```
classification_report(y_test,y_pred)
```

```
              precision    recall  f1-score   support

           0       0.96      0.82      0.89     10035
           1       0.84      0.97      0.90     10021

    accuracy                           0.90     20056
   macro avg       0.90      0.90      0.89     20056
weighted avg       0.90      0.90      0.89     20056
```

```
confusion_matrix(y_test,y_pred)
```

```
array([[8242, 1793],
       [ 308, 9713]], dtype=int64)
```

</td>
<td>

```
y_pred_kn=KNN(x_train,x_test,y_train,y_test)
```

```
Confusion matrix
[[8451 1584]
 [ 356 9665]]
Classification report
              precision    recall  f1-score   support

           0       0.96      0.84      0.90     10035
           1       0.86      0.96      0.91     10021

    accuracy                           0.90     20056
   macro avg       0.91      0.90      0.90     20056
weighted avg       0.91      0.90      0.90     20056
```

</td>
</tr>
<tr>
<td>GradientBoosting Classifier</td>
<td>

```
classification_report(y_test,y_pred)
```

```
              precision    recall  f1-score   support

           0       0.88      0.84      0.86     10035
           1       0.85      0.89      0.87     10021

    accuracy                           0.86     20056
   macro avg       0.87      0.86      0.86     20056
weighted avg       0.87      0.86      0.86     20056
```

```
confusion_matrix(y_test,y_pred)
```

```
array([[8409, 1626],
       [1094, 8927]], dtype=int64)
```

</td>
<td>

```
y_pred_xg=xgboost(x_train,x_test,y_train,y_test)
```

```
Confusion matrix
[[8409 1626]
 [1094 8927]]
Classification report
              precision    recall  f1-score   support

           0       0.88      0.84      0.86     10035
           1       0.85      0.89      0.87     10021

    accuracy                           0.86     20056
   macro avg       0.87      0.86      0.86     20056
weighted avg       0.87      0.86      0.86     20056
```

</td>
</tr>
</table>

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
RandomForest Classifier	The RandomForest Classifier was chosen because it performs well across a range of datasets and doesn't require a lot of hyperparameter tweaking. By combining several decision trees—each trained on a different collection of attributes and observations it successfully minimizes overfitting. This ensemble technique

	effectively manages noisy data and outliers while enhancing generalization.