Model Optimization and Tuning Phase Report

Date	03 may 2024
Team ID	738323
Project Title	SmartLender - Applicant Credibility Prediction for Loan Approval
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):

<table>
<tr><th>Model</th><th>Tuned Hyperparameters</th><th>Optimal Values</th></tr>
<tr><td>Decision Tree</td><td><pre>#Define the hyperparameters and thier possible values for tuning
param_grid={
    'criterion':['gini','entropy'],
    'splitter':['best','random'],
    'max_depth':[None,10,20,30,40,50],
    'min_samples_split':[2,5,10],
    'min_samples_leaf':[1,2,4],

}</pre></td><td><pre>#evaluate the performence of the tuned model
accuracy=accuracy_score(y_test,dt_pred)
print(f'optimal parameters:{best_params}')
print(f'accuracy on test set:{accuracy}')

optimal parameters:{'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 1,
accuracy on test set:0.7928571428571428</pre></td></tr>
<tr><td>Random

Forest</td><td><pre>rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

#define the hyperparametrs and thier possible values for tuning
param_grid={
    'n_estimators':[50,100,200],
    'criterion':['gini','entropy'],
    'max_depth':[None,10,20,30],
    'min_samples_split':[2,5,10],
    'min_samples_leaf':[1,2,4],
}</pre></td><td><pre>[192] #evaluate the performence of the tuned model
      accuracy=accuracy_score(y_test,rf_pred)
      print(f'optimal parameters:{best_params}')
      print(f'accuracy on test set:{accuracy}')

      optimal parameters:{'learning_rate': 0.2, 'max_depth': 5, 'min_samples_leaf': 1, '
      accuracy on test set:0.85</pre></td></tr>
</table>

KNN	[164] knn_model = KNeighborsClassifier(n_neighbors=5)
	
	[165] param_grid={
	'n_neighbors':[3,5,7,9],
	'weights':['uniform','distance'],
	'p':[1,2]
	}
		#evaluate the performence of the tuned model
		accuracy=accuracy_score(y_test,knn_pred)
		print(f'optimal parameters:{best_params}')
		print(f'accuracy on test set:{accuracy}')
		
		optimal parameters:{'n_neighbors': 9, 'p': 1, 'weights': 'distance'}
		accuracy on test set:0.8142857142857143

Gradient Boosting	174] xgb=XGBClassifier()
	
	175] #define hyperparamaters and thier possible values for tuning
	param_grid={
	'n_estimators':[50,100,200],
	'learning_rate':[0.01,0.1,0.2],
	'max_depth':[2,5,10],
	'min_samples_leaf':[1,2,4],
	'subsample':[0.8,1.0]
	}
		#evaluate the performence of the tuned model
		accuracy=accuracy_score(y_test,xgb_pred)
		print(f'optimal parameters:{best_params}')
		print(f'accuracy on test set:{accuracy}')
		
		optimal parameters:{'learning_rate': 0.2, 'max_depth': 5, 'min_samples_leaf
		accuracy on test set:0.8428571428571429

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric
Decision Tree	print(confusion_matrix(y_test, Loading... print(classification_report(y_test,dt_pred)) [[54 16] [13 57]] precision recall f1-score support 0 0.81 0.77 0.79 70 1 0.78 0.81 0.80 70 accuracy 0.79 140 macro avg 0.79 0.79 0.79 140 weighted avg 0.79 0.79 0.79 140

<table>
<tr>
<td>Random Forest</td>
<td>

```
print(confusion_matrix(y_test,rf_pred))
print(classification_report(y_test,rf_pred))
```

```
[[58 12]
 [ 9 61]]
              precision    recall  f1-score   support

           0       0.87      0.83      0.85        70
           1       0.84      0.87      0.85        70

    accuracy                           0.85       140
   macro avg       0.85      0.85      0.85       140
weighted avg       0.85      0.85      0.85       140
```

</td>
</tr>
<tr>
<td>KNN</td>
<td>

```
173] print(confusion_matrix(y_test,knn_pred))
     print(classification_report(y_test,knn_pred))
```

```
[[57 13]
 [13 57]]
              precision    recall  f1-score   support

           0       0.81      0.81      0.81        70
           1       0.81      0.81      0.81        70

    accuracy                           0.81       140
   macro avg       0.81      0.81      0.81       140
weighted avg       0.81      0.81      0.81       140
```

</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>

```
print(confusion_matrix(y_test,xgb_pred))
print(classification_report(y_test,xgb_pred))
```

```
[[57 13]
 [ 9 61]]
              precision    recall  f1-score   support

           0       0.86      0.81      0.84        70
           1       0.82      0.87      0.85        70

    accuracy                           0.84       140
   macro avg       0.84      0.84      0.84       140
weighted avg       0.84      0.84      0.84       140
```

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

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random Forest	The Random Forest 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.