Model Optimization and Tuning Phase Template

Date	12 July 2024
Team ID	SWTID1720108776
Project Title	Ecommerce Shipping 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):

Selected 3 models for Hyper parameter Tuning

Model	Tuned Hyperparameters	Optimal Values

SVM(Support Vector Machine)	Kernel, C, gamma.
	`parameters={ 'kernel':['rbf'], 'C':[0.1,0.01], 'gamma':[0.01,0.0001] }`
		Accuracy = 70%
		`print(fit1.best_estimator_,fit1.best_params_,fit1.best_score_)`
		`SVC(C=0.1, gamma=0.01) {'C': 0.1, 'gamma': 0.01, 'kernel': 'rbf'} 0.6996190476190476`

Random Forest	n_estimators, criterion, max_depth, max_features	Accuracy = 74%

	param_grid = {
	'n_estimators': [200, 300, 500],
	'criterion': ['entropy'],
	'max_depth': [8,9],
	'max_features': ['log2','sqrt']
	}
		print(fit2.best_estimator_,fit2.best_params_,fit2.best_score_)
		RandomForestClassifier(criterion='entropy', max_depth=9, max_features='log2',
		n_estimators=500, random_state=1) {'criterion': 'entropy', 'max_depth': 9, 'max_features': 'log2', 'n_estimators': 500} 0.7364761904761905

XG Boost	min_child_weight, gamma,
	colsample_bytree, max_depth
	
	params = {
	'min_child_weight': [10, 20],
	'gamma': [1.5, 2.0, 2.5],
	'colsample_bytree': [0.6, 0.8, 0.9],
	'max_depth': [4, 5, 6]
	}
		Accuracy = 72%
		
		print(fit3.best_estimator_,fit3.best_params_,fit3.best_score_)
		XGBClassifier(base_score=None, booster=None, callbacks=None,
		colsample_bylevel=None, colsample_bynode=None,
		colsample_bytree=0.6, device=None, early_stopping_rounds=None,
		enable_categorical=False, eval_metric=None, feature_types=None,
		gamma=2.5, grow_policy=None, importance_type=None,
		interaction_constraints=None, learning_rate=0.5, max_bin=None,
		max_cat_threshold=None, max_cat_to_onehot=None,
		max_delta_step=None, max_depth=5, max_leaves=None,
		min_child_weight=10, missing=nan, monotone_constraints=None,
		multi_strategy=None, n_estimators=100, n_jobs=None, nthread=3,
		num_parallel_tree=None, ...) {'colsample_bytree': 0.6, 'gamma': 2.5, 'max_depth': 5, 'min_child_weight': 10} 0.7291428571428572

Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric	Optimized Metric
Logistic Regression	ypred=lr.predict(x_test) print(classification_report(y_test,ypred)) precision recall f1-score support 0 0.66 0.73 0.69 1321 1 0.69 0.62 0.65 1305 accuracy 0.67 2626 macro avg 0.68 0.67 0.67 2626 weighted avg 0.68 0.67 0.67 2626 [27]: print(confusion_matrix(y_test,ypred)) [[966 355] [500 805]]	Not done Hyper parametric Tuning, Selected best 3 for tuning

<table>
<tr>
<td>Decision Tree</td>
<td>

```
ypred2=dt.predict(x_test)
print(classification_report(y_test,ypred2))
```

```
              precision    recall  f1-score   support

           0       0.71      0.69      0.70      1321
           1       0.69      0.71      0.70      1305

    accuracy                           0.70      2626
   macro avg       0.70      0.70      0.70      2626
weighted avg       0.70      0.70      0.70      2626
```

[33]:

```
print(confusion_matrix(y_test,ypred2))
```

```
[[910 411]
 [377 928]]
```

</td>
<td>Not done Hyper parametric Tuning,

Selected best 3 for tuning</td>
</tr>
<tr>
<td>KNN</td>
<td>

```
ypred3=knn.predict(x_test)
print(classification_report(y_test,ypred3))
```

```
              precision    recall  f1-score   support

           0       0.68      0.78      0.72      1321
           1       0.74      0.63      0.68      1305

    accuracy                           0.70      2626
   macro avg       0.71      0.70      0.70      2626
weighted avg       0.71      0.70      0.70      2626
```

[36]:

```
print(confusion_matrix(y_test,ypred3))
```

```
[[1028  293]
 [ 487  818]]
```

</td>
<td>Not done Hyper parametric Tuning,

Selected best 3 for tuning</td>
</tr>
<tr>
<td>SVM</td>
<td>

[38]:

```
ypred4=model.predict(x_test)
print(classification_report(y_test,ypred4))
```

```
              precision    recall  f1-score   support

           0       0.66      0.94      0.77      1321
           1       0.89      0.51      0.65      1305

    accuracy                           0.73      2626
   macro avg       0.77      0.72      0.71      2626
weighted avg       0.77      0.73      0.71      2626
```

[39]:

```
print(confusion_matrix(y_test,ypred4))
```

```
[[1238   83]
 [ 638  667]]
```

</td>
<td>

```
print(fit1.best_estimator_,fit1.best_params_,fit1.best_score_)
```

```
SVC(C=0.1, gamma=0.01) {'C': 0.1, 'gamma': 0.01, 'kernel': 'rbf'} 0.6996190476190476
```

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

Random forest	ypred1=rf.predict(x_test) print(classification_report(y_test,ypred1)) precision recall f1-score support 0 0.69 0.86 0.77 1321 1 0.81 0.62 0.70 1305 accuracy 0.74 2626 macro avg 0.75 0.74 0.73 2626 weighted avg 0.75 0.74 0.73 2626 [30]: print(confusion_matrix(y_test,ypred1)) [[1130 191] [501 804]]	print(fit2.best_estimator_,fit2.best_params_,fit2.best_score_) RandomForestClassifier(criterion='entropy', max_depth=9, max_features='log2', n_estimators=500, random_state=1) {'criterion': 'entropy', 'max_depth': 9, 'max_features': 'log2', 'n_estimators': 500} 0.7364761904761905
XG boost	[41]: ypred5=xg.predict(x_test) print(classification_report(y_test,ypred5)) precision recall f1-score support 0 0.70 0.80 0.75 1321 1 0.76 0.66 0.70 1305 accuracy 0.73 2626 macro avg 0.73 0.73 0.73 2626 weighted avg 0.73 0.73 0.73 2626 [42]: print(confusion_matrix(y_test,ypred5)) [[1052 269] [449 856]]	print(fit3.best_estimator_,fit3.best_params_,fit3.best_score_) XGBClassifier(base_score=None, booster=None, callbacks=None, colsample_bylevel=None, colsample_bynode=None, colsample_bytree=0.6, device=None, early_stopping_rounds=None, enable_categorical=False, eval_metric=None, feature_types=None, gamma=2.5, grow_policy=None, importance_type=None, interaction_constraints=None, learning_rate=0.5, max_bin=None, max_cat_threshold=None, max_cat_to_onehot=None, max_delta_step=None, max_depth=5, max_leaves=None, min_child_weight=10, missing=nan, monotone_constraints=None, multi_strategy=None, n_estimators=100, n_jobs=None, nthread=3, num_parallel_tree=None, ...) {'colsample_bytree': 0.6, 'gamma': 2.5, 'max_depth': 5, 'min_child_weight': 10} 0.7291428571428572

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random Forest	This model was chosen for its superior performance, achieving an accuracy of 74%, the highest among all evaluated models. Random Forest is renowned for its robustness and ability to handle large datasets with high dimensionality. It operates by constructing multiple decision trees during training and outputting the mode of the classes for classification tasks, ensuring improved accuracy and reduced overfitting.