Model Optimization and Tuning Phase Template

Date	15 March 2024
Team ID	SWTID1720437019
Project Title	Thyroid Classification
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 Model	[39]: `from sklearn.ensemble import RandomForestClassifier` `RFclassifier = RandomForestClassifier(max_leaf_nodes=30)` `RFclassifier.fit(x_train, y_train)` [42]: `from sklearn.model_selection import GridSearchCV` `rf = {` `'n_estimators': [100, 200, 300],` `'max_depth': [3, 5, 7, 10, None]` `}` `RFclassifier = RandomForestClassifier(random_state=42)` `grid_rf = GridSearchCV(RFclassifier, rf, cv=5)` `grid_rf.fit(x_train, y_train)` `print("Best parameters for Random Forest:", grid_rf.best_params_)` `y_pred = grid_rf.predict(x_test)` `print(classification_report(y_test, y_pred))`	[44]: `RFAcc = accuracy_score(y_pred,y_test)` `print('Random Forest accuracy is: {:.2f}%'.format(RFAcc*100))` Random Forest accuracy is: 94.20%

SVC Model	[45]: ```from sklearn.svm import SVC SVCclassifier = SVC(kernel='linear', max_iter=251) SVCclassifier.fit(x_train, y_train)```	[50]: ```SVCAcc = accuracy_score(y_pred,y_test) print('SVC accuracy is: {:.2f}%'.format(SVCAcc*100))``` SVC accuracy is: 86.61%
	[46]: ```svc_params = { 'kernel': ['linear', 'poly', 'rbf', 'sigmoid'], 'C': [1, 10, 100], 'gamma': ['scale', 'auto'] } grid_svc = GridSearchCV(SVC(), svc_params, cv=5) grid_svc.fit(x_train, y_train) print("Best parameters for SVC:", grid_svc.best_params_)```	
XGB Classifier Model	[52]: ```from xgboost import XGBClassifier from sklearn.preprocessing import LabelEncoder le = LabelEncoder() y_train_encoded = le.fit_transform(y_train) xgb = XGBClassifier() xgb.fit(x_train, y_train_encoded)``` [55]: ```from sklearn.model_selection import GridSearchCV param_grid = { 'learning_rate': [0.01, 0.1, 0.2], 'max_depth': [3, 5, 7], 'n_estimators': [100, 200, 300] } grid_search = GridSearchCV(xgb, param_grid, cv=5) grid_search.fit(x_train, y_train_encoded) print("Best parameters for XGBClassifier:", grid_search.best_params_) y_pred = grid_search.best_estimator_.predict(x_test) print(classification_report(y_test_encoded, y_pred))```	[57]: ```XGBAcc = accuracy_score(y_test_encoded, y_pred) print('XGB accuracy is: {:.2f}%'.format(XGBAcc*100))``` XGB accuracy is: 95.54%

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric

Random Forest Model

```
[40]: y_pred = RFclassifier.predict(x_test)

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

```
                precision    recall  f1-score   support

             0       0.00      0.00      0.00         7
             1       0.86      0.86      0.86        74
             2       0.90      1.00      0.95        85
             3       0.86      0.82      0.84        38
             4       0.95      1.00      0.97       122
             5       0.92      0.86      0.89        51
             6       0.99      0.93      0.96        71

      accuracy                           0.92       448
     macro avg       0.78      0.78      0.78       448
  weighted avg       0.91      0.92      0.91       448

[[  0   0   0   0   6   1   0]
 [  0  64   5   3   1   1   0]
 [  0   0  85   0   0   0   0]
 [  0   4   1  31   0   1   1]
 [  0   0   0   0 122   0   0]
 [  0   4   1   2   0  44   0]
 [  0   2   2   0   0   1  66]]
```

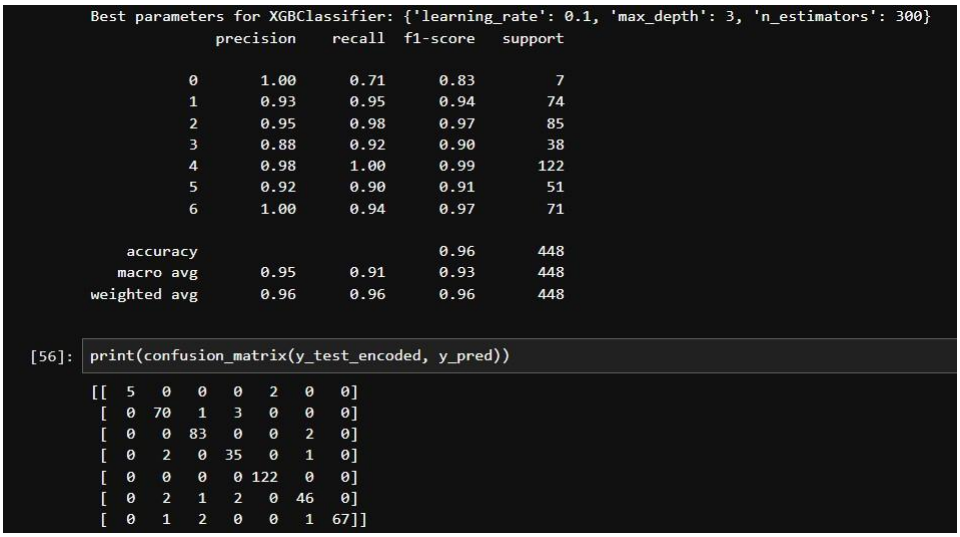
SVC Model

```
[47]: y_pred = SVCclassifier.predict(x_test)
      print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.67	0.86	0.75	7
1	0.79	0.80	0.79	74
2	0.83	0.74	0.78	85
3	0.73	0.58	0.65	38
4	0.89	0.95	0.92	122
5	0.76	0.75	0.75	51
6	0.87	0.96	0.91	71
accuracy			0.83	448
macro avg	0.79	0.80	0.79	448
weighted avg	0.83	0.83	0.83	448

```
[48]: print(confusion_matrix(y_test, y_pred))
```

```
[[  6   0   0   0   1   0   0]
 [  1  59   7   3   2   2   0]
 [  1   4  63   0   9   3   5]
 [  0   6   2  22   0   7   1]
 [  1   1   0   0 116   0   4]
 [  0   4   3   4   2  38   0]
 [  0   1   1   1   0   0  68]]
```

XGB Classifier	

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
XGB classifier Model	The XGB classifier model was selected for its superior performance, exhibiting high accuracy during hyperparameter tuning. Its ability to handle complex relationships, minimize overfitting, and optimize predictive accuracy aligns with project objectives, justifying its selection as the final model.