Model Optimization and Tuning Phase Template

Date	July 2024
Team ID	738691
Project Title	Frappe Activity: mobile Phone Activity classification
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining neural network 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 (8 Marks):

Model	Tuned Hyperparameters
Bagging Classifier	The (params) define a grid for hyperparameter tuning of the Bagging Classifier (BClassifier), including min_child_weight, gamma, colsample_bytree, and max_depth. The Bagging Classifier is configured with a learning rate of 0.5, 100 estimators, using a binary logistic regression objective, and utilizing 3 threads for processing. GridSearchCV (xg_cv) is used with 5-fold cross-validation (cv=5), refitting the best model (refit=True), evaluating based on accuracy (scoring="accuracy")

```
# Define the hyperparameters and their possible values for tuning

param_grid = {

    'n_estimators': [10, 50, 100],
    'max_samples': [0.5, 0.7, 1.0],
    'max_features': [0.5, 0.7, 1.0],
    'bootstrap': [True, False],
    'bootstrap_features': [True, False]
}
```

```
random_search = RandomizedSearchCV(estimator=bagging_classifier, param_distributions=param_grid,
                                   scoring='accuracy', cv=2, random_state=42)
```

```
random_search.fit(X_train,y_train)
```

```
[46]
    print("Best Parameters:",random_search.best_params_)
    print("Best Score:",random_search.best_score_)
```

```
...  Best Parameters: {'n_estimators': 100, 'max_samples': 0.7, 'max_features': 1.0, 'bootstrap_features': True, 'bootstrap': False}
     Best Score: 0.6545086119554204
```

```
[47]
    print("Best Score:",random_search.score(X_test,y_test))
```

```
...  Best Score: 0.6861702127659575
```

Decision Tree	The parameters (params) define a grid for hyperparameter tuning of the Decision Tree Classifier (DecisionTreeClassifier), including max_depth, min_samples_leaf, and criterion ('gini' or 'entropy'). GridSearchCV (dec_cv) is used with 5-fold cross-validation (cv=5), evaluating model performance based on accuracy (scoring="accuracy")

```
# Define the hyperparameters and their possible values for tuning
param_grid = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
    'max_depth': [None, 2, 4, 6, 8,10],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': [None, 'sqrt', 'log2'],
    'min_impurity_decrease': [0.0, 0.1, 0.2],
    'ccp_alpha': [0.0, 0.1 ,0.2]
}
```

```
# Initialize RandomizedSearchCV with DecisionTreeClassifier
random_search = RandomizedSearchCV(estimator=dt_classifier,
                                   param_distributions=param_grid,
                                   scoring='accuracy',
                                   cv=3,
                                   n_iter=100,
                                   random_state=42)
```

```
random_search.fit(X_train, y_train)
RandomizedSearchCV(cv=3,estimator=DecisionTreeClassifier(),n_iter=100,
                   param_distributions={'ccp_alpha':[0.0, 0.1, 0.2],
                                        'criterion':['gini','entropy'],
                                        'max_depth':[None, 2, 4, 6, 8, 10],
                                        'max_features':[None,'sqrt','log2'],
                                        'min_impurity_decrease':[0.0, 0.1, 0.2],
                                        'min_samples_leaf':[1, 2, 4],
                                        'min_samples_split':[2, 5, 10],
                                        'splitter': ['best', 'random']},
random_state=42, scoring='accuracy')

print("Best Parameters:", random_search.best_params_)

print("Best Score:", random_search.best_score_)
```

Final Model Selection Justification (2 Marks):

Final Model	Reasoning

Bagging Classifier	Bagging Classifier model is chosen for its robustness in handling complex datasets and its ability to mitigate overfitting while providing high predictive accuracy.
	
	`print(classification_report(y_test,y_pred,digits=4))`
	
	`precision recall f1-score support`
	
	`0 0.5500 0.6396 0.5914 60466`
	`1 0.6052 0.4767 0.5333 60427`
	`2 0.6683 0.7010 0.6843 60715`
	
	`accuracy 0.6060 181608`
	`macro avg 0.6078 0.6058 0.6030 181608`
	`weighted avg 0.6079 0.6060 0.6031 181608`
	
	Above all the models Bagging classifier have the highest accuracy among all the models.