

Model Optimization and Tuning Phase Template

Date	15 March 2024
Team ID	XXXXXX
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
Decision Tree	<pre># Function to train and evaluate a Decision Tree model with hyperparameter tuning def decisionTree(X_train, X_test, y_train, y_test): # Define the parameter grid param_grid = { 'max_depth': [None, 10, 20, 30, 40, 50], 'min_samples_split': [2, 10, 20], 'min_samples_leaf': [1, 5, 10], 'criterion': ['gini', 'entropy'] } </pre>	<p>Best Parameters found by GridSearchCV:</p> <pre>{'criterion': 'entropy', 'max_depth': 40, 'min_samples_leaf': 1, 'min_samples_split': 2}</pre> <p>Accuracy: 0.94</p>
Random Forest	<pre># Function to train and evaluate a Random Forest model def randomForest(X_train, X_test, y_train, y_test): # Define the parameter grid param_grid = { 'n_estimators': [100, 200, 300], 'max_depth': [None, 10, 20, 30], 'min_samples_split': [2, 5, 10], 'min_samples_leaf': [1, 2, 4], 'bootstrap': [True, False] } </pre>	<p>Best Parameters found by GridSearchCV:</p> <pre>{'bootstrap': False, 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 100}</pre> <p>Accuracy: 0.96</p>

KNN	<pre># Function to train and evaluate a KNN model with hyperparameter tuning def KNN(X_train, X_test, y_train, y_test): # Define the parameter grid param_grid = { 'n_neighbors': [3, 5, 7, 9, 11], 'weights': ['uniform', 'distance'], 'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'], 'p': [1, 2] } # ... (rest of the function code) ...</pre>	<p>Best Parameters found by GridSearchCV: {'algorithm': 'ball_tree', 'n_neighbors': 3, 'p': 1, 'weights': 'distance'}</p> <p>Accuracy: 0.93</p>
Gradient Boost	<pre># Function to train and evaluate a Gradient Boosting model with hyperparameter tuning def xgboost(X_train, X_test, y_train, y_test): # Define the parameter grid param_grid = { 'n_estimators': [100, 200, 300], 'learning_rate': [0.01, 0.1, 0.2], 'max_depth': [3, 4, 5], 'subsample': [0.8, 0.9, 1.0], 'min_samples_split': [2, 5, 10] } # ... (rest of the function code) ...</pre>	<p>Accuracy: 0.93</p> <p>Best Parameters found by GridSearchCV: {'learning_rate': 0.2, 'max_depth': 5, 'min_samples_split': 5, 'n_estimators': 300, 'subsample': 0.8}</p>

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric
Decision Tree	<pre>Confusion Matrix: [[8668 612] [453 8809]] Classification Report: precision recall f1-score support 0 0.95 0.93 0.94 9280 1 0.94 0.95 0.94 9262 accuracy 0.94 macro avg 0.94 weighted avg 0.94 Accuracy: 0.94</pre> <p>... DecisionTreeClassifier</p> <pre>DecisionTreeClassifier(criterion='entropy', max_depth=40, random_state=42)</pre>

Gradient Boost	<pre> Confusion Matrix: [[8678 602] [695 8567]] Classification Report: precision recall f1-score support 0 0.93 0.94 0.93 9280 1 0.93 0.92 0.93 9262 accuracy 0.93 0.93 18542 macro avg 0.93 0.93 0.93 18542 weighted avg 0.93 0.93 0.93 18542 Accuracy: 0.93 ... GradientBoostingClassifier GradientBoostingClassifier(learning_rate=0.2, max_depth=5, min_samples_split=5, n_estimators=300, random_state=42, subsample=0.8) </pre>
KNN	<pre> Confusion Matrix: [[8294 986] [222 9040]] Classification Report: precision recall f1-score support 0 0.97 0.89 0.93 9280 1 0.90 0.98 0.94 9262 accuracy 0.93 0.93 18542 macro avg 0.94 0.93 0.93 18542 weighted avg 0.94 0.93 0.93 18542 Accuracy: 0.93 ... KNeighborsClassifier KNeighborsClassifier(algorithm='ball_tree', n_neighbors=3, p=1, weights='distance') </pre>
Random Forest	<pre> Confusion Matrix: [[8959 321] [421 8841]] Classification Report: precision recall f1-score support 0 0.96 0.97 0.96 9280 1 0.96 0.95 0.96 9262 accuracy 0.96 0.96 18542 macro avg 0.96 0.96 0.96 18542 weighted avg 0.96 0.96 0.96 18542 Accuracy: 0.96 ... RandomForestClassifier RandomForestClassifier(bootstrap=False, min_samples_split=5, random_state=42) </pre>

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random Forest	I chose the Random Forest model as the final model for predicting employee promotions due to its superior accuracy (96%) compared to other models like Decision Tree, KNN, and Gradient Boosting. Random Forest is robust, handles overfitting well, and provides insights into feature importance. It captures complex, non-linear relationships within the data and is scalable for large datasets. Additionally, hyperparameter tuning further optimized its performance, making it a reliable and efficient choice for this task