

Model Optimization and Tuning Phase Report

Date	25 June 2025
Team ID	SWUID20250176209
Project Title	Machine Learning Approach for Employee Performance Prediction
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
Linear Regression	<pre># Linear Regression: from sklearn.linear_model import LinearRegression from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score # Initialize Linear Regression model: model_lr = LinearRegression() # Train the model: model_lr.fit(X_train, y_train) # Predict on the test set: pred_test = model_lr.predict(X_test) # Model Evaluation: mae_lr = mean_absolute_error(y_test, pred_test) mse_lr = mean_squared_error(y_test, pred_test) r2_lr = r2_score(y_test, pred_test) # Results: print("Linear Regression Evaluation:") print(f"Mean Absolute Error (MAE): {mae_lr:.4f}") print(f"Mean Squared Error (MSE): {mse_lr:.4f}") print(f"R² Score: {r2_lr:.4f}")</pre>	<pre>===== Linear Regression Evaluation: Mean Absolute Error (MAE): 0.1040 Mean Squared Error (MSE): 0.0205 R² Score: 0.3025 =====</pre>
Random Forest	<pre># Random Forest: from sklearn.ensemble import RandomForestRegressor # Initialize Random Forest model: model_rf = RandomForestRegressor() # Train the model: model_rf.fit(X_train, y_train) # Predict on the test set: pred = model_rf.predict(X_test) # Model Evaluation: mae_rf = mean_absolute_error(y_test, pred) mse_rf = mean_squared_error(y_test, pred) r2_rf = r2_score(y_test, pred) # Results: print("Random Forest Evaluation:") print(f"Mean Absolute Error (MAE): {mae_rf:.4f}") print(f"Mean Squared Error (MSE): {mse_rf:.4f}") print(f"R² Score: {r2_rf:.4f}")</pre>	<pre>===== Random Forest Evaluation: Mean Absolute Error (MAE): 0.0681 Mean Squared Error (MSE): 0.0118 R² Score: 0.6003 =====</pre>

XGBoost	<pre># XGBoost: from xgboost import XGBRegressor # Initialize XGBoost Regressor: model_xgb = XGBRegressor() # Train the model: model_xgb.fit(X_train, y_train) # Predict on the test set: pred3 = model_xgb.predict(X_test) # Model Evaluation: mae_xgb = mean_absolute_error(y_test, pred3) mse_xgb = mean_squared_error(y_test, pred3) r2_xgb = r2_score(y_test, pred3) # Results: print("XGBoost Evaluation:") print(f"Mean Absolute Error (MAE): {mae_xgb:.4f}") print(f"Mean Squared Error (MSE): {mse_xgb:.4f}") print(f"R² Score: {r2_xgb:.4f}")</pre>	<pre>XGBoost Evaluation: Mean Absolute Error (MAE): 0.0673 Mean Squared Error (MSE): 0.0110 R² Score: 0.6264</pre>
---------	--	--

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric
XGBoost	<pre>Model Performance Comparison: Model MAE MSE R² Score 0 Linear Regression 0.103975 0.020518 0.302503 1 Random Forest 0.068143 0.011757 0.600309 2 XGBoost 0.067346 0.010990 0.626399</pre>

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
XGBoost	<p>The XGBoost Regressor was selected as the final model due to its high R² score of 0.62 (62%), low mean absolute error, and strong performance on imbalanced and noisy data. It handled both categorical and numerical features effectively and required minimal preprocessing. Its ability to capture complex, non-linear patterns in garment worker productivity made it a reliable and efficient choice that aligned well with the project's objectives.</p>