



## **Model Optimization and Tuning Phase Template**

Date	10 July 2024
Team ID	739842
Project Title	Trip-Based Modelling of Fuel Consumption in Modern Fleet Vehicles 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.

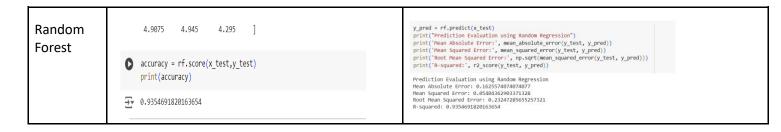
**Hyperparameter Tuning Documentation (6 Marks):** 

Model	Tuned Hyperparameters	Optimal Values
Linear Regression	No Hyperparameters used	
Lasso Regression	No Hyperparameters used	
SVM	No Hyperparameters used	
Decision Tree	No Hyperparameters used	
Random Forest	No Hyperparameters used	

Model	Accuracy	Metrics
Linear Regression	print(linReg.coef_,linReg.intercept_) [ 0.00523674 -0.02371772 -0.14711979 -0.03724498	print("Prediction Evaluation using Linear Regression") print('Mean Absolute Error:', mean absolute error(y_test, y_pred)) print('Rean Squared Error:', mean squared error(y_test, y_pred)) print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred))) print('R-squared:', r2_score(y_test, y_pred))  Prediction Evaluation using Linear Regression Mean Absolute Error: 0.663576182069623 Mean Squared Error: 0.742453260904708 Root Mean Squared Error: 0.8616572757800562 R-squared: 0.1134733714697449
Lasso Regression	<pre>accuracy = lassoReg.score(x_test,y_test) print(accuracy) 0.1456141532515728</pre>	<pre> y_pred = lassoReg.predict(x_test) print("Prediction Evaluation using lasso Regression") print("Mean Absolute Error: ', mean_absolute_error(y_test, y_pred)) print('Mean Squared Error: ', mean_squared error(y_test, y_pred)) print('Root Mean Squared Error: ', np.sqrt(mean_squared_error(y_test, y_pred))) print('R-squared: ', r2_score(y_test, y_pred))  Prediction Evaluation using lasso Regression Mean Absolute Error: 0.6296444264267669 Mean Squared Error: 0.7155358198781405 Root Mean Squared Error: 0.8458935638633058 R-squared: 0.1456141532515728 </pre>
SVM	[43] #SVM MODEL	a)  y_pred = svr.predict(x_test) print("Prediction Evaluation using svr Regression") print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred)) print('Mean Squared Error:', mean_squared_error(y_test, y_pred)) print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred))) print('R-squared:', r2_score(y_test, y_pred))  Prediction Evaluation using svr Regression Mean Absolute Error: 0.48771357102448615 Root Mean Squared Error: 0.698364926828722 R-squared: 0.4176454053391483
Decision Tree	# (80) db = DecisionTreategressor(random_State = 0) db = DecisionTreategressor(random_State = 0) db = DecisionTreategressor(random_State = 0) dc = Decision	y_pred = dt.predict(x_test) print("Prediction Evaluation using decisiontree Regression") print("Mean Absolute Error:', mean_absolute_error(y_test, y_pred)) print("Mean Squared Error:', mean_squared_error(y_test, y_pred)) print("Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred))) print("Rost Respured:', r2_score(y_test, y_pred))  Prediction Evaluation using decisiontree Regression Mean Absolute Error: 0.10560560560560560 Mean Squared Error: 0.10561832633943249 R-squared: 0.9864521202267205







## **Performance Metrics Comparison Report (2 Marks):**

## Final Model Selection Justification (2 Marks):

Final Model Selection	Reasoning
Decision Tree	Decision Tree model was selected for its superior performance, exhibiting high accuracy than any other models .  We chose the decision tree because it gives very accurate predictions, can handle complex patterns in data, and avoids overfitting. It works well with different types of data and allows us to see which features are most important. This makes it a reliable and effective model for our task