Model Optimization and Tuning Phase Template

Date	8 JULY 2024
Team ID	SWTID1720000556
Project Title	Predicting Co2 Emission By Countries 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)

<table>
<tr><th>Model</th><th>Tuned Hyperparameters</th><th>Optimal Values</th></tr>
<tr><td>Linear Regression</td><td>--NO,Hyper Parameter Used--</td><td>--NA--</td></tr>
<tr><td>Ridge Regression</td><td>Ridge Regression
<pre>
ridge_reg = Ridge()

x=np.logspace(-4, 4, 50)
param_distributions = {'alpha': x}

ridge_random = RandomizedSearchCV(ridge_reg, param_distributions, n_iter=10,random_state=42)
ridge_random.fit(x_train, y_train)

RandomizedSearchCV
best_estimator_: Ridge
Ridge
</pre></td><td><pre>
ridgebest_alpha = ridge_random.best_params_
best_score = ridge_random.best_score_
print(best_score)
print(ridgebest_alpha)

0.0002386836157977523
{'alpha': 6866.488450042998}
</pre></td></tr>
</table>

Decision Tree Regressor	`tree_reg = DecisionTreeRegressor()`
	`param_distributions = {`
	`'max_depth': randint(1, 20),`
	`'min_samples_split': randint(2, 20),`
	`'min_samples_leaf': randint(1, 20),`
	`'max_features': ['auto', 'sqrt', 'log2', None]`
	`}`
	
	`tree_random = RandomizedSearchCV(tree_reg, param_distributions, n_iter=100, cv=5, random_state=42, n_jobs=-1)`
	`tree_random.fit(x_train, y_train)`
	
	`RandomizedSearchCV`
	`best_estimator_: DecisionTreeRegressor`
	`DecisionTreeRegressor`
		`best_params = tree_random.best_params_`
		`best_score = tree_random.best_score_`
		`print(best_params)`
		`print(best_score)`
		`{'max_depth': 19, 'max_features': None, 'min_samples_leaf': 5, 'min_samples_split': 10}`
		`0.723929434630653 7`

Random Forest Regressor	-- NO HyperParameter used--	--NA--
XGBoost Regression	--No HyperParameter Used--	--NA--

Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric	Optimized Metric

Linear Regression	`mse_lin_reg = mean_squared_error(y_test, y_pred_lin_reg)`
	`print(f'Linear Regression MSE: {mse_lin_reg}')`
	`lin_reg.score(x_test,y_test)`
	`Linear Regression MSE: 2.3664620738212395e+27`
		`mse_lin_reg = mean_squared_error(y_test, y_pred_lin_reg)`
		`print(f'Linear Regression MSE: {mse_lin_reg}')`
		`lin_reg.score(x_test,y_test)`
		`Linear Regression MSE: 2.3664620738212395e+27`
		`0.0002686736146634283`
		
		`print(lin_reg.score(x_train,y_train))`
		`print(f"mse for train data set", mean_squared_error(y_train,two_pred_lin_reg))`
		`0.00024196666451214988`
		`mse for train data set 2.3388476124039764e+27`
		
		The test data dosen't fit in either test data or train data using Linear Regression, Both scores are almost the same, showing that the model performs equally poorly on both the training and test sets, which is characteristic of **underfitting.**

Ridge Regression	`ridge_random.score(x_test,y_test)`
	`0.0002686734707454397`
	`y_pred3=ridge_random.predict(x_test)`
	`mse5=mean_squared_error(y_test,y_pred3)`
	`mse5`
	`2.3664620741619075e+27`
		`ridgebest_alpha = ridge_random.best_params_`
		`best_score = ridge_random.best_score_`
		`print(best_score)`
		`print(ridgebest_alpha)`
		`0.0002386836157977523`
		`{'alpha': 6866.488450042998}`
		`ridge_random.score(x_test,y_test)`
		`0.0002686734707454397`
		`y_pred3=ridge_random.predict(x_test)`
		`mse5=mean_squared_error(y_test,y_pred3)`
		`mse5`
		The test data doesn’t fit in either test data or train data using Ridge Regression, Both scores are almost the same, showing that the model performs equally poorly on both the training and test sets, which is characteristic of **underfitting.**

Decision Tree Regressor	`tree_random.score(x_test,y_test)`
	`0.7593560151530763`
	`mse1=mean_squared_error(y_test,y_prediction)`
	`mse1`
	`5.696279074223585e+26`
		`tree_random.score(x_test,y_test)`
		`0.7593560151530763`
		`mse1=mean_squared_error(test_pred_tree,y_test)`
		`mse2=mean_squared_error(train_pred_tree,y_train)`
		`print(mse2) # train`
		`print(mse1) # test`
		`3.3566794752688856e+26`
		`5.696279074223585e+26`
		Overfitting is raised as mse of train data is less than that of test data

Random Forest Regressor	`y_pred_random =model.predict(x_test)`
	`mse_random = mean_squared_error(y_test, y_pred_random)`
	`print(f'Random Forest Regression MSE: {mse_random}')`
	`model.score(x_train,y_train)`
	`Random Forest Regression MSE: 1.7008807722525123e+26`
	`0.9830788955337789`
		`y_pred_random =model.predict(x_test)`
		`train_pred=model.predict(x_train)`
		`mse_random = mean_squared_error(y_test, y_pred_random)`
		`print(f'Random Forest Regression MSE: {mse_random}')`
		`model.score(x_train,y_train)`
		`Random Forest Regression MSE: 1.700880772252512e+26`
		`0.9830788955337789`
		`mse_random_train=mean_squared_error(y_train,train_pred)`
		`model.score(x_test,y_test)`
		`0.9281448957378994`

		Compared to test data,it gives Overfitting

XGBoost Regression	`y_pred_xgb = xgb_reg.predict(x_test)`
	
	`mse_xgb = mean_squared_error(y_test, y_pred_xgb)`
	`print(f'XGBoost Regression MSE: {mse_xgb}')`
	`xgb_reg.score(x_train,y_train)`
	`xgb_reg.score(x_test,y_test)`
	
	`XGBoost Regression MSE: 1.7127083409170892e+27`
	`0.2764523039190644`
		`mse_xgb = mean_squared_error(y_test, y_pred_xgb)`
		`print(f'XGBoost Regression MSE: {mse_xgb}')`
		`print(xgb_reg.score(x_train,y_train))`
		`print(xgb_reg.score(x_test,y_test))`
		
		`XGBoost Regression MSE: 1.7127083409170892e+27`
		`0.3248889392152211`
		`0.2764523039190644`
		
		The test data doesn't fit in either test data or train data using Ridge Regression, Both scores are almost the same, showing that the model performs equally poorly on both the training and test sets, which is characteristic of **underfitting.**

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random Forest Regressor	The Random Forest Regressor was chosen as the optimal model due to its superior performance in terms of Mean Squared Error (MSE) and R2 score when compared to other regression models.