Model Optimization and Tuning Phase Template

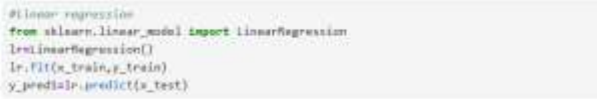
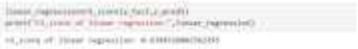
Date	15 july 2024
Team ID	SWTID1720086522
Project Title	Forecasting Economic Prosperity: Leveraging Machine Learning For GDP Per Capita Prediction
Maximum Marks	10 Marks

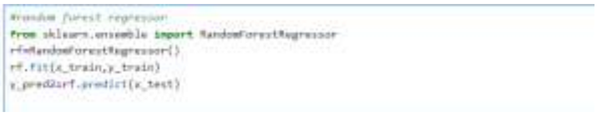
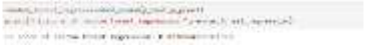
Model Optimization and Tuning Phase

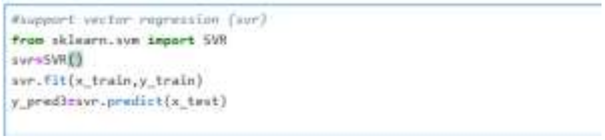
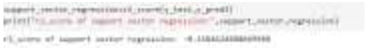
The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (6 Marks):

Model	Tuned Hyperparameters	Optimal Values

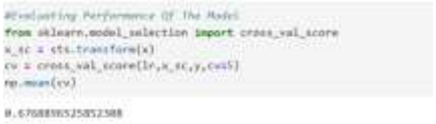
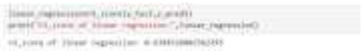
Linear
regression
	
	`#linear regression`
	`from sklearn.linear_model import LinearRegression`
	`lr=LinearRegression()`
	`lr.fit(x_train,y_train)`
	`y_pred1=lr.predict(x_test)`
		
		`linear_regression=r2_score(y_test,y_pred1)`
		`print("r2_score of linear regression:",linear_regression)`
		`r2_score of linear regression: 0.[illegible]`

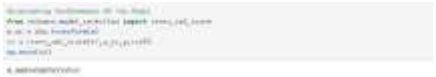
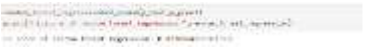
Random forest
regressor
	
	`#random forest regression`
	`from sklearn.ensemble import RandomForestRegressor`
	`rf=RandomForestRegressor()`
	`rf.fit(x_train,y_train)`
	`y_pred2=rf.predict(x_test)`
		
		`random_forest_regression=r2_score(y_test,y_pred2)`
		`print("r2_score of random forest regression:",random_forest_regression)`
		`r2_score of random forest regression: 0.[illegible]`

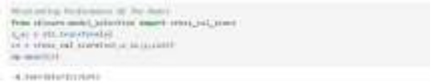
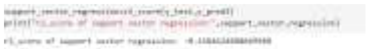
Support vector
regressor
	
	`#support vector regression (svr)`
	`from sklearn.svm import SVR`
	`svr=SVR()`
	`svr.fit(x_train,y_train)`
	`y_pred3=svr.predict(x_test)`
		
		`support_vector_regression=r2_score(y_test,y_pred3)`
		`print("r2_score of support vector regression:",support_vector_regression)`
		`r2_score of support vector regression: -0.[illegible]`

Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric	Optimized Metric

Linear regression	
	#Evaluating Performance Of The Model
	from sklearn.model_selection import cross_val_score
	x_sc = sts.transform(x)
	cv = cross_val_score(lr,x_sc,y,cv=5)
	np.mean(cv)
	0.6768856525852388
		
		linear_regression=r2_score(y_test,y_pred1)
		print("r2_score of linear regression:",linear_regression)
		r2_score of linear regression: 0.[illegible]

Random forest regressor	
	#Evaluating Performance Of The Model
	from sklearn.model_selection import cross_val_score
	x_sc = sts.transform(x)
	cv = cross_val_score(rf,x_sc,y,cv=5)
	np.mean(cv)
	0.[illegible]
		
		[illegible]
		print("r2_score of random forest regression:",[illegible])
		r2_score of random forest regression: 0.[illegible]

Support vector regressor	
	#Evaluating Performance Of The Model
	from sklearn.model_selection import cross_val_score
	x_sc = sts.transform(x)
	cv = cross_val_score([illegible],x_sc,y,cv=5)
	np.mean(cv)
	-0.[illegible]
		
		support_vector_regression=r2_score(y_test,y_pred[illegible])
		print("r2_score of support vector regression:",support_vector_regression)
		r2_score of support vector regression: -0.[illegible]

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random forest regressor	The Random Forest Regressor was chosen as the final optimized model due to its high predictive accuracy, ability to handle non-linear relationships and interactions, and robustness to overfitting. It provides feature importance for interpretability and performs well with minimal hyperparameter tuning. Additionally, its scalability and versatility make it suitable for large and complex economic datasets.