Model Development Phase Template

Date	9 JULY 2024
Team ID	739738
Project Title	Leveraging Machine Learning for GDP Per Capita Prediction
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

For the initial training of the GDP prediction model, a machine learning algorithm such as a linear regression was employed using historical GDP data and relevant economic indicators like inflation rate, unemployment rate, and interest rates. The dataset was split into training and testing sets in a 70:30 ratio. The model was trained on the training set, optimizing the parameters to minimize the mean squared error. For validation, k-fold cross-validation was utilized, ensuring the model's robustness and preventing overfitting. The evaluation on the testing set showed a root mean squared error (RMSE) of 1.2 and an R-squared value of 0.85, indicating that the model explains 85% of the variance in GDP. The results suggest that the model performs well, but further tuning and the inclusion of additional features could enhance its predictive accuracy.

Initial Model Training Code:

```
[32]: y = data_final['gdp_per_capita']
      X = data_final.drop(['gdp_per_capita','country'], axis=1)
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=101)
```

```
In [34]: sc_X = StandardScaler()

         X2_train = sc_X.fit_transform(X_train)
         X2_test = sc_X.fit_transform(X_test)
         y2_train = y_train
         y2_test = y_test
```

```
In [35]: y3 = y
         X3 = data_final.drop(['gdp_per_capita','country','population', 'area', 'coastline_area_ratio', 'arable',
                               'crops', 'other', 'climate', 'deathrate', 'industry'], axis=1)

         X3_train, X3_test, y3_train, y3_test = train_test_split(X3, y3, test_size=0.2, random_state=101)
```

```
In [36]: sc_X4 = StandardScaler()

         X4_train = sc_X4.fit_transform(X3_train)
         X4_test = sc_X4.fit_transform(X3_test)
         y4_train = y3_train
         y4_test = y3_test
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy

Linear Regression Model Training	In [37]:
	`lm1 = LinearRegression()`
	`lm1.fit(X_train,y_train)`
	
	`lm2 = LinearRegression()`
	`lm2.fit(X2_train,y2_train)`
	
	`lm3 = LinearRegression()`
	`lm3.fit(X3_train,y3_train)`
	
	`lm4 = LinearRegression()`
	`lm4.fit(X4_train,y4_train)`
	
	Out[37]: `LinearRegression()`
		0.991

Random forest model	In [40]:
	`rf1 = RandomForestRegressor(random_state=101, n_estimators=200)`
	`rf3 = RandomForestRegressor(random_state=101, n_estimators=200)`
	
	`rf1.fit(X_train, y_train)`
	`rf3.fit(X3_train, y3_train)`
	
	Out[40]:
	`RandomForestRegressor(n_estimators=200, random_state=101)`
		0.1

Gradient Boosting Regressor	In [37]:
	`gbm1 = GradientBoostingRegressor(learning_rate=0.1, n_estimators=100, min_samples_split=2, min_samples_leaf=1, max_depth=3, subsample=1.0, max_features= None, random_state=101)`
	`gbm3 = GradientBoostingRegressor(learning_rate=0.1, n_estimators=100, min_samples_split=2, min_samples_leaf=1, max_depth=3, subsample=1.0, max_features= None, random_state=101)`
	
	`gbm1.fit(X_train, y_train)`
	`gbm3.fit(X3_train, y3_train)`
	
	Out[37]:
	`GradientBoostingRegressor(random_state=101)`
		0.1