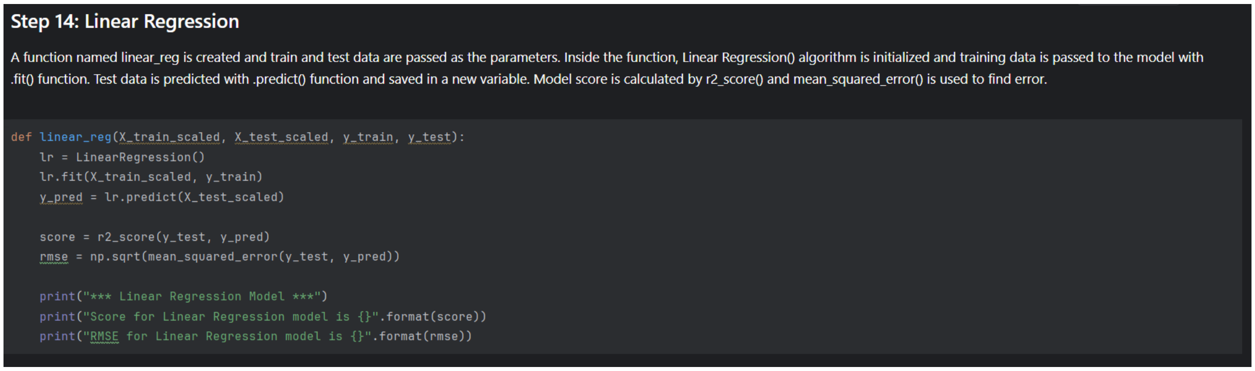
Model Development Phase Template

Date	18 June 2025
Team ID	SWTID1749653449
Project Title	Economic Growth: A Machine Learning Approach to GDP per Capita Prediction
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

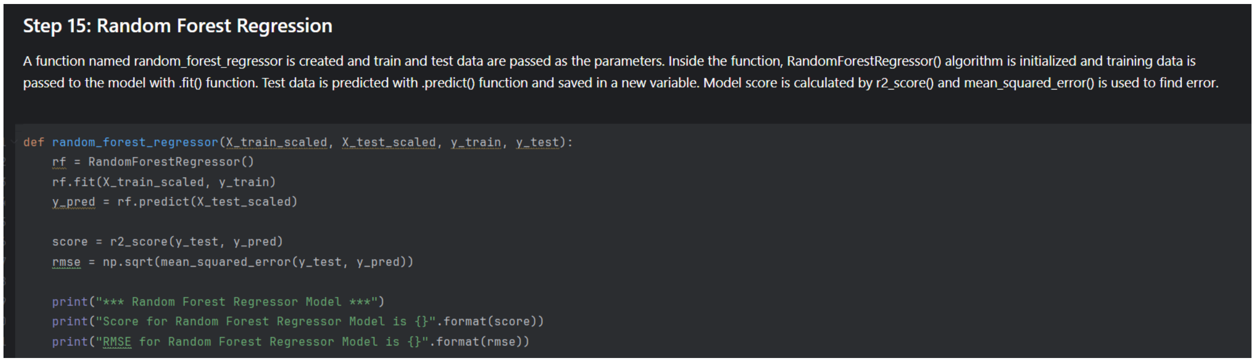
Initial Model Training Code:



Step 14: Linear Regression

A function named linear_reg is created and train and test data are passed as the parameters. Inside the function, Linear Regression() algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in a new variable. Model score is calculated by r2_score() and mean_squared_error() is used to find error.

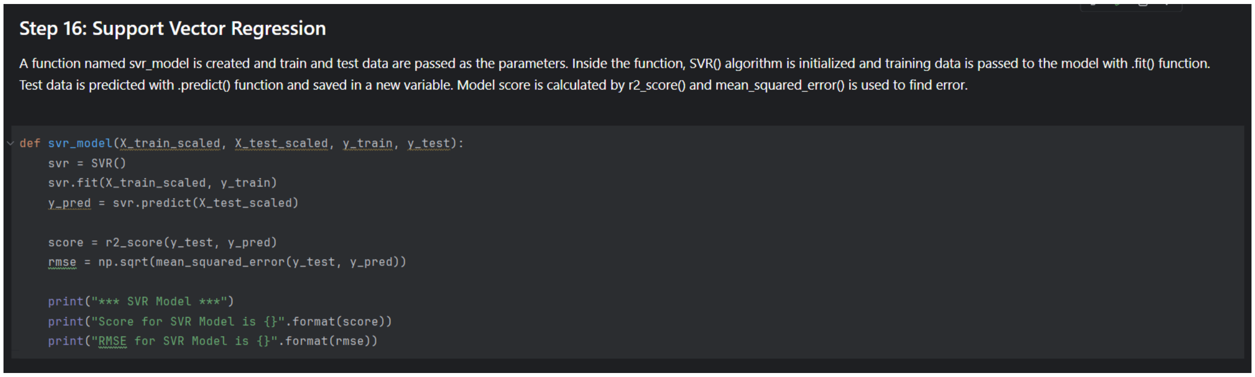
```python
def linear_reg(X_train_scaled, X_test_scaled, y_train, y_test):
    lr = LinearRegression()
    lr.fit(X_train_scaled, y_train)
    y_pred = lr.predict(X_test_scaled)

    score = r2_score(y_test, y_pred)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))

    print("*** Linear Regression Model ***")
    print("Score for Linear Regression model is {}".format(score))
    print("RMSE for Linear Regression model is {}".format(rmse))
```



Step 15: Random Forest Regression

A function named random_forest_regressor is created and train and test data are passed as the parameters. Inside the function, RandomForestRegressor() algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in a new variable. Model score is calculated by r2_score() and mean_squared_error() is used to find error.

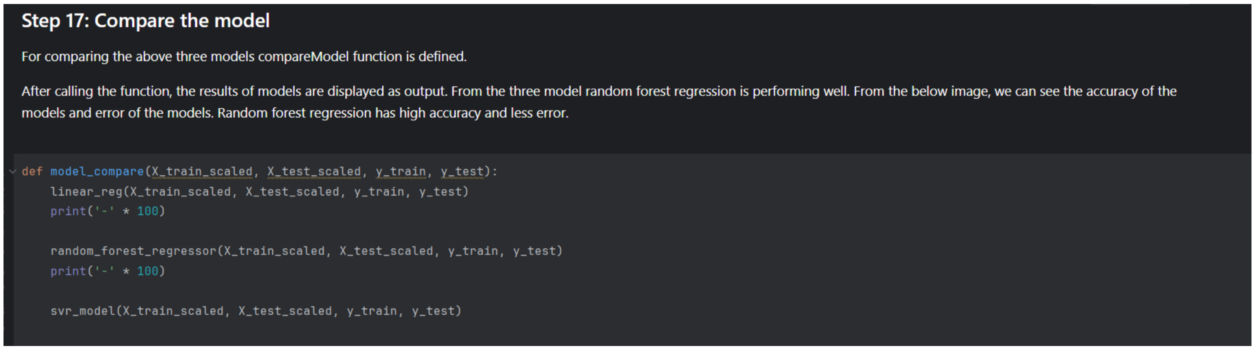
```python
def random_forest_regressor(X_train_scaled, X_test_scaled, y_train, y_test):
    rf = RandomForestRegressor()
    rf.fit(X_train_scaled, y_train)
    y_pred = rf.predict(X_test_scaled)

    score = r2_score(y_test, y_pred)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))

    print("*** Random Forest Regressor Model ***")
    print("Score for Random Forest Regressor Model is {}".format(score))
    print("RMSE for Random Forest Regressor Model is {}".format(rmse))
```



Step 16: Support Vector Regression

A function named svr_model is created and train and test data are passed as the parameters. Inside the function, SVR() algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in a new variable. Model score is calculated by r2_score() and mean_squared_error() is used to find error.

```
def svr_model(X_train_scaled, X_test_scaled, y_train, y_test):
    svr = SVR()
    svr.fit(X_train_scaled, y_train)
    y_pred = svr.predict(X_test_scaled)

    score = r2_score(y_test, y_pred)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))

    print("*** SVR Model ***")
    print("Score for SVR Model is {}".format(score))
    print("RMSE for SVR Model is {}".format(rmse))
```

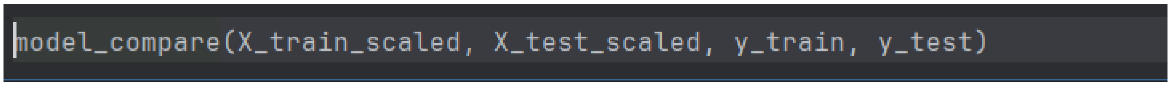


Step 17: Compare the model

For comparing the above three models compareModel function is defined.

After calling the function, the results of models are displayed as output. From the three model random forest regression is performing well. From the below image, we can see the accuracy of the models and error of the models. Random forest regression has high accuracy and less error.

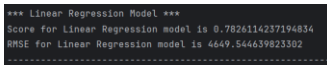
```
def model_compare(X_train_scaled, X_test_scaled, y_train, y_test):
    linear_reg(X_train_scaled, X_test_scaled, y_train, y_test)
    print('-' * 100)

    random_forest_regressor(X_train_scaled, X_test_scaled, y_train, y_test)
    print('-' * 100)

    svr_model(X_train_scaled, X_test_scaled, y_train, y_test)
```

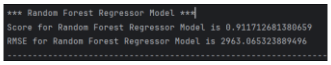


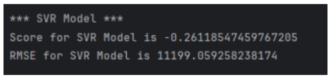
```
model_compare(X_train_scaled, X_test_scaled, y_train, y_test)
```

Model Validation and Evaluation Report:

Model	Regression Parameters Report	Accuracy	Confusion Matrix

Linear Regression	
	`*** Linear Regression Model ***`
	`Score for Linear Regression model is 0.7826114237194834`
	`RMSE for Linear Regression model is 4649.544639823302`
	`--------------------------------------------------------`
		+78 %
		RMSE: ~4650
			NIL (Not applicable for regression analysis)

Random Forest Regression	
	`*** Random Forest Regressor Model ***`
	`Score for Random Forest Regressor Model is 0.911712681380659`
	`RMSE for Random Forest Regressor Model is 2963.065323889496`
	`--------------------------------------------------------`
		+91 %
		RMSE: ~2964
			NIL (Not applicable for regression analysis)

Support Vector Regression	 *** SVR Model *** Score for SVR Model is -0.26118547459767205 RMSE for SVR Model is 11199.059258238174	-26 % RMSE: ~11120	NIL (Not applicable for regression analysis)