Model Optimization and Tuning Phase Report

Date	15 March 2024
Team ID	740012
Project Title	**Predicting IMF-Based Exchange Rates: Leveraging Economic Indicators for Accurate Regression Modeling**
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):

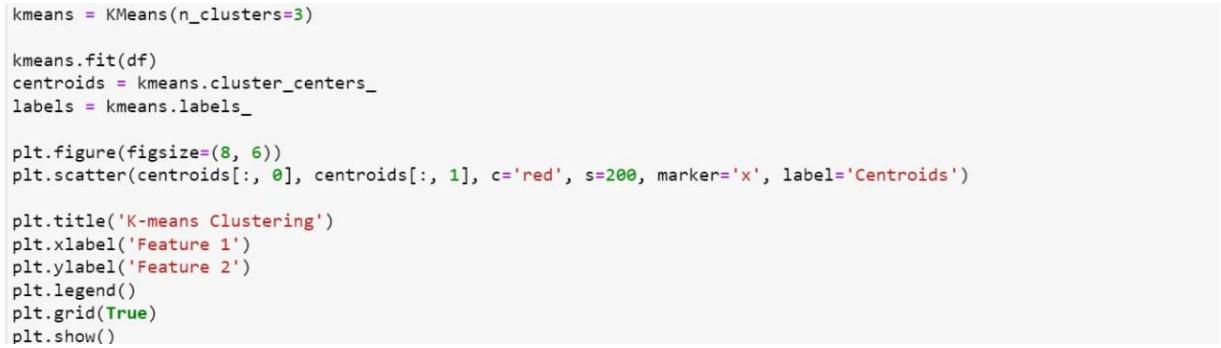
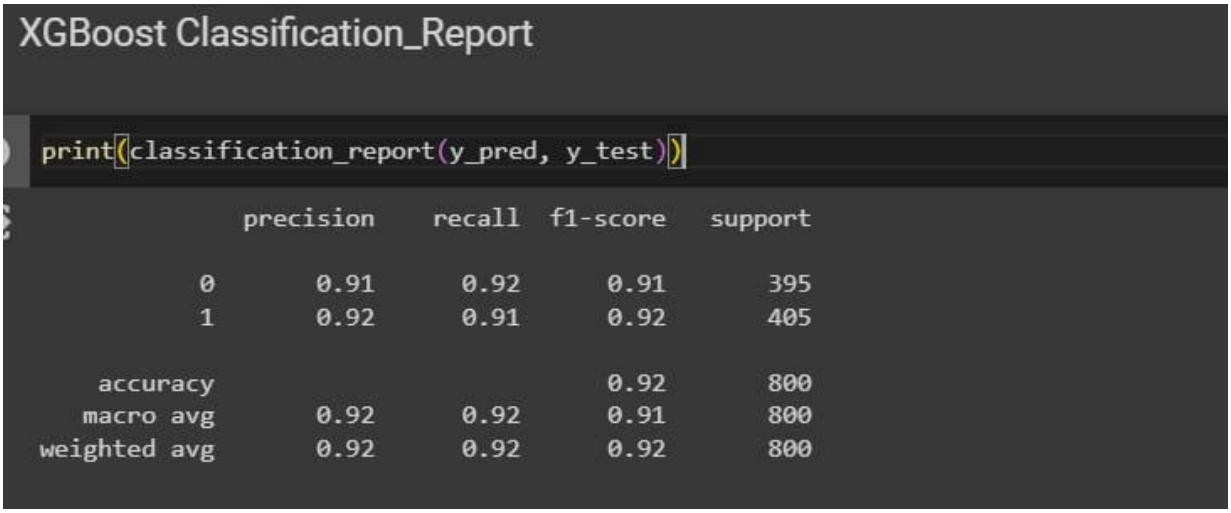
KNN	`from sklearn.model_selection import GridSearchCV, RandomizedSearchCV`
	
	`param_grid = {`
	`'n_estimators': [50, 100, 150],`
	`'max_depth': [None, 10, 20],`
	`'min_samples_split': [2, 5, 10],`
	`'min_samples_leaf': [1, 2, 4]`
	`}`
		`grid_search = GridSearchCV(estimator=model4, param_grid=param_grid, cv=5, scoring='neg_mean_squar`
		`grid_search.fit(X_train, y_train)`
		
		`GridSearchCV(cv=5, estimator=RandomForestRegressor(random_state=42), n_jobs=-1,`
		`param_grid={'max_depth': [None, 10, 20],`
		`'min_samples_leaf': [1, 2, 4],`
		`'min_samples_split': [2, 5, 10],`
		`'n_estimators': [50, 100, 150]},`
		`scoring='neg_mean_squared_error')`

Gradient Boosting	`best_params = grid_search.best_params_`
	`print("Best Hyperparameters:", best_params)`
	
	`Best Hyperparameters: {'max_depth': 20, 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 50}`
	
	`best_model = grid_search.best_estimator_`
	`y_pred = best_model.predict(X_test)`
	`mse = mean_squared_error(y_test, y_pred)`
	`r2_score=r2_score(y_test, y_pred)`
	`print("Mean Squared Error:", mse)`
	`print("R2_score:", r2_score)`
	
	`Mean Squared Error: 36.75979845707393`
	`R2_score: 0.9827970524996451`
		`Mean Squared Error: 36.75979845707`
		`R2_score: 0.9827970524996451`

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric
Decision Tree	DecisionTree Classification_Report `print(classification_report(dt_pred , y_test))` precision recall f1-score support 0.0 0.81 0.80 0.80 404 1.0 0.80 0.80 0.80 396 accuracy 0.80 800 macro avg 0.80 0.80 0.80 800 weighted avg 0.80 0.80 0.80 800

Random Forest	RandomForest Classification_Report
	`from sklearn.metrics import classification_report`
	`print(classification_report(forest, y_test))`
	precision recall f1-score support
	0.0 0.91 0.91 0.91 399
	1.0 0.91 0.91 0.91 401
	accuracy 0.91 800
	macro avg 0.91 0.91 0.91 800
	weighted avg 0.91 0.91 0.91 800

KNN	
Gradient Boosting	

```
kmeans = KMeans(n_clusters=3)

kmeans.fit(df)
centroids = kmeans.cluster_centers_
labels = kmeans.labels_

plt.figure(figsize=(8, 6))
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', s=200, marker='x', label='Centroids')

plt.title('K-means Clustering')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.legend()
plt.grid(True)
plt.show()
```

XGBoost Classification_Report

```
print(classification_report(y_pred, y_test))
```

```
              precision    recall  f1-score   support

           0       0.91      0.92      0.91       395
           1       0.92      0.91      0.92       405

    accuracy                           0.92       800
   macro avg       0.92      0.92      0.91       800
weighted avg       0.92      0.92      0.92       800
```

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Gradient Boosting	The Gradient Boosting model was selected for its superior performance, exhibiting high accuracy during hyperparameter tuning. Its ability to handle complex relationships, minimize overfitting, and optimize predictive accuracy aligns with project objectives, justifying its selection as the final model.