Model Development Phase Template

Date	03 October 2024
Team ID	LTVIP2024TMID24947
Project Title	SmartLender - Applicant Credibility Prediction for Loan Approval
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:

```
#importing and building the random forest model
def RandomForest(X_tarin,X_test,y_train,y_test):
    model = RandomForestClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))
```

```
#printing the train accuracy and test accuracy respectively
RandomForest(X_train,X_test,y_train,y_test)
```

```
#importing and building the Decision tree model
def decisionTree(X_train,X_test,y_train,y_test):
    model = DecisionTreeClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))
```

```
#printing the train accuracy and test accuracy respectively
decisionTree(X_train,X_test,y_train,y_test)
```

```
#importing and building the KNN model
def KNN(X_train,X_test,y_train,y_test):
    model = KNeighborsClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))
```

```
#printing the train accuracy and test accuracy respectively
KNN(X_train,X_test,y_train,y_test)
```

```
#importing and building the Xg boost model
def XGB(X_train,X_test,y_train,y_test):
    model = GradientBoostingClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))
```

```
#printing the train accuracy and test accuracy respectively
XGB(X_train,X_test,y_train,y_test)
```

Model Validation and Evaluation Report:

Model	Classification Report	F1 Score	Confusion Matrix
Random Forest	Classification Report: precision recall f1-score support 0 0.97 0.95 0.96 319 1 0.97 0.98 0.98 535 accuracy 0.97 854 macro avg 0.97 0.97 0.97 854 weighted avg 0.97 0.97 0.97 854	98%	Confusion Matrix: [[304 15] [11 524]]

Decision Tree	Decision Tree Classification Report: precision recall f1-score support 0 0.98 0.99 0.99 313 1 0.99 0.99 0.99 541 accuracy 0.99 854 macro avg 0.99 0.99 0.99 854 weighted avg 0.99 0.99 0.99 854	98%	Decision Tree Confusion Matrix: [[310 3] [5 536]]
KNN	KNN Classification Report: precision recall f1-score support 0 0.87 0.93 0.90 313 1 0.96 0.92 0.94 541 accuracy 0.92 854 macro avg 0.91 0.92 0.92 854 weighted avg 0.93 0.92 0.92 854	92%	KNN Confusion Matrix: [[290 23] [42 499]]
Gradient Boosting	XGBoost Classification Report: precision recall f1-score support 0 0.98 0.99 0.98 313 1 0.99 0.99 0.99 541 accuracy 0.99 854 macro avg 0.99 0.99 0.99 854 weighted avg 0.99 0.99 0.99 854	98%	XGBoost Confusion Matrix: [[309 4] [7 534]]