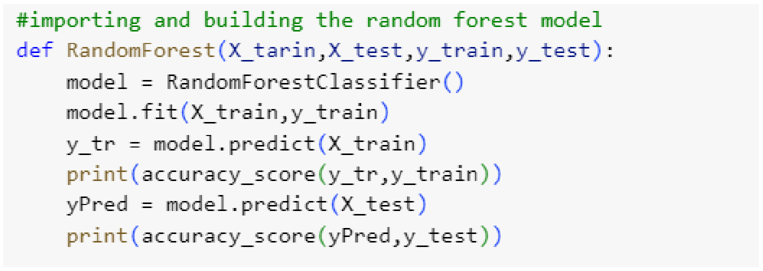
Model Development Phase Template

Date	15 March 2024
Team ID	739926
Project Title	Loan Sanction Amount Prediction Data With Ml
Maximum Marks	4 Marks

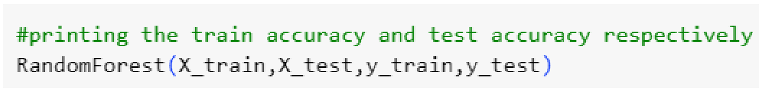
Initial Model Training Code, Model Validation and Evaluation Report

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.

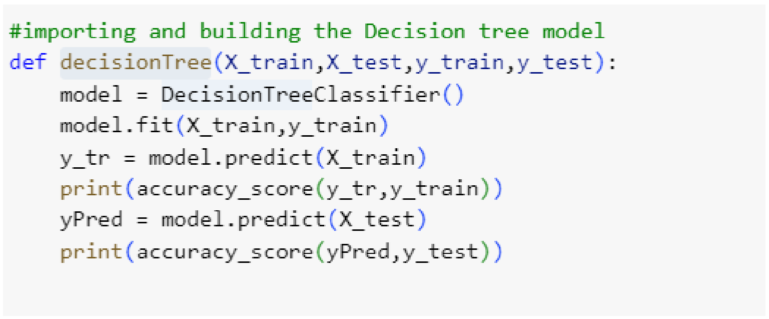
Initial Model Training Code:



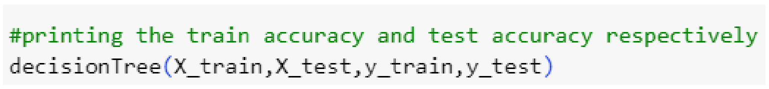
```
#importing and building the random forest model
def RandomForest(X_tarin,X_test,y_train,y_test):
    model = RandomForestClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))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



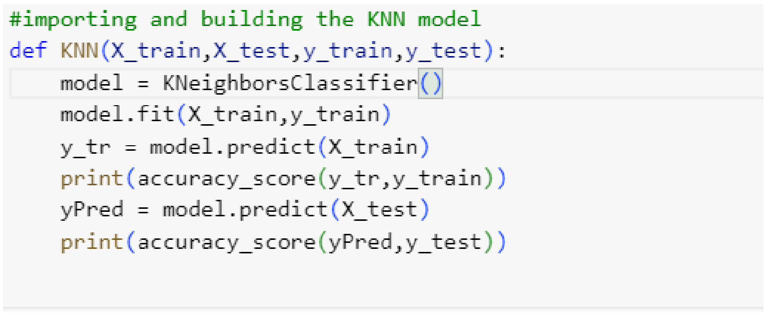
```
#printing the train accuracy and test accuracy respectively
RandomForest(X_train,X_test,y_train,y_test)
```



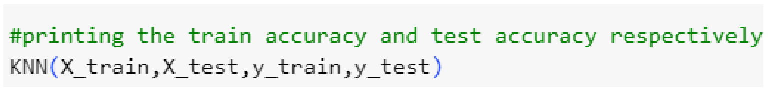
```
#importing and building the Decision tree model
def decisionTree(X_train,X_test,y_train,y_test):
    model = DecisionTreeClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))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



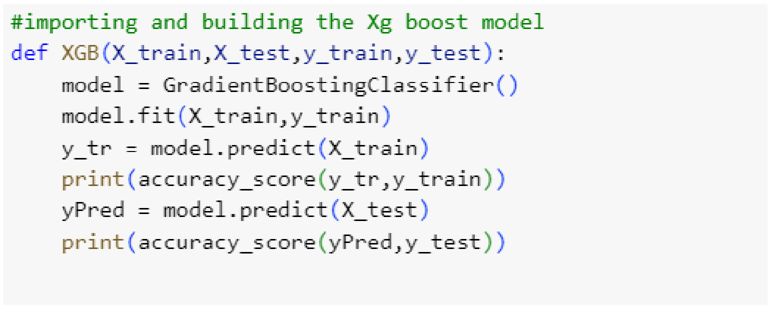
```
#printing the train accuracy and test accuracy respectively
decisionTree(X_train,X_test,y_train,y_test)
```



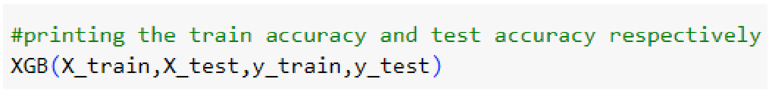
```
#importing and building the KNN model
def KNN(X_train,X_test,y_train,y_test):
    model = KNeighborsClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))
```



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



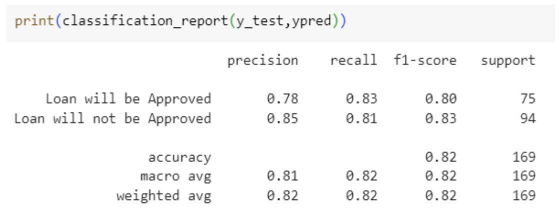
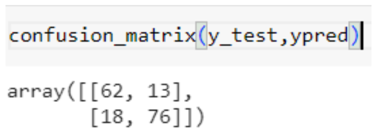
```
#importing and building the Xg boost model
def XGB(X_train,X_test,y_train,y_test):
    model = GradientBoostingClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))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

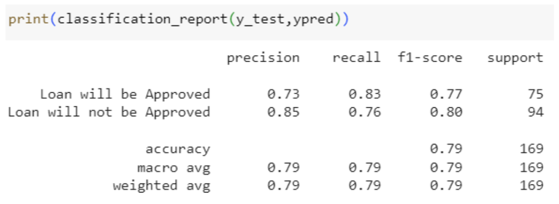
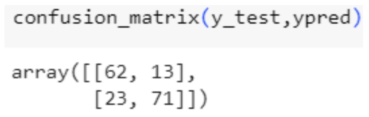


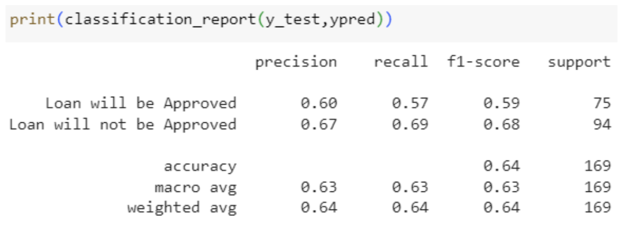
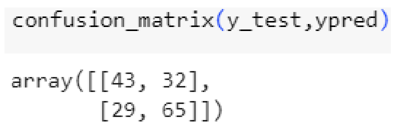
```
#printing the train accuracy and test accuracy respectively
XGB(X_train,X_test,y_train,y_test)
```

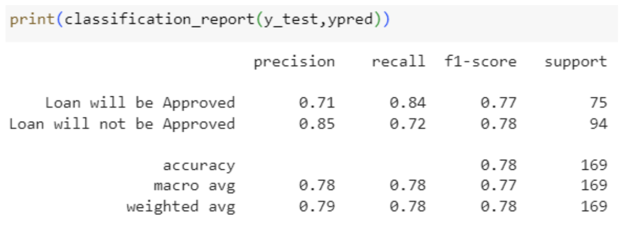
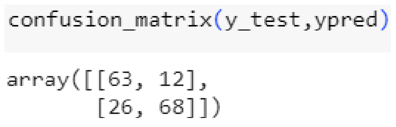
Model Validation and Evaluation Report:

Model	Classification Report	F1 Score	Confusion Matrix

Random Forest	 `print(classification_report(y_test,ypred))`
	 `precision recall f1-score support`
	 `Loan will be Approved 0.78 0.83 0.80 75`
	 `Loan will not be Approved 0.85 0.81 0.83 94`
	 `accuracy 0.82 169`
	 `macro avg 0.81 0.82 0.82 169`
	 `weighted avg 0.82 0.82 0.82 169`
		81%	 `confusion_matrix(y_test,ypred)`
			 `array([[62, 13],`
			 `[18, 76]])`

Decision Tree	 `print(classification_report(y_test,ypred))`
	`                          precision    recall  f1-score   support`
	`    Loan will be Approved       0.73      0.83      0.77        75`
	`Loan will not be Approved       0.85      0.76      0.80        94`
	`                 accuracy                           0.79       169`
	`                macro avg       0.79      0.79      0.79       169`
	`             weighted avg       0.79      0.79      0.79       169`
		79%	 `confusion_matrix(y_test,ypred)`
			`array([[62, 13],`
			`       [23, 71]])`

KNN	 `print(classification_report(y_test,ypred))`
	`                          precision    recall  f1-score   support`
	`    Loan will be Approved       0.60      0.57      0.59        75`
	`Loan will not be Approved       0.67      0.69      0.68        94`
	`                 accuracy                           0.64       169`
	`                macro avg       0.63      0.63      0.63       169`
	`             weighted avg       0.64      0.64      0.64       169`
		64%	 `confusion_matrix(y_test,ypred)`
			`array([[43, 32],`
			`       [29, 65]])`

Gradient Boosting	 `print(classification_report(y_test,ypred))`
	`                          precision    recall  f1-score   support`
	`    Loan will be Approved       0.71      0.84      0.77        75`
	`Loan will not be Approved       0.85      0.72      0.78        94`
	`                 accuracy                           0.78       169`
	`                macro avg       0.78      0.78      0.77       169`
	`             weighted avg       0.79      0.78      0.78       169`
		78%	 `confusion_matrix(y_test,ypred)`
			`array([[63, 12],`
			`       [26, 68]])`