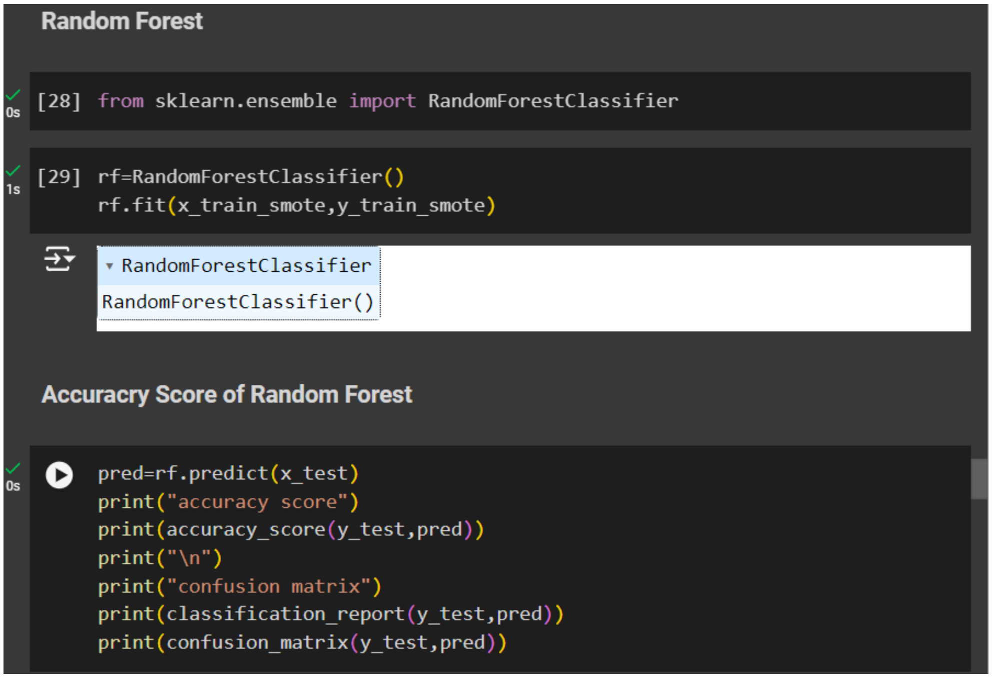
Model Development Phase Template

Date	06-07-2024
Team ID	739694
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:



Random Forest

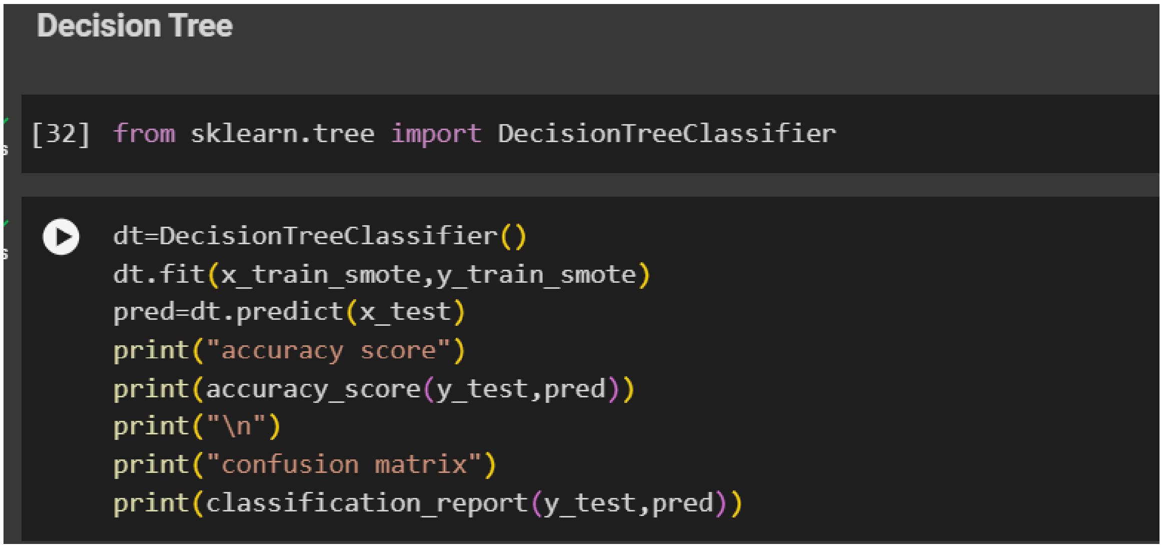
```
[28] from sklearn.ensemble import RandomForestClassifier
```

```
[29] rf=RandomForestClassifier()
     rf.fit(x_train_smote,y_train_smote)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

Accuracy Score of Random Forest

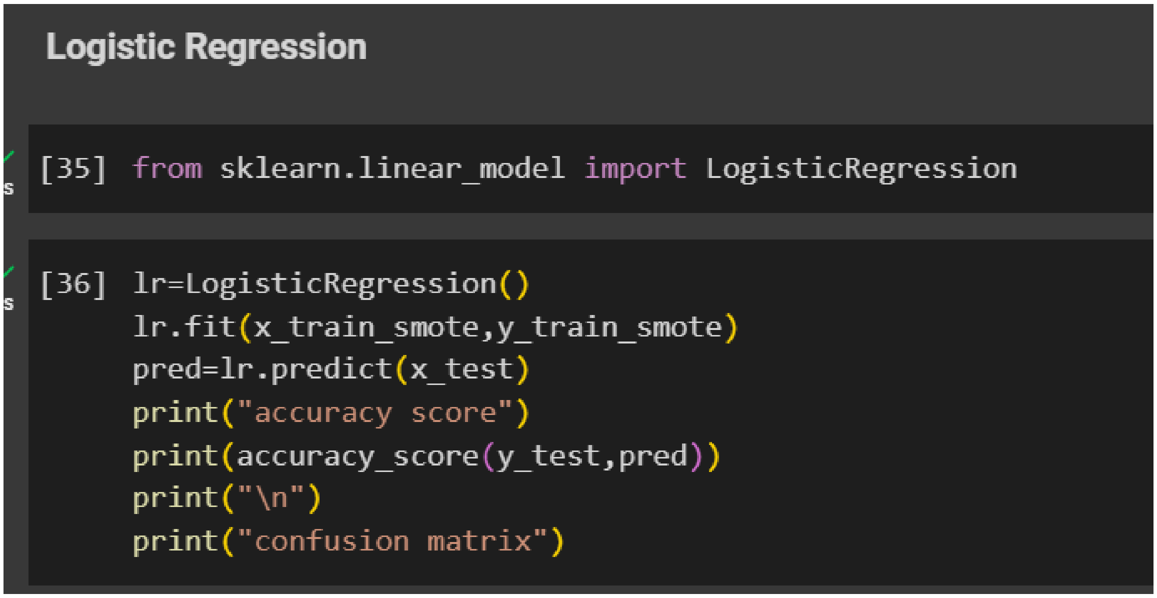
```
pred=rf.predict(x_test)
print("accuracy score")
print(accuracy_score(y_test,pred))
print("\n")
print("confusion matrix")
print(classification_report(y_test,pred))
print(confusion_matrix(y_test,pred))
```



Decision Tree

```
[32] from sklearn.tree import DecisionTreeClassifier
```

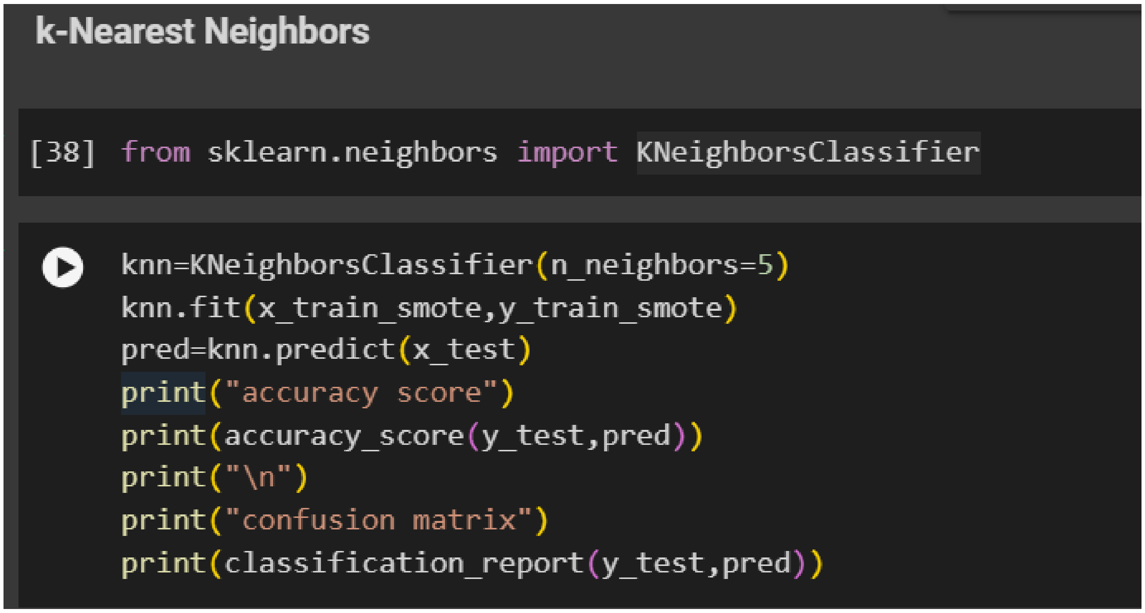
```
dt=DecisionTreeClassifier()
dt.fit(x_train_smote,y_train_smote)
pred=dt.predict(x_test)
print("accuracy score")
print(accuracy_score(y_test,pred))
print("\n")
print("confusion matrix")
print(classification_report(y_test,pred))
```



Logistic Regression

```
[35] from sklearn.linear_model import LogisticRegression
```

```
[36] lr=LogisticRegression()
     lr.fit(x_train_smote,y_train_smote)
     pred=lr.predict(x_test)
     print("accuracy score")
     print(accuracy_score(y_test,pred))
     print("\n")
     print("confusion matrix")
```

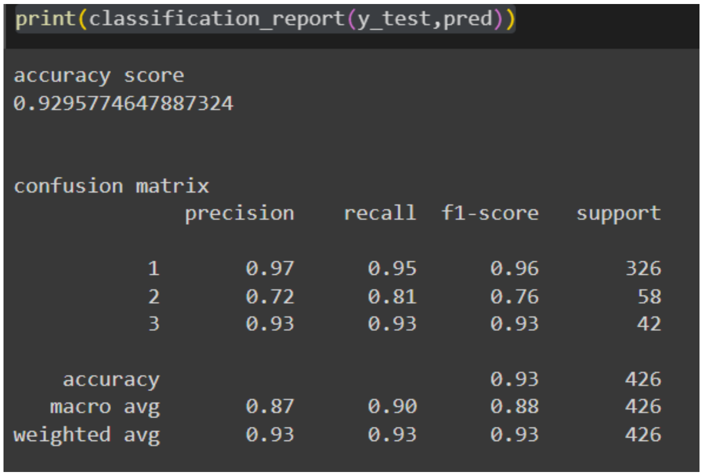


k-Nearest Neighbors

```
[38] from sklearn.neighbors import KNeighborsClassifier
```

```
knn=KNeighborsClassifier(n_neighbors=5)
knn.fit(x_train_smote,y_train_smote)
pred=knn.predict(x_test)
print("accuracy score")
print(accuracy_score(y_test,pred))
print("\n")
print("confusion matrix")
print(classification_report(y_test,pred))
```

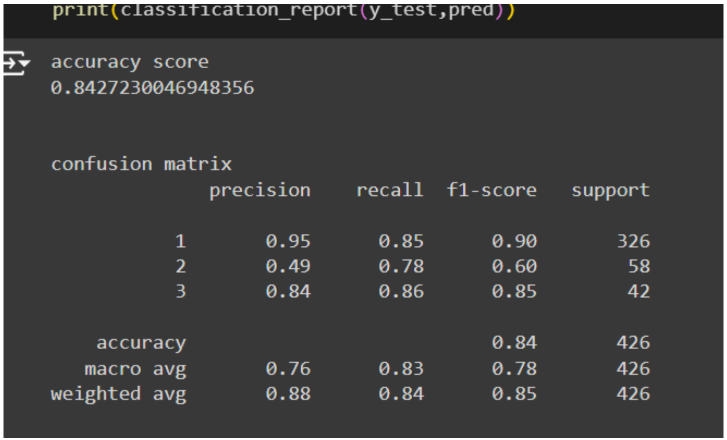
Model Validation and Evaluation Report:

<table>
<tr><th>Model</th><th>Classification Report/ Confusion Matrix</th><th>F1 score</th></tr>
<tr><td>Decision Tree</td><td>
<pre>
print(classification_report(y_test,pred))

accuracy score
0.9295774647887324

confusion matrix
              precision   recall f1-score   support

           1   0.97   0.95   0.96   326
           2   0.72   0.81   0.76   58
           3   0.93   0.93   0.93   42

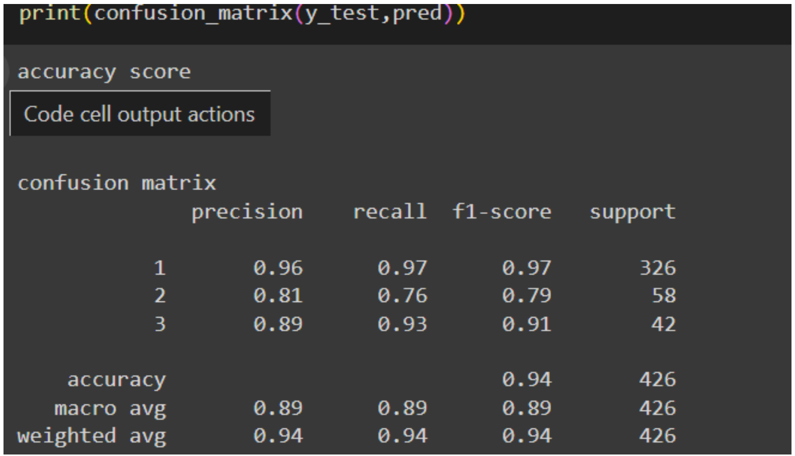
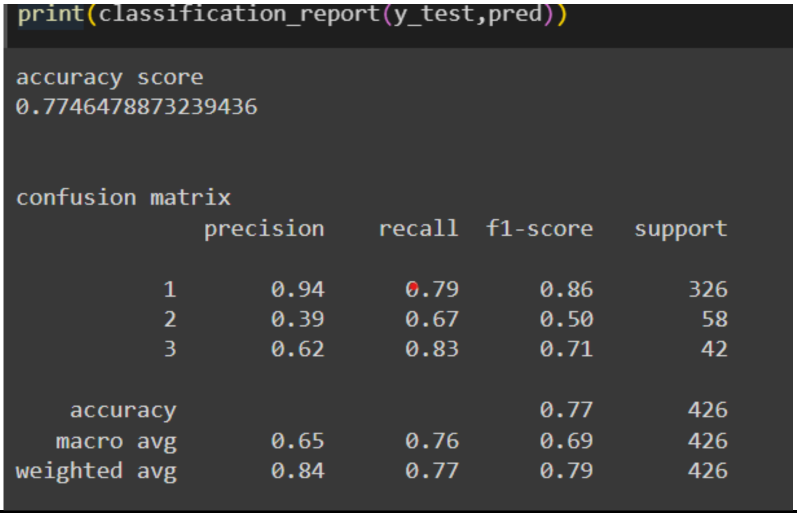
    accuracy   0.93   426
   macro avg   0.87   0.90   0.88   426
weighted avg   0.93   0.93   0.93   426
</pre></td><td>92%</td></tr>
<tr><td>KNN</td><td>
<pre>
print(classification_report(y_test,pred))

accuracy score
0.8427230046948356

confusion matrix
              precision   recall f1-score   support

           1   0.95   0.85   0.90   326
           2   0.49   0.78   0.60   58
           3   0.84   0.86   0.85   42

    accuracy   0.84   426
   macro avg   0.76   0.83   0.78   426
weighted avg   0.88   0.84   0.85   426
</pre></td><td>77%</td></tr>
</table>

Random Forest		93%
Logistic Regression		78%

```
print(confusion_matrix(y_test,pred))

accuracy score
Code cell output actions

confusion matrix
              precision    recall  f1-score   support

           1       0.96      0.97      0.97       326
           2       0.81      0.76      0.79        58
           3       0.89      0.93      0.91        42

    accuracy                           0.94       426
   macro avg       0.89      0.89      0.89       426
weighted avg       0.94      0.94      0.94       426
```

```
print(classification_report(y_test,pred))

accuracy score
0.7746478873239436


confusion matrix
              precision    recall  f1-score   support

           1       0.94      0.79      0.86       326
           2       0.39      0.67      0.50        58
           3       0.62      0.83      0.71        42

    accuracy                           0.77       426
   macro avg       0.65      0.76      0.69       426
weighted avg       0.84      0.77      0.79       426
```