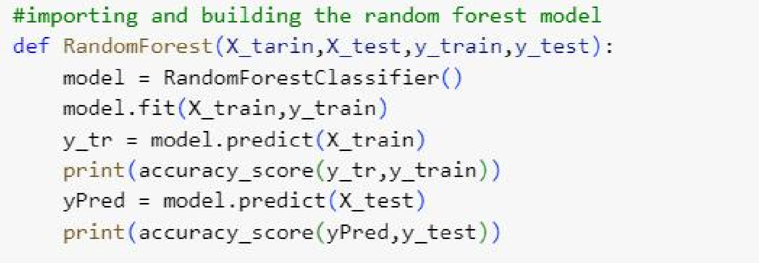
Model Development Phase Template

Date	18 June 2024
Team ID	739642
Project Title	Customer shopping segmentation using machine learning
Maximum Marks	4 Marks

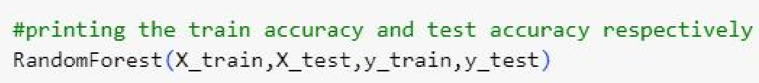
Initial Model Training Code, Model Validation and Evaluation Report

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 for multiple models, presented through respective screenshots.

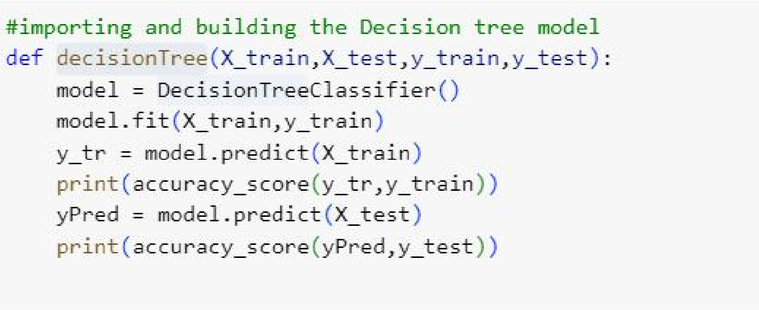
Initial Model Training Code:



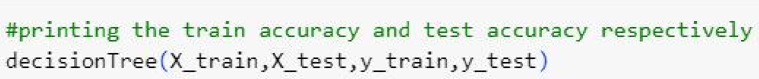
```
#importing and building the random forest model
def RandomForest(X_tarin,X_test,y_train,y_test):
    model = RandomForestClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))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



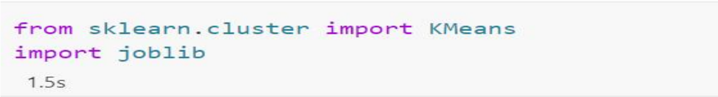
```
#printing the train accuracy and test accuracy respectively
RandomForest(X_train,X_test,y_train,y_test)
```



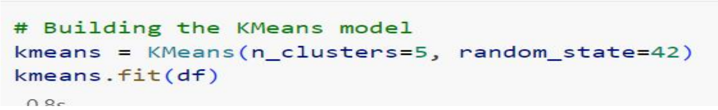
```
#importing and building the Decision tree model
def decisionTree(X_train,X_test,y_train,y_test):
    model = DecisionTreeClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))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



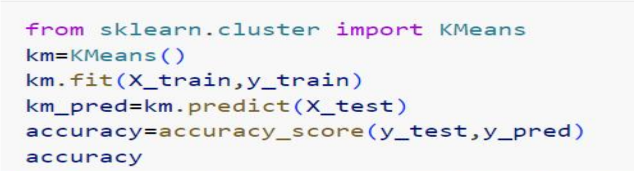
```
#printing the train accuracy and test accuracy respectively
decisionTree(X_train,X_test,y_train,y_test)
```



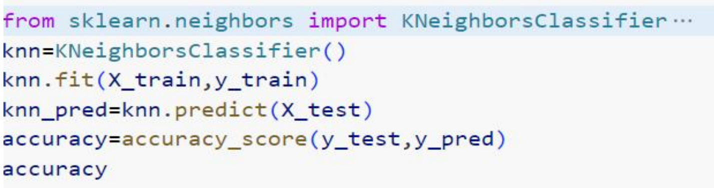
```
from sklearn.cluster import KMeans
import joblib
 1.5s
```



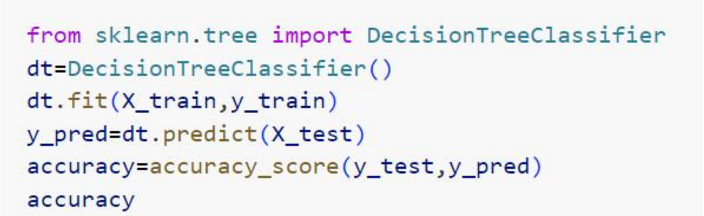
```
# Building the KMeans model
kmeans = KMeans(n_clusters=5, random_state=42)
kmeans.fit(df)
 0.8s
```



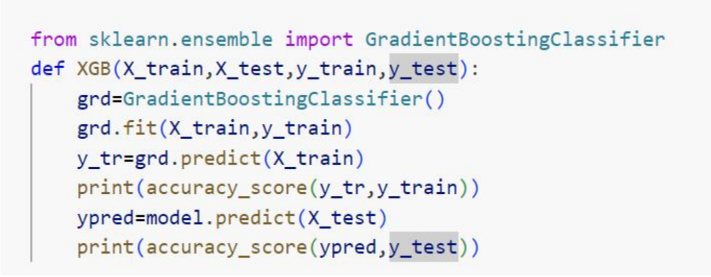
```
from sklearn.cluster import KMeans
km=KMeans()
km.fit(X_train,y_train)
km_pred=km.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
accuracy
```



```
from sklearn.neighbors import KNeighborsClassifier···
knn=KNeighborsClassifier()
knn.fit(X_train,y_train)
knn_pred=knn.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
accuracy
```

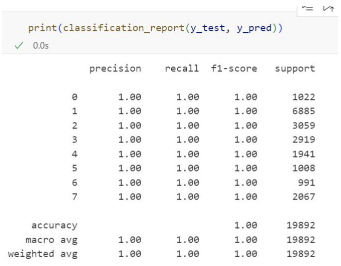


```
from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()
dt.fit(X_train,y_train)
y_pred=dt.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
accuracy
```



```
from sklearn.ensemble import GradientBoostingClassifier
def XGB(X_train,X_test,y_train,y_test):
    grd=GradientBoostingClassifier()
    grd.fit(X_train,y_train)
    y_tr=grd.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    ypred=model.predict(X_test)
    print(accuracy_score(ypred,y_test))
```

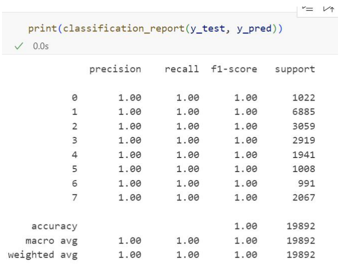
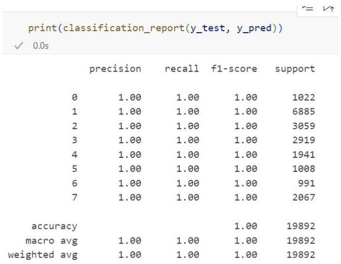
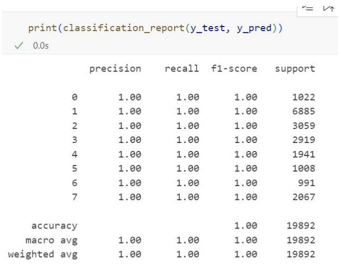
Model Validation and Evaluation Report:

<table>
<tr><th>Model</th><th>Classification Report</th><th>F1 Score</th><th>Confusion Matrix</th></tr>
<tr><td>kmeans</td><td>
<pre>
print(classification_report(y_test, y_pred))
✓ 0.0s

              precision   recall f1-score   support

           0   1.00   1.00   1.00   1022
           1   1.00   1.00   1.00   6885
           2   1.00   1.00   1.00   3059
           3   1.00   1.00   1.00   2919
           4   1.00   1.00   1.00   1941
           5   1.00   1.00   1.00   1008
           6   1.00   1.00   1.00   991
           7   1.00   1.00   1.00   2067

    accuracy   1.00   19892
   macro avg   1.00   1.00   1.00   19892
weighted avg   1.00   1.00   1.00   19892
</pre></td><td>100%</td><td></td></tr>
</table>

Decision Tree	 print(classification_report(y_test, y_pred)) ✓ 0.0s precision recall f1-score support 0 1.00 1.00 1.00 1022 1 1.00 1.00 1.00 6885 2 1.00 1.00 1.00 3059 3 1.00 1.00 1.00 2919 4 1.00 1.00 1.00 1941 5 1.00 1.00 1.00 1008 6 1.00 1.00 1.00 991 7 1.00 1.00 1.00 2067 accuracy 1.00 19892 macro avg 1.00 1.00 1.00 19892 weighted avg 1.00 1.00 1.00 19892	100%	
KNN	 print(classification_report(y_test, y_pred)) ✓ 0.0s precision recall f1-score support 0 1.00 1.00 1.00 1022 1 1.00 1.00 1.00 6885 2 1.00 1.00 1.00 3059 3 1.00 1.00 1.00 2919 4 1.00 1.00 1.00 1941 5 1.00 1.00 1.00 1008 6 1.00 1.00 1.00 991 7 1.00 1.00 1.00 2067 accuracy 1.00 19892 macro avg 1.00 1.00 1.00 19892 weighted avg 1.00 1.00 1.00 19892	100%	
Gradient Boosting	 print(classification_report(y_test, y_pred)) ✓ 0.0s precision recall f1-score support 0 1.00 1.00 1.00 1022 1 1.00 1.00 1.00 6885 2 1.00 1.00 1.00 3059 3 1.00 1.00 1.00 2919 4 1.00 1.00 1.00 1941 5 1.00 1.00 1.00 1008 6 1.00 1.00 1.00 991 7 1.00 1.00 1.00 2067 accuracy 1.00 19892 macro avg 1.00 1.00 1.00 19892 weighted avg 1.00 1.00 1.00 19892	100%	