Model Development Phase Template

Date	05 July 2024
Team ID	739985
Project Title	Anticipating Business Bankruptcy
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 svc
from sklearn.svm import SVC
svm= SVC(kernel='rbf',random_state=0)
svm.fit(x_train,y_train)
```

```
y_pred_svc=svm.predict(x_test)
```

```
print('Training Set:',svm.score(x_train,y_train))
print('Testing Set:',svm.score(x_test,y_test))
```

```
accuracy_SVC=svm.score(x_test,y_test)
print('Accuracy_SVM:',accuracy_SVC*100)
```

```
#importing and building decision classifier
!pip install scikit-learn  # Install scikit-learn if you haven't alrefrom sklearn.tree import DecisionTreeClassifier  # Import the DecisionTreeClass
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error
dt = DecisionTreeClassifier()  # Now you can create an instance
```

```
dt.fit(x_train,y_train)
```
Python

```
y_pred_dt=dt.predict(x_test)
```

```
print('Training Set: ',dt.score(x_train,y_train))
print('Test Set: ',dt.score(x_test,y_test))
```

```
from sklearn import metrics # Import the metrics module
dt = DecisionTreeClassifier()
print("Accuracy:",metrics.accuracy_score(y_test,y_pred_dt)*100) # Now you can use metrics.accuracy_score

accuracy_dt=accuracy_score(y_test,y_pred_dt)
print('Accuracy_DT: ', accuracy_dt*100)


#importing and building random forest
from sklearn.ensemble import RandomForestClassifier
rand_forest= RandomForestClassifier(random_state=42)
rand_forest.fit(x_train,y_train)
```
Python

```
predictionRF=rand_forest .predict(x_test)
#checking the accuracy on the training set
print('Training set :', rand_forest.score(x_train,y_train))
#checking the accuracy on the testing set
print('Testing set :', rand_forest.score(x_test,y_test))
```
Python

```
accuracy_RF=rand_forest.score(x_test, y_test)
print('Accuracy_RF: ', accuracy_RF*100)
```
Python

Model Validation and Evaluation Report:

Model	Classification Report	F1 Score	Confusion Matrix

Support vector classifier	`from sklearn.metrics import classification_report`
	`# Assuming y_test is your tr` (variable) y_test: Any `F is your predicted labels`
	`print(classification_report(y_test,y_pred_svc))`
	
	precision recall f1-score support
	
	0 0.63 0.67 0.65 1991
	1 0.66 0.63 0.64 2063
	
	accuracy 0.65 4054
	macro avg 0.65 0.65 0.65 4054
	weighted avg 0.65 0.65 0.65 4054
		64%	`print(confusion_matrix(y_test, y_pred_dt))`
			
			[[1730 261]
			[192 1871]]

<table>
<tr>
<td>Random Forest</td>
<td>

```
from sklearn.metrics import classification_report

# Assuming y_test is your true labels and predictionRF is your predicted labels
print(classification_report(y_test, predictionRF))
```

```
              precision    recall  f1-score   support

           0       0.95      0.93      0.94      1991
           1       0.94      0.95      0.94      2063

    accuracy                           0.94      4054
   macro avg       0.94      0.94      0.94      4054
weighted avg       0.94      0.94      0.94      4054
```

</td>
<td>94%</td>
<td>

```
print(confusion_matrix(y_test, predictionRF))
```

```
[[1860  131]
 [  98 1965]]
```

</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>

```
from sklearn.metrics import classification_report

# Assuming y_test is your true labels and predictionRF is your predicted labels
print(classification_report(y_test,y_pred_dt))
```

```
              precision    recall  f1-score   support

           0       0.90      0.87      0.88      1991
           1       0.88      0.91      0.89      2063

    accuracy                           0.89      4054
   macro avg       0.89      0.89      0.89      4054
weighted avg       0.89      0.89      0.89      4054
```

</td>
<td>88%</td>
<td>

```
print(confusion_matrix(y_test, y_pred_dt))
```

```
[[1730  261]
 [ 192 1871]]
```

</td>
</tr>
</table>