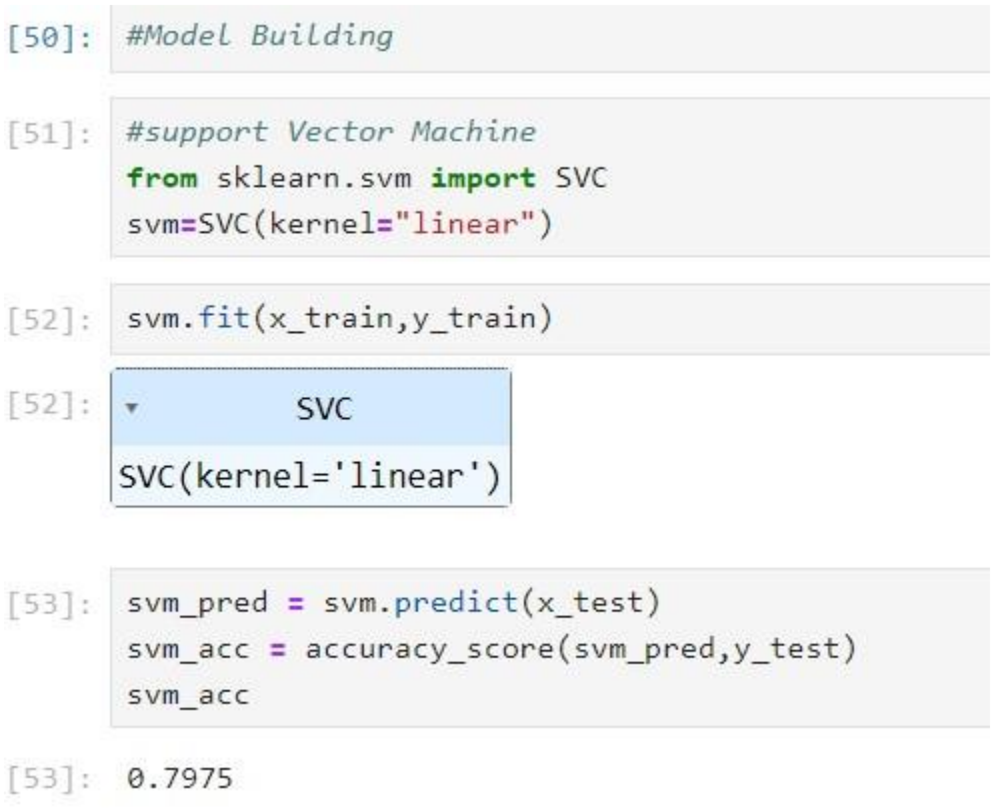
Model Development Phase Template

Date	15 July 2024
Team ID	739886
Project Title	Telecom Customer Churn Prediction
Maximum Marks	4 Marks

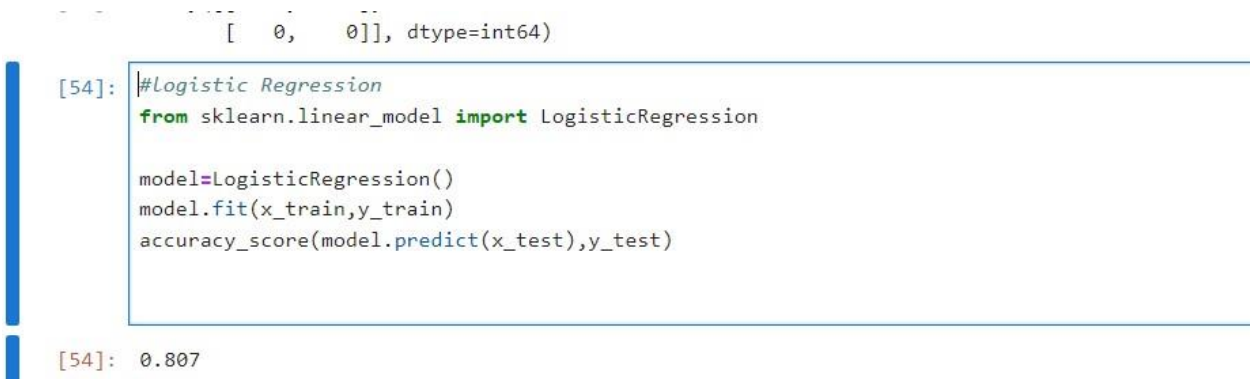
Initial Model Training Code, Model Validation and Evaluation Report

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:

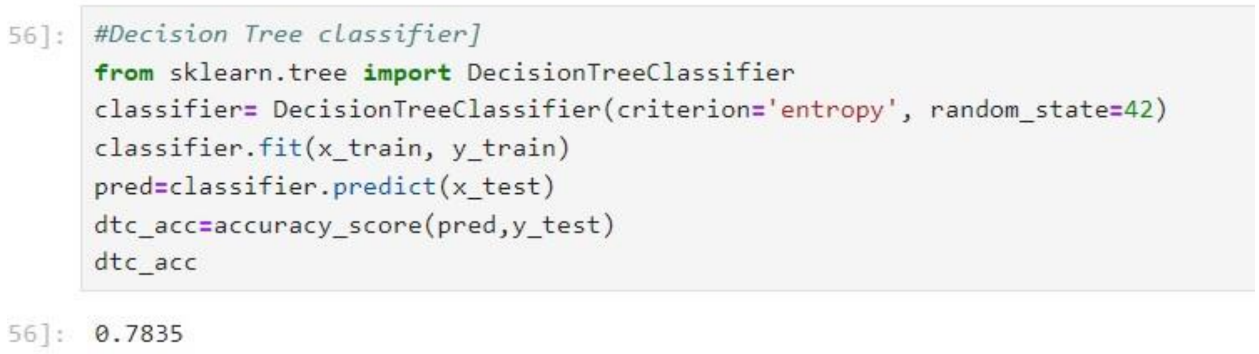


```
[50]: #Model Building

[51]: #support Vector Machine
      from sklearn.svm import SVC
      svm=SVC(kernel="linear")

[52]: svm.fit(x_train,y_train)

[52]: ▾ SVC
      SVC(kernel='linear')

[53]: svm_pred = svm.predict(x_test)
      svm_acc = accuracy_score(svm_pred,y_test)
      svm_acc

[53]: 0.7975
```

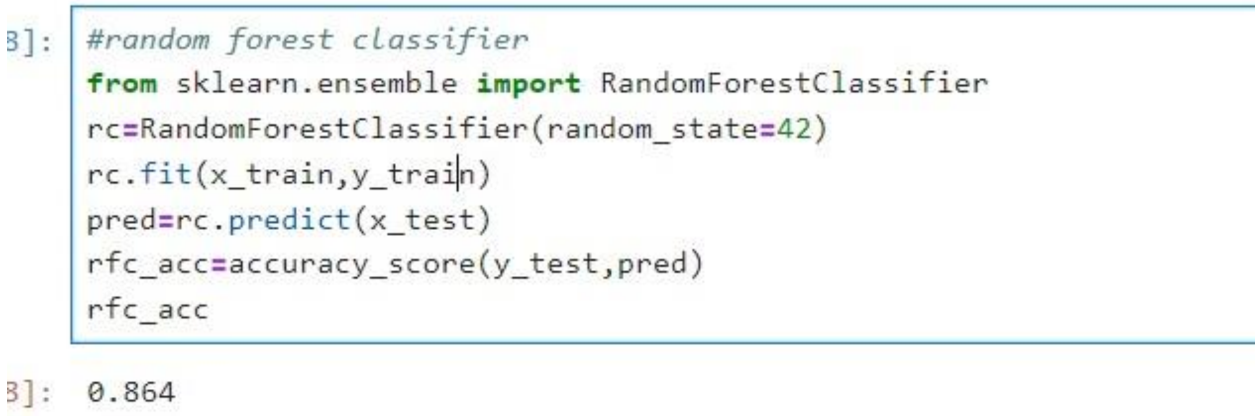


```
          [    0,     0]], dtype=int64)
```

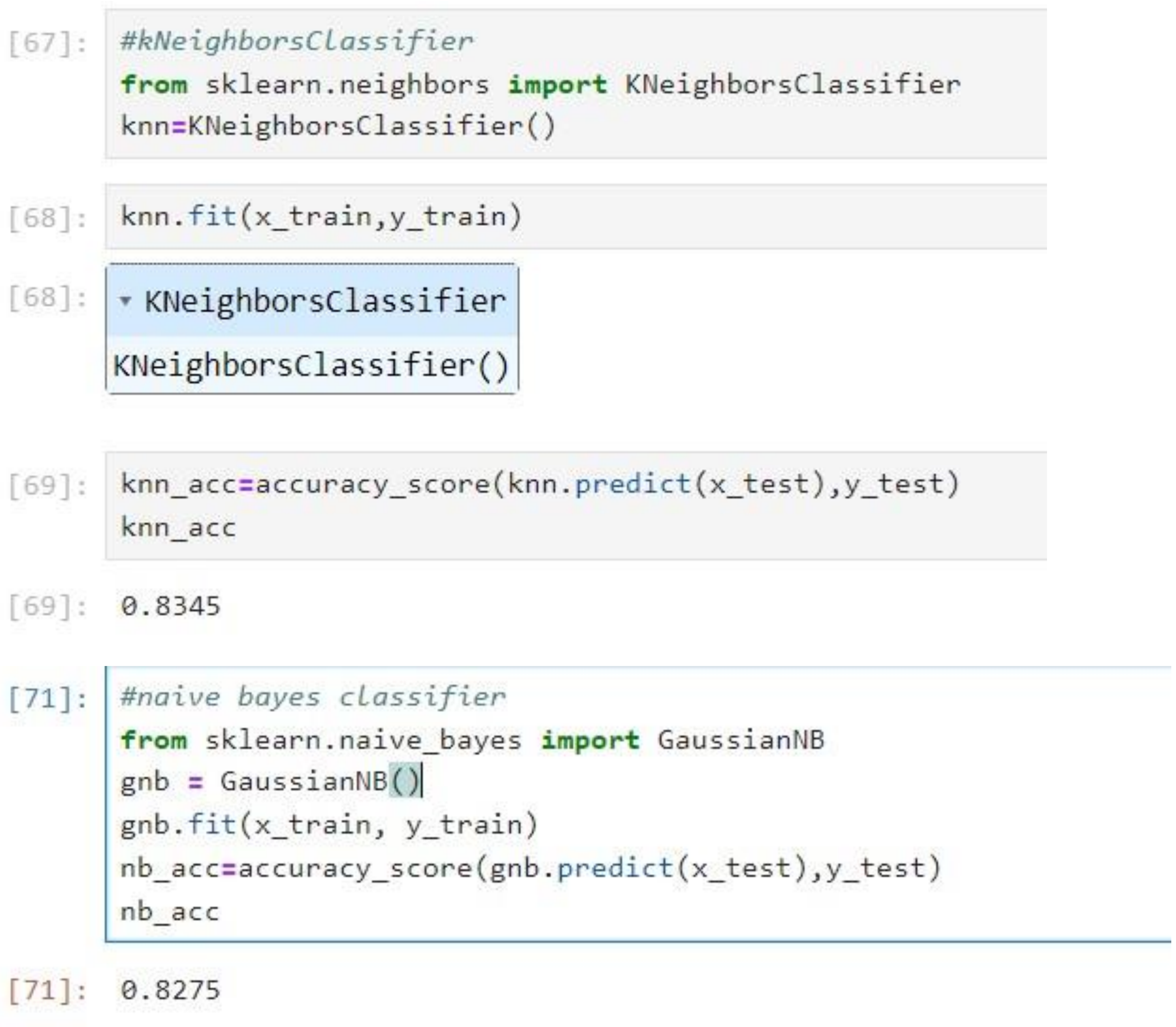
```
[54]: #Logistic Regression
      from sklearn.linear_model import LogisticRegression

      model=LogisticRegression()
      model.fit(x_train,y_train)
      accuracy_score(model.predict(x_test),y_test)

[54]: 0.807
```



```
56]: #Decision Tree classifier]
     from sklearn.tree import DecisionTreeClassifier
     classifier= DecisionTreeClassifier(criterion='entropy', random_state=42)
     classifier.fit(x_train, y_train)
     pred=classifier.predict(x_test)
     dtc_acc=accuracy_score(pred,y_test)
     dtc_acc

56]: 0.7835
```



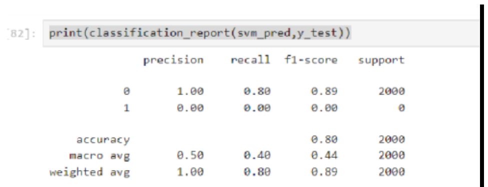
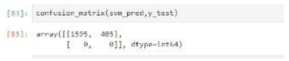
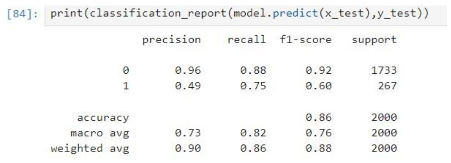
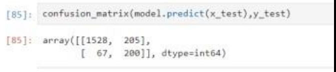
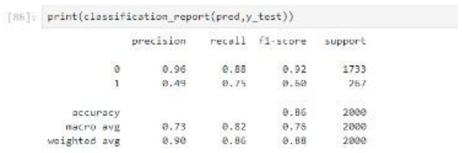
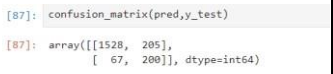
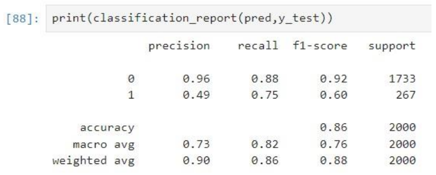
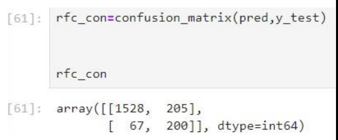
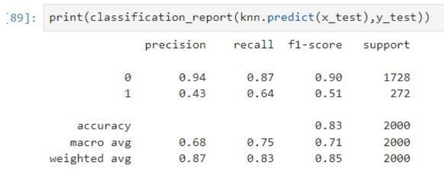
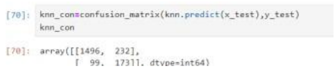
```
3]: #random forest classifier
    from sklearn.ensemble import RandomForestClassifier
    rc=RandomForestClassifier(random_state=42)
    rc.fit(x_train,y_train)
    pred=rc.predict(x_test)
    rfc_acc=accuracy_score(y_test,pred)
    rfc_acc

3]: 0.864
```



```
[67]: #kNeighborsClassifier
      from sklearn.neighbors import KNeighborsClassifier
      knn=KNeighborsClassifier()

[68]: knn.fit(x_train,y_train)

[68]: ▾ KNeighborsClassifier
      KNeighborsClassifier()

[69]: knn_acc=accuracy_score(knn.predict(x_test),y_test)
      knn_acc

[69]: 0.8345

[71]: #naive bayes classifier
      from sklearn.naive_bayes import GaussianNB
      gnb = GaussianNB()
      gnb.fit(x_train, y_train)
      nb_acc=accuracy_score(gnb.predict(x_test),y_test)
      nb_acc

[71]: 0.8275
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix

svm		79	
Logistic regression		80	
Decision Tree		78	
Random Forest		86	
knn		83	

```
82]: print(classification_report(svm_pred,y_test))

              precision    recall  f1-score   support

           0       1.00      0.80      0.89      2000
           1       0.00      0.00      0.00         0

    accuracy                           0.80      2000
   macro avg       0.50      0.40      0.44      2000
weighted avg       1.00      0.80      0.89      2000
```

```
[83]: confusion_matrix(svm_pred,y_test)

[83]: array([[1595,  405],
             [   0,    0]], dtype=int64)
```

```
[84]: print(classification_report(model.predict(x_test),y_test))

              precision    recall  f1-score   support

           0       0.96      0.88      0.92      1733
           1       0.49      0.75      0.60       267

    accuracy                           0.86      2000
   macro avg       0.73      0.82      0.76      2000
weighted avg       0.90      0.86      0.88      2000
```

```
[85]: confusion_matrix(model.predict(x_test),y_test)

[85]: array([[1528,  205],
             [  67,  200]], dtype=int64)
```

```
[86]: print(classification_report(pred,y_test))

              precision    recall  f1-score   support

           0       0.96      0.88      0.92      1733
           1       0.49      0.75      0.60       267

    accuracy                           0.86      2000
   macro avg       0.73      0.82      0.76      2000
weighted avg       0.90      0.86      0.88      2000
```

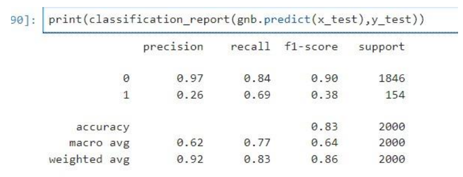
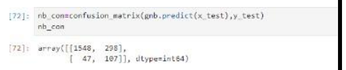
```
[87]: confusion_matrix(pred,y_test)

[87]: array([[1528,  205],
             [  67,  200]], dtype=int64)
```

```
[88]: print(classification_report(pred,y_test))

              precision    recall  f1-score   support

           0       0.96      0.88      0.92      1733
           1       0.49      0.75      0.60       267

    accuracy                           0.86      2000
   macro avg       0.73      0.82      0.76      2000
weighted avg       0.90      0.86      0.88      2000
```

```
[61]: rfc_con=confusion_matrix(pred,y_test)


      rfc_con

[61]: array([[1528,  205],
             [  67,  200]], dtype=int64)
```

```
89]: print(classification_report(knn.predict(x_test),y_test))

              precision    recall  f1-score   support

           0       0.94      0.87      0.90      1728
           1       0.43      0.64      0.51       272

    accuracy                           0.83      2000
   macro avg       0.68      0.75      0.71      2000
weighted avg       0.87      0.83      0.85      2000
```

```
[70]: knn_con=confusion_matrix(knn.predict(x_test),y_test)
      knn_con

[70]: array([[1496,  232],
             [  99,  173]], dtype=int64)
```

Naïve bayes		82	

```
90]:  print(classification_report(gnb.predict(x_test),y_test))

              precision    recall  f1-score   support

           0       0.97      0.84      0.90      1846
           1       0.26      0.69      0.38       154

    accuracy                           0.83      2000
   macro avg       0.62      0.77      0.64      2000
weighted avg       0.92      0.83      0.86      2000
```

```
[72]: nb_con=confusion_matrix(gnb.predict(x_test),y_test)
      nb_con

[72]: array([[1548,  298],
             [  47,  107]], dtype=int64)
```