Model Development Phase Template

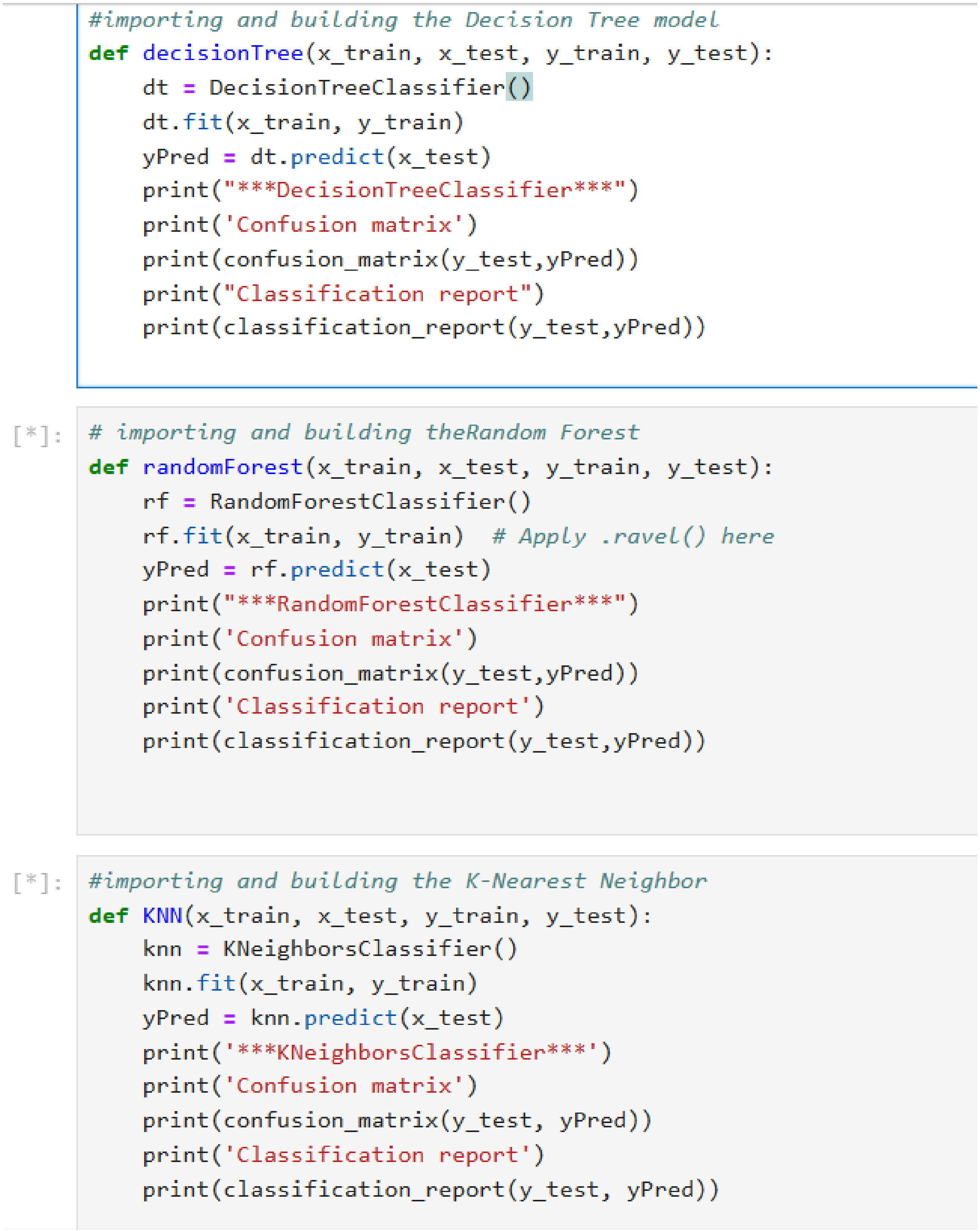
Date	24 April 2024
Team ID	739847
Project Title	One Year Life Expectancy post on Thoracic Surgery using Machine Learning
Maximum Marks	4 Marks

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, and confusion matrices for multiple models, presented through respective screenshots.

- Thoracic surgery involves procedures on the lungs, esophagus, and chest, which come with significant risks.
- Accurate predictions of post-surgery survival rates help in tailoring patient care and making informed surgical decisions.
- The ML model leverages a dataset comprising various patient attributes and health indicators to make these predictions.
- Key features in the dataset include diagnosis type, lung function tests (FVC and FEV1), performance status, symptoms (pain, haemoptysis, dyspnoea, cough, weakness), tumor size, and presence of comorbidities such as diabetes, recent myocardial infarction, peripheral arterial disease, smoking, and asthma.
- The target variable is the patient's one-year death outcome.
- Machine learning algorithms can process these diverse and complex data points, identifying patterns and correlations that may not be immediately evident through traditional statistical methods.
- By training models on historical patient data, ML can predict the likelihood of a patient surviving one-year post surgery, providing valuable insights for clinicians.

Initial Model Training Code:



```python
#importing and building the Decision Tree model
def decisionTree(x_train, x_test, y_train, y_test):
    dt = DecisionTreeClassifier()
    dt.fit(x_train, y_train)
    yPred = dt.predict(x_test)
    print("***DecisionTreeClassifier***")
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))
    print("Classification report")
    print(classification_report(y_test,yPred))
```

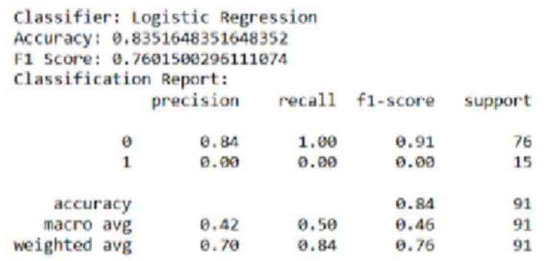
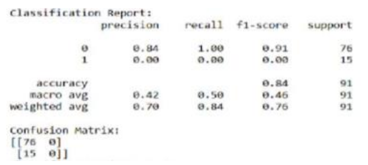
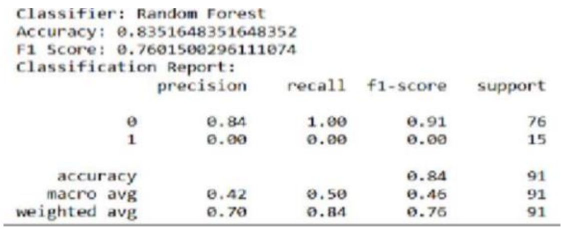
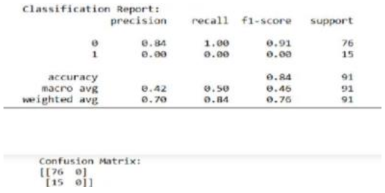
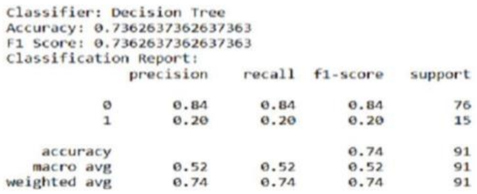
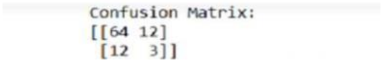
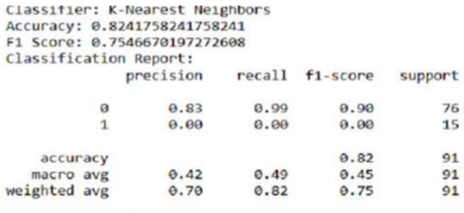
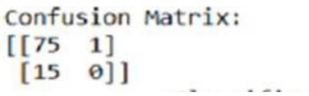
[*]:

```python
# importing and building theRandom Forest
def randomForest(x_train, x_test, y_train, y_test):
    rf = RandomForestClassifier()
    rf.fit(x_train, y_train)  # Apply .ravel() here
    yPred = rf.predict(x_test)
    print("***RandomForestClassifier***")
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))
    print('Classification report')
    print(classification_report(y_test,yPred))
```

[*]:

```python
#importing and building the K-Nearest Neighbor
def KNN(x_train, x_test, y_train, y_test):
    knn = KNeighborsClassifier()
    knn.fit(x_train, y_train)
    yPred = knn.predict(x_test)
    print('***KNeighborsClassifier***')
    print('Confusion matrix')
    print(confusion_matrix(y_test, yPred))
    print('Classification report')
    print(classification_report(y_test, yPred))
```

Model Validation and Evaluation Report:

Model	Classification Report	F1-Score	Confusion Matrix
Logistic Regression		**76%**	
Random forest		**76%**	
Decision Tree		**73%**	
KNN		**75%**	

```
Classifier: Logistic Regression
Accuracy: 0.8351648351648352
F1 Score: 0.7601500296111074
Classification Report:
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        76
           1       0.00      0.00      0.00        15

    accuracy                           0.84        91
   macro avg       0.42      0.50      0.46        91
weighted avg       0.70      0.84      0.76        91
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        76
           1       0.00      0.00      0.00        15

    accuracy                           0.84        91
   macro avg       0.42      0.50      0.46        91
weighted avg       0.70      0.84      0.76        91

Confusion Matrix:
[[76  0]
 [15  0]]
```

```
Classifier: Random Forest
Accuracy: 0.8351648351648352
F1 Score: 0.7601500296111074
Classification Report:
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        76
           1       0.00      0.00      0.00        15

    accuracy                           0.84        91
   macro avg       0.42      0.50      0.46        91
weighted avg       0.70      0.84      0.76        91
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        76
           1       0.00      0.00      0.00        15

    accuracy                           0.84        91
   macro avg       0.42      0.50      0.46        91
weighted avg       0.70      0.84      0.76        91

Confusion Matrix:
[[76  0]
 [15  0]]
```

```
Classifier: Decision Tree
Accuracy: 0.7362637362637363
F1 Score: 0.7362637362637363
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.84      0.84        76
           1       0.20      0.20      0.20        15

    accuracy                           0.74        91
   macro avg       0.52      0.52      0.52        91
weighted avg       0.74      0.74      0.74        91
```

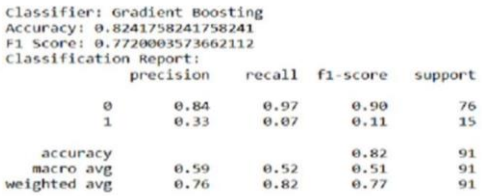
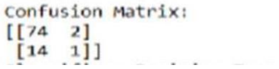
```
Confusion Matrix:
[[64 12]
 [12  3]]
```

```
Classifier: K-Nearest Neighbors
Accuracy: 0.8241758241758241
F1 Score: 0.7546670197272608
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.99      0.90        76
           1       0.00      0.00      0.00        15

    accuracy                           0.82        91
   macro avg       0.42      0.49      0.45        91
weighted avg       0.70      0.82      0.75        91
```

```
Confusion Matrix:
[[75  1]
 [15  0]]
```

Gradient boosting	 Classifier: Gradient Boosting Accuracy: 0.8241758241758241 F1 Score: 0.7720003573662112 Classification Report: precision recall f1-score support 0 0.84 0.97 0.90 76 1 0.33 0.07 0.11 15 accuracy 0.82 91 macro avg 0.59 0.52 0.51 91 weighted avg 0.76 0.82 0.77 91	**77%**	 Confusion Matrix: [[74 2] [14 1]]