MA691: Advanced Statistical Algorithms

Instructor*:* Dr.Arabin Kumar Dey

Submission Date: 19.11.21

Group Members:

Ankita Singh	**180122008**
Bineeta Oram	**180123009**
Kashan Hasan	**180123022**
Sandra	**180121037**

DISCLAIMER: This work is for learning purpose only. This work cannot be used for publication or as commercial products, etc. without instructor's consent.

- **Task :** Implementation of PyCobra in UCI Haberman's Survival Dataset
- **Type :** Classification
- **Dataset:** https://archive.ics.uci.edu/ml/datasets/haberman's+survival
- Performing various EDA techniques using python.
- Use basic PyCobra for ensembling the result of machines available.
- Separate analysis of results using those classifiers machines individually.

1. Understanding the dataset

Title: Haber man's Survival Data

Description: The dataset contains cases from a study that was conducted between 1958 and 1970 at the University of Chicago's Billings Hospital on the survival of patients who had undergone surgery for breast cancer.

Attribute Information:

Age of patient at the time of operation (numerical)

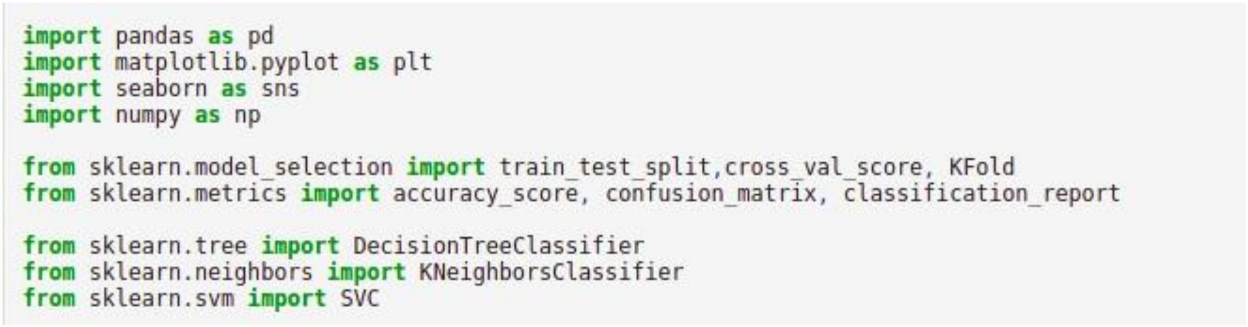
Patient's year of operation (year — 1900, numerical)

Number of positive axillary nodes detected.

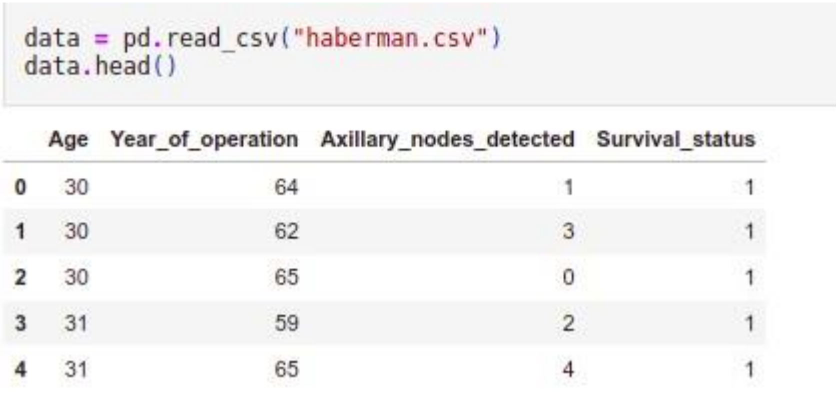
Survival status (class attribute):

1 = the patient survived 5 years or longer

2 = the patient died within 5 years

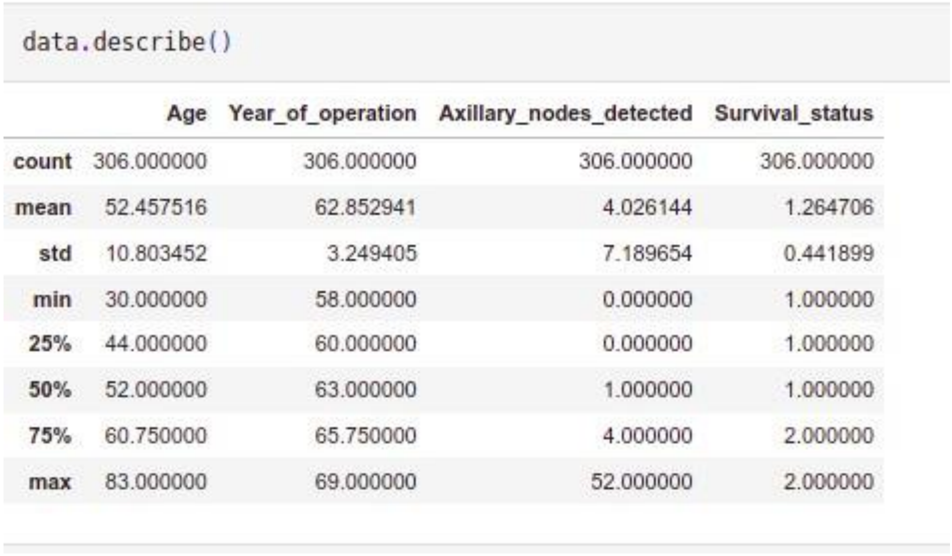


```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

from sklearn.model_selection import train_test_split,cross_val_score, KFold
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
```



```
data = pd.read_csv("haberman.csv")
data.head()
```

	Age	Year_of_operation	Axillary_nodes_detected	Survival_status
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1



```
data.describe()
```

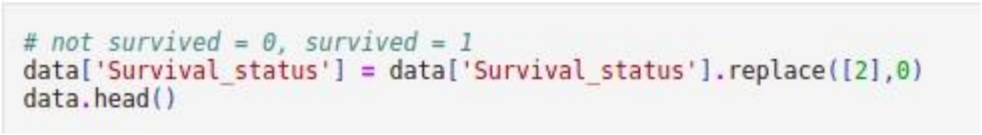
	Age	Year_of_operation	Axillary_nodes_detected	Survival_status
count	306.000000	306.000000	306.000000	306.000000
mean	52.457516	62.852941	4.026144	1.264706
std	10.803452	3.249405	7.189654	0.441899
min	30.000000	58.000000	0.000000	1.000000
25%	44.000000	60.000000	0.000000	1.000000
50%	52.000000	63.000000	1.000000	1.000000
75%	60.750000	65.750000	4.000000	2.000000
max	83.000000	69.000000	52.000000	2.000000

Observations:

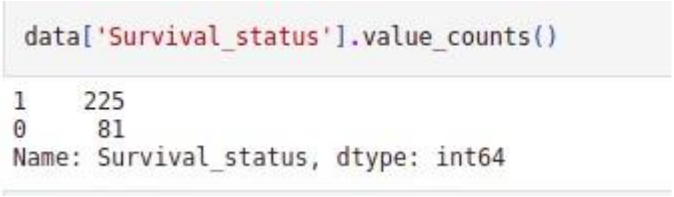
1. There are no missing values in this data set.
2. All the columns are of the integer data type.
3. The datatype of the status is an integer, it has to be converted to a categorical datatype.

4. In the status column, the value 1 means the patient has survived 5 years or longer and the value 2 can be mapped to ‘0’ which means the patient died within 5 years.

2. Analysing the attributes and their correlation:



```
# not survived = 0, survived = 1
data['Survival_status'] = data['Survival_status'].replace([2],0)
data.head()
```



```
data['Survival_status'].value_counts()
```

```
1    225
0     81
Name: Survival_status, dtype: int64
```

Observations:

1. The value_counts() function tells how many data points for each class are present. Here, it tells how many patients survived and how many did not survive.
2. Out of 306 patients, 225 patients survived and 81 did not.

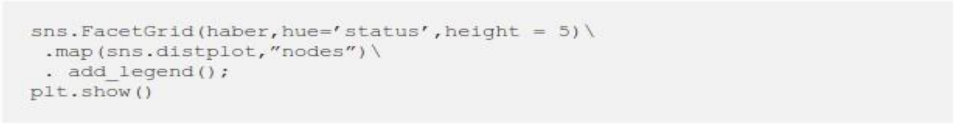
The dataset is imbalanced.

3. Univariate Analysis

The major purpose of the univariate analysis is to describe, summarize and find patterns in the single feature.

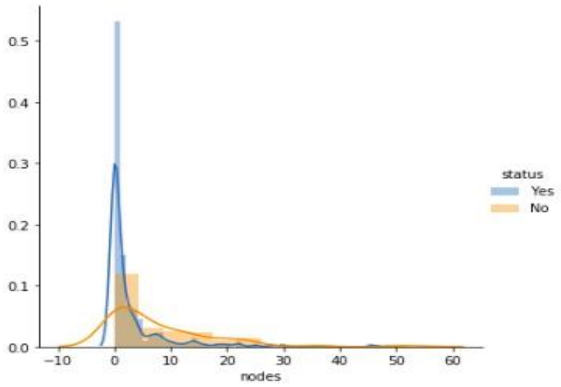
A. Probability Density Function (PDF)

Probability Density Function (PDF) is the probability that the variable takes a value x. (a smoothed version of the histogram). Here the height of the bar denotes the percentage of data points under the corresponding group



```
sns.FacetGrid(haber,hue=’status’,height = 5)\
 .map(sns.distplot,”nodes”)\
 . add_legend();
plt.show()
```

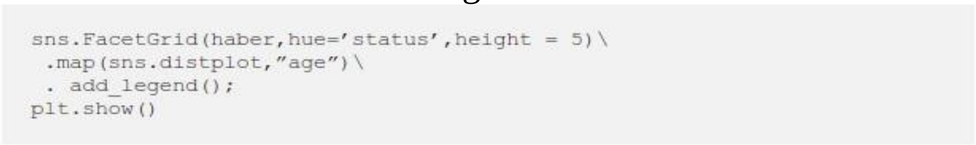
Output:



PDF of Nodes

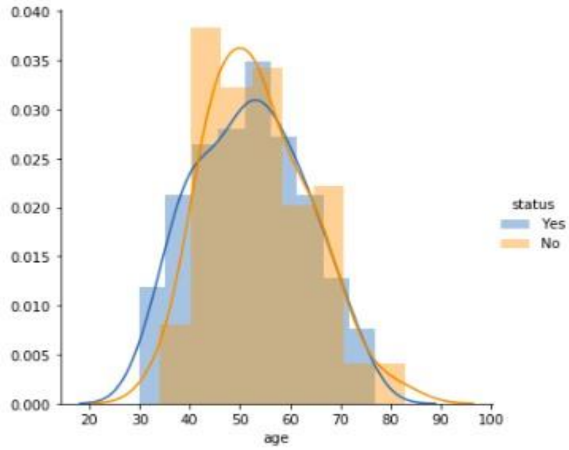
Observations:

Patients with no nodes or 1 node are more likely to survive. There are very few chances of surviving if there are 25 or more nodes.



```
sns.FacetGrid(haber,hue=’status’,height = 5)\
 .map(sns.distplot,”age”)\
 . add_legend();
plt.show()
```

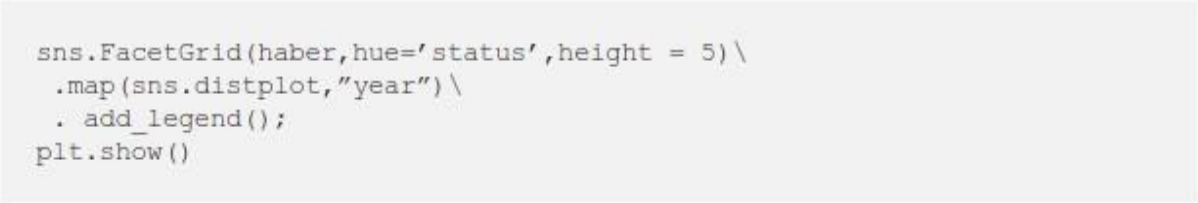
Output:



PDF of Age

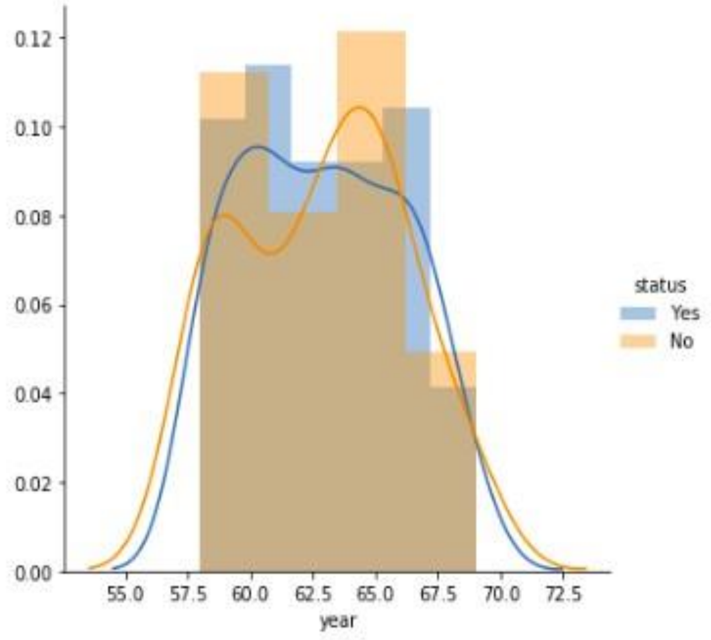
Observations:

1. Major overlapping is observed, which tells us that survival chances are irrespective of a person’s age.
2. Although there is overlap, we can vaguely tell that people whose age is in the range 30–40 are more likely to survive, and 40–60 are less likely to survive. While people whose age is in the range 60–75 have equal chances of surviving and not surviving.
3. Yet, this cannot be our final conclusion. We cannot decide the survival chances of a patient just by considering the age parameter



```
sns.FacetGrid(haber,hue=’status’,height = 5)\
 .map(sns.distplot,”year”)\
 . add_legend();
plt.show()
```

Output:



PDF of Year

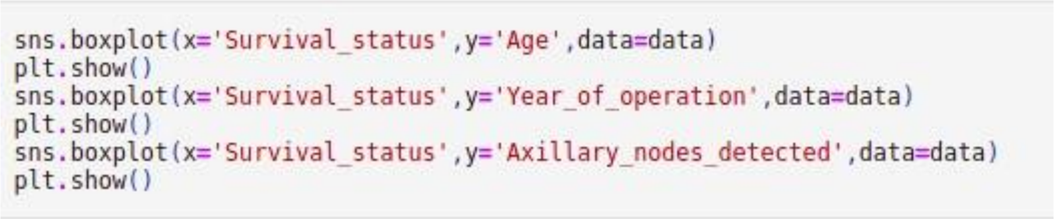
Observations:

1. There is major overlapping observed. This graph only tells how many of the operations were successful and how many weren't. This cannot be a parameter to decide the patient's survival chances.
2. However, it can be observed that in the years 1960 and 1965 there were more unsuccessful operations.

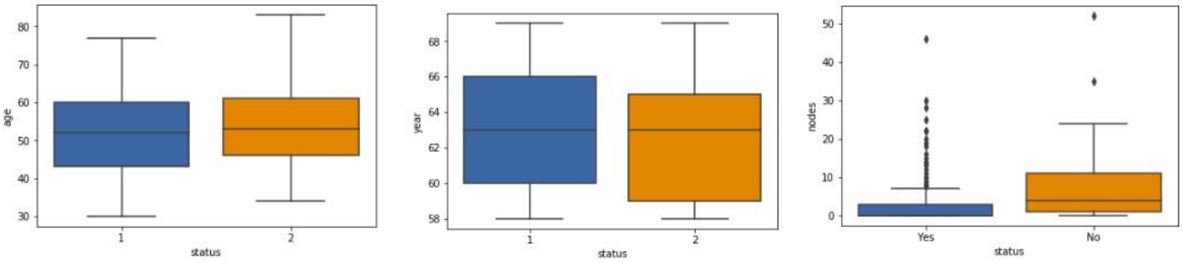
B. Box Plots and Violin Plots

The box extends from the lower to upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Outlier points are those past the end of the whiskers.

Violin plot is the combination of a box plot and probability density function(CDF).



```
sns.boxplot(x='Survival_status',y='Age',data=data)
plt.show()
sns.boxplot(x='Survival_status',y='Year_of_operation',data=data)
plt.show()
sns.boxplot(x='Survival_status',y='Axillary_nodes_detected',data=data)
plt.show()
```



Box plots

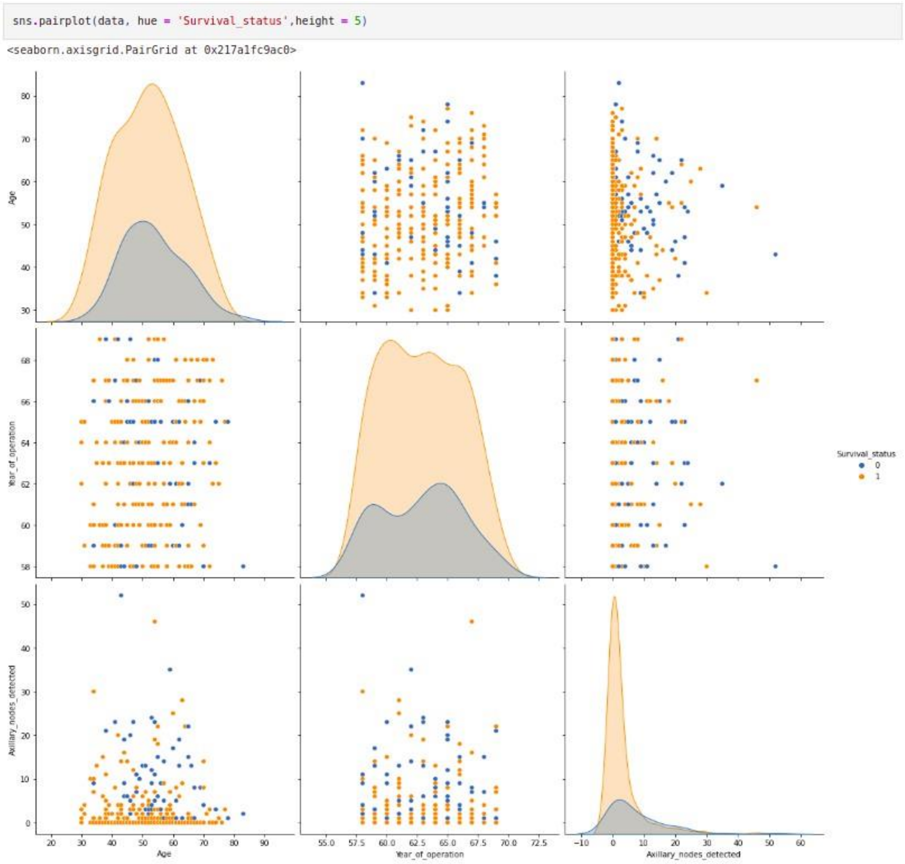
Observations:

1. Patients with more than 1 node are not likely to survive. More the number of nodes, lesser the survival chances.
2. A large percentage of patients who survived had 0 nodes. Yet there is a small percentage of patients who had no positive axillary nodes died within 5 years of operation, thus an absence of positive axillary nodes cannot always guarantee survival.

3. There were comparatively more people who got operated in the year 1965 did not survive for more than 5 years.
4. There were comparatively more people in the age group 45 to 65 who did not survive. Patient age alone is not an important parameter in determining the survival of a patient.
5. The box plots and violin plots for age and year parameters give similar results with a substantial overlap of data points. The overlap in the box plot and the violin plot of nodes is less compared to other features but the overlap still exists and thus it is difficult to set a threshold to classify both classes of patients.

4 .Bi-Variate analysis

It is a two-dimensional data visualization that uses dots to represent the values obtained for two different variables — one plotted along the x-axis and the other plotted along the y-axis.



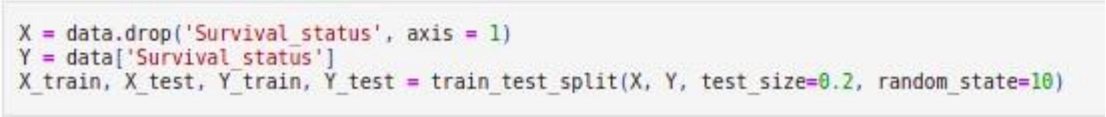
```
sns.pairplot(data, hue = 'Survival_status',height = 5)
```

```
<seaborn.axisgrid.PairGrid at 0x217a1fc9ac0>
```

5. Conclusions:

1. Patient’s age and operation year alone are not deciding factors for his/her survival. Yet, people less than 35 years have more chance of survival.
2. Survival chance is inversely proportional to the number of positive axillary nodes. We also saw that the absence of positive axillary nodes cannot always guarantee survival.
3. The objective of classifying the survival status of a new patient based on the given features is a difficult task as the data is imbalanced.

6. Train-Test Split

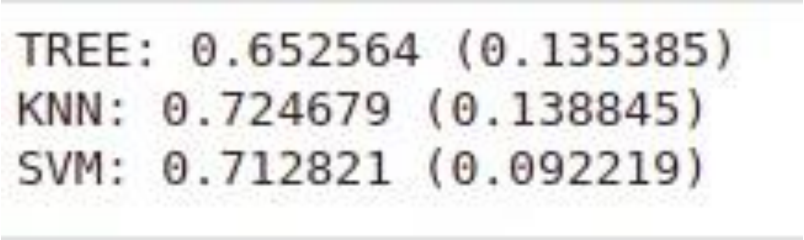


```
X = data.drop('Survival_status', axis = 1)
Y = data['Survival_status']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=10)
```

7. Training using different classifier

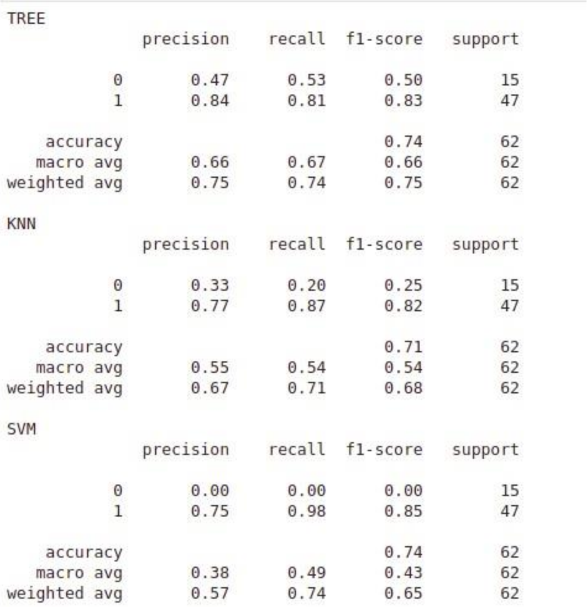
We choose three classifiers namely:

1. Decision Tree
2. K Nearest neighbour
3. Support Vector Machine



```
TREE: 0.652564 (0.135385)
KNN: 0.724679 (0.138845)
SVM: 0.712821 (0.092219)
```

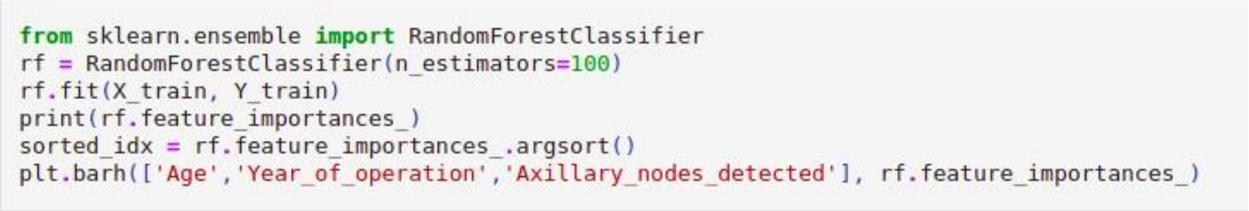
The precision, recall and F1 score for all three classifiers are:



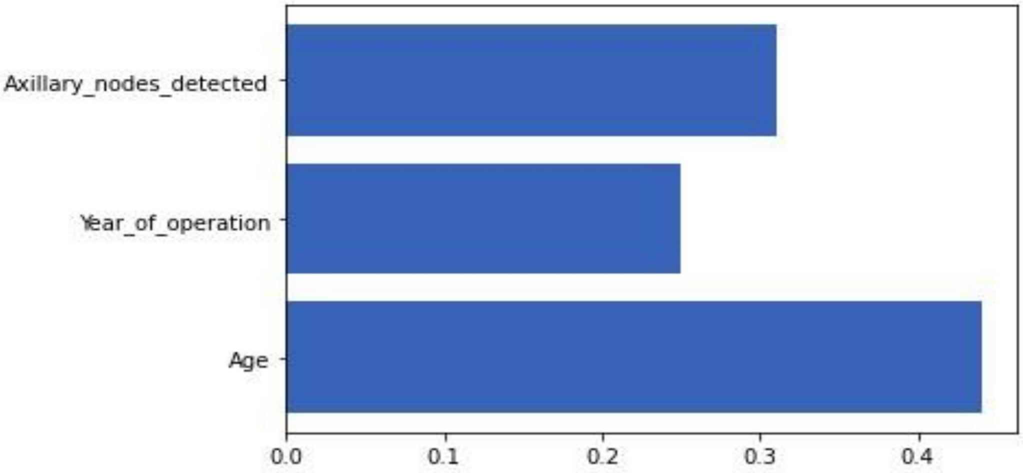
```
TREE
              precision    recall  f1-score   support

           0       0.47      0.53      0.50        15
           1       0.84      0.81      0.83        47

    accuracy                           0.74        62
   macro avg       0.66      0.67      0.66        62
weighted avg       0.75      0.74      0.75        62

KNN
              precision    recall  f1-score   support

           0       0.33      0.20      0.25        15
           1       0.77      0.87      0.82        47

    accuracy                           0.71        62
   macro avg       0.55      0.54      0.54        62
weighted avg       0.67      0.71      0.68        62

SVM
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        15
           1       0.75      0.98      0.85        47

    accuracy                           0.74        62
   macro avg       0.38      0.49      0.43        62
weighted avg       0.57      0.74      0.65        62
```

8. Feature Importance

Different attributes contribute in different proportion to the labels. Using Random Forest as classifier machine, gives the following graph,



```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100)
rf.fit(X_train, Y_train)
print(rf.feature_importances_)
sorted_idx = rf.feature_importances_.argsort()
plt.barh(['Age','Year_of_operation','Axillary_nodes_detected'], rf.feature_importances_)
```

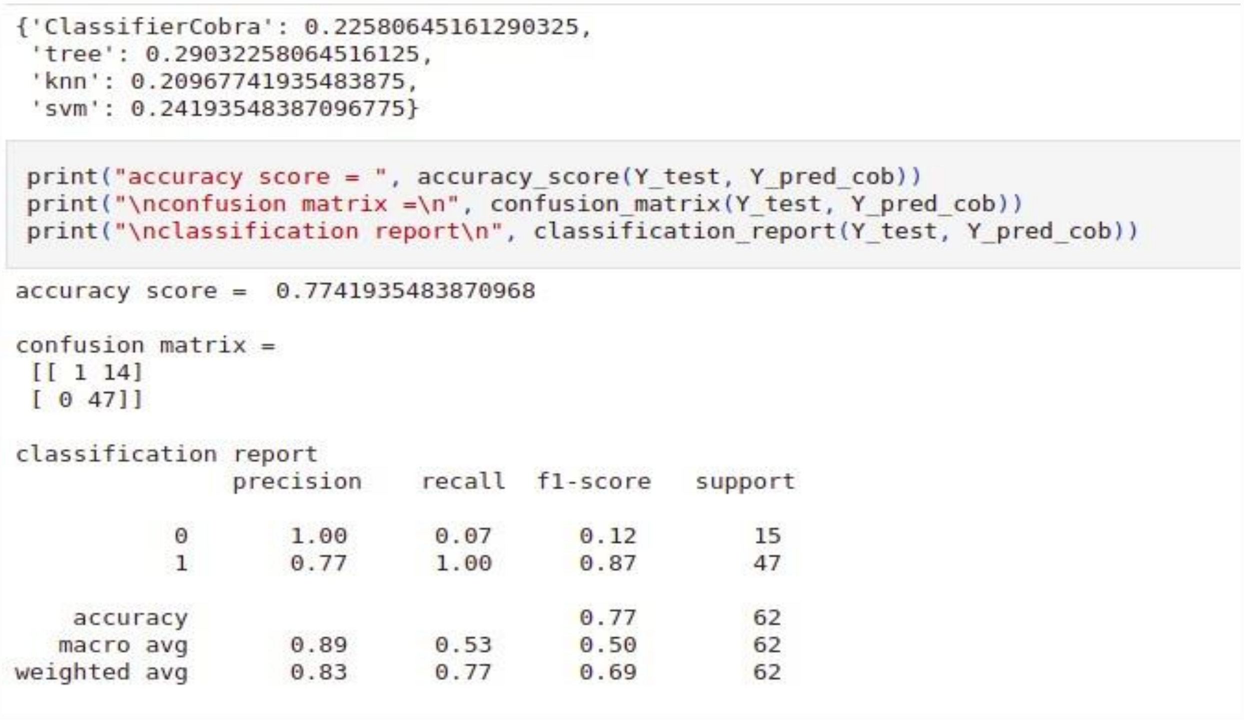


Thus we see that Age is the most prominent attribute(greater than 40 %), while year of operation is the least prominent(close to 25 %).

9. Using the Cobra Classifier

Pycobra is a python library for ensemble learning. It serves as a toolkit for regression and classification using these ensembled machines, and also for visualisation of the performance of the new machine and constituent machines.

Ensembling all the three machines we have used, into pycobra and gives an accuracy close to 77 %.



```
{'ClassifierCobra': 0.22580645161290325,
 'tree': 0.29032258064516125,
 'knn': 0.20967741935483875,
 'svm': 0.24193548387096775}
```

```
print("accuracy score = ", accuracy_score(Y_test, Y_pred_cob))
print("\nconfusion matrix =\n", confusion_matrix(Y_test, Y_pred_cob))
print("\nclassification report\n", classification_report(Y_test, Y_pred_cob))
```

```
accuracy score =  0.7741935483870968

confusion matrix =
 [[ 1 14]
 [ 0 47]]

classification report
               precision    recall  f1-score   support

           0       1.00      0.07      0.12        15
           1       0.77      1.00      0.87        47

    accuracy                           0.77        62
   macro avg       0.89      0.53      0.50        62
weighted avg       0.83      0.77      0.69        62
```