VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

VELLORE INSTITUTE OF TECHNOLOGY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

MACHINE LEARNING

DIGITAL ASSIGNMENT

AYUSHI GHOSH

21MAI0031

M ASHMITHA

21MAI0039

INTRODUCTION:

The problem statement is to find which supervised learning algorithm best suits the liver disease detection dataset. The supervised algorithm used in this problem are – Decision Tree Classifier, Random Forest Classifier, AdaBoost and Support Vector Machine(SVM).

- Decision Tree Classifier:
  The decision tree classifier creates the classification model by building a decision tree. each node among the tree specifies a check on associate attribute, each branch degressive from that node corresponds to one of the achievable values for that attribute.

- Random Forest Classifier:
  A random forest is a meta knowledgeable that matches style of decision tree classifiers on varied sub-samples of the dataset and uses averaging to boost the prognostic accuracy and management over-fitting.

- AdaBoost:
  An AdaBoost classifier might be a meta-estimator that begins by fitting a classifier on the initial dataset then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances unit adjusted nominal future classifiers focus plenty of on robust cases.

- Support Vector Machine (SVM):
  The goal of the SVM algorithmic rule is to create the foremost effective line or decision boundary able to} segregate n-dimensional house into classes thus we have a tendency to square measure able to merely place the new data among the right category among the longer term. This best decision boundary is termed a hyperplane.

METHODOLOGY:

- Dataset:
  Here we have used Liver disease dataset from Kaggle.

First we have to import the dataset and then do preprocessing of data. The preprocessing steps are:

- Encoding the categorical values (eg., Male: 0, Female:1)
- Drop Null values.

Split the data columns into input variables and output variable. And then we are splitting the dataset into training data and testing data. Then we will import the above mention algorithms from sklearn and then apply it on the training dataset. Then we use test dataset to predict using the model that we developed. Then we calculate the accuracy and confusion matrix.

Preprocessing:

```
from google.colab import files
uploaded=files.upload()


import pandas as pd
dataset=pd.read_csv('indian_liver_patient.csv')
dataset=dataset.dropna()


dataset['Gender']=dataset['Gender'].map({'Male':1 , 'Female':0})


dataset.head()


X=dataset.drop('Liver_disease', axis='columns')
y=dataset['Liver_disease']


from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test= train_test_split(X, y, test_size=0.1,
 random_state=40)
```

- Decision Tree Classifier:

  Code:

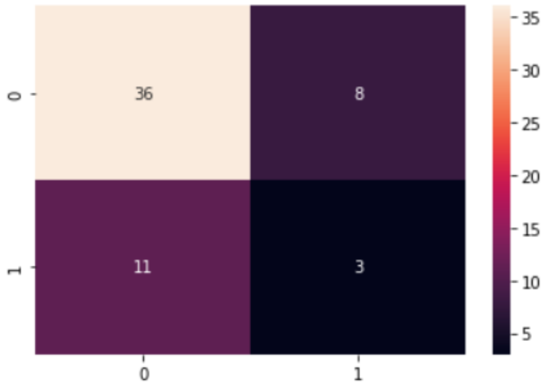
```
from sklearn.tree import DecisionTreeClassifier
clf=DecisionTreeClassifier()
clf.fit(X_train, y_train)


y_pred=clf.predict(X_test)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix, classific
ation_report
accuracy score(y test, y pred)
```

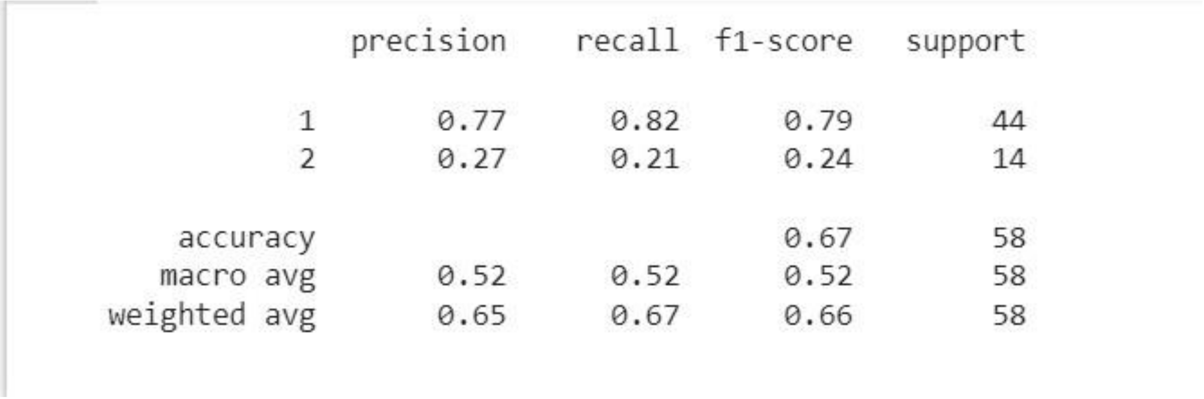
```
0.6724137931034483
```

```
import seaborn as sns
cm=confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2d1d594d90>
```



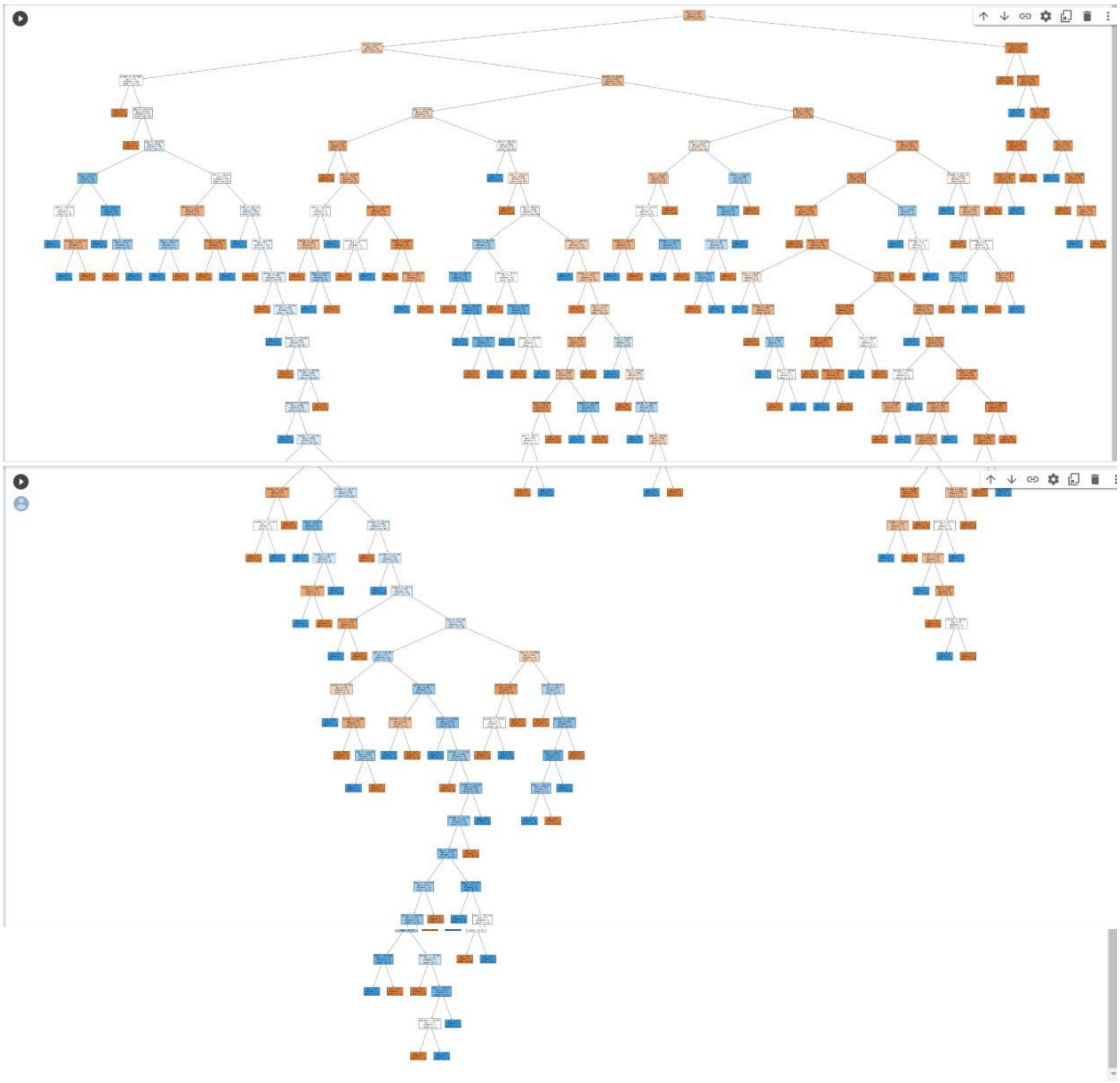
```
print(classification_report(y_test, y_pred))
```



```
              precision    recall  f1-score   support

           1       0.77      0.82      0.79        44
           2       0.27      0.21      0.24        14

    accuracy                           0.67        58
   macro avg       0.52      0.52      0.52        58
weighted avg       0.65      0.67      0.66        58
```

Plotting:

```
import matplotlib.pyplot as plt
from sklearn import tree
fig=plt.figure(figsize=(100,100))
tree=tree.plot_tree(clf,feature_names=X.columns, filled=True)
```

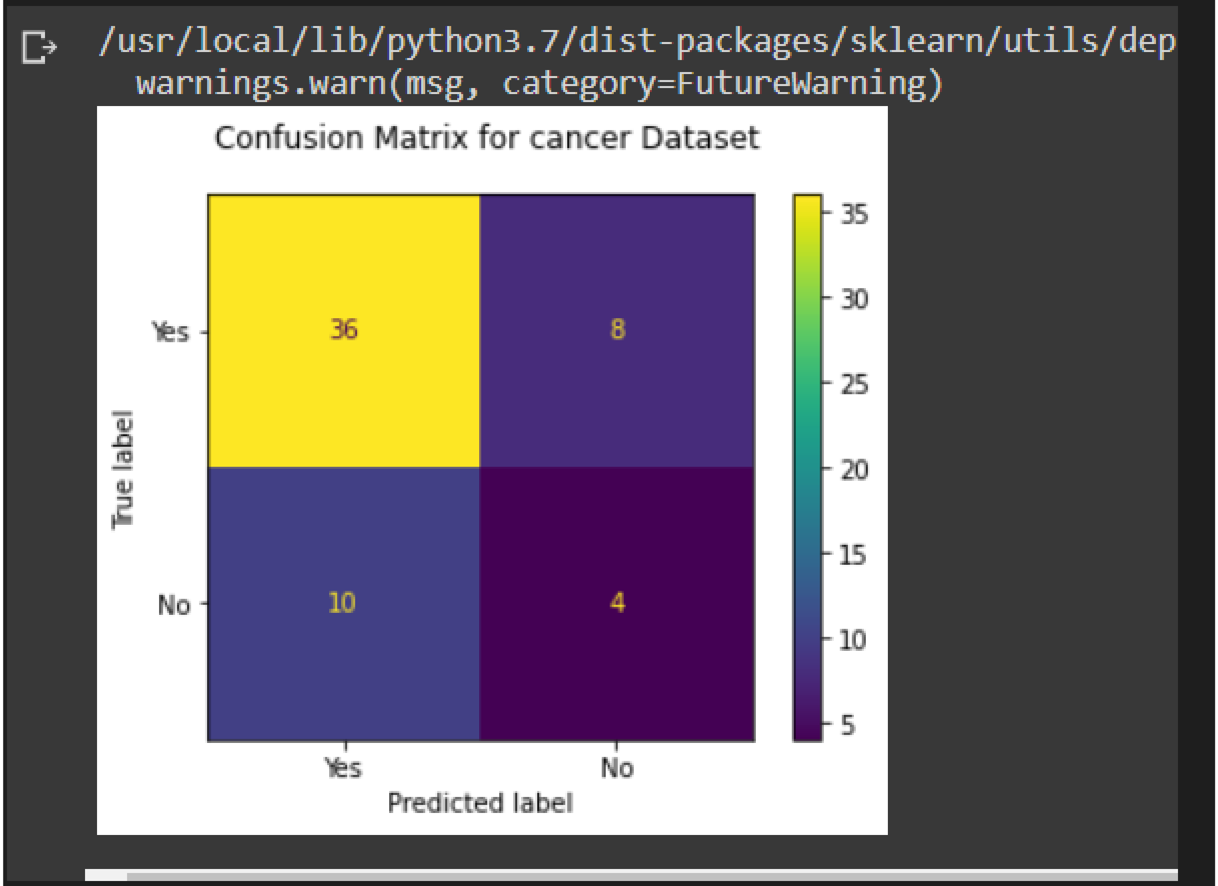


- Random forest Classifier:

Code:

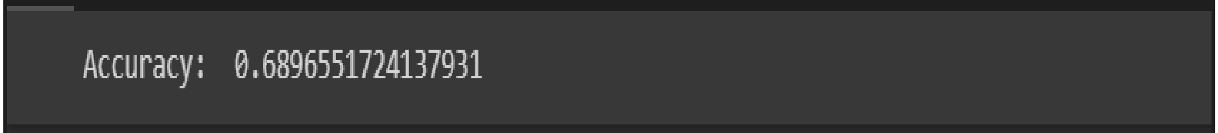
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)


from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier(bootstrap=True, class_weight=None, c
riterion='gini',max_depth=None, max_features='auto', max_leaf_nod
es=None, min_samples_leaf=1, min_samples_split=2, min_weight_frac
tion_leaf=0.0, n_estimators=90, n_jobs=None, oob_score=False, ran
dom_state=None,verbose=0, warm_start=True)
```

```
clf.fit(X_train,y_train)

y_pred = clf.predict(X_test)

#confusion Matrix
from sklearn.metrics import plot_confusion_matrix
fig=plot_confusion_matrix(clf, X_test, y_test,display_labels=["Yes","No
"])
fig.figure_.suptitle("Confusion Matrix for cancer Dataset")
plt.show()
```



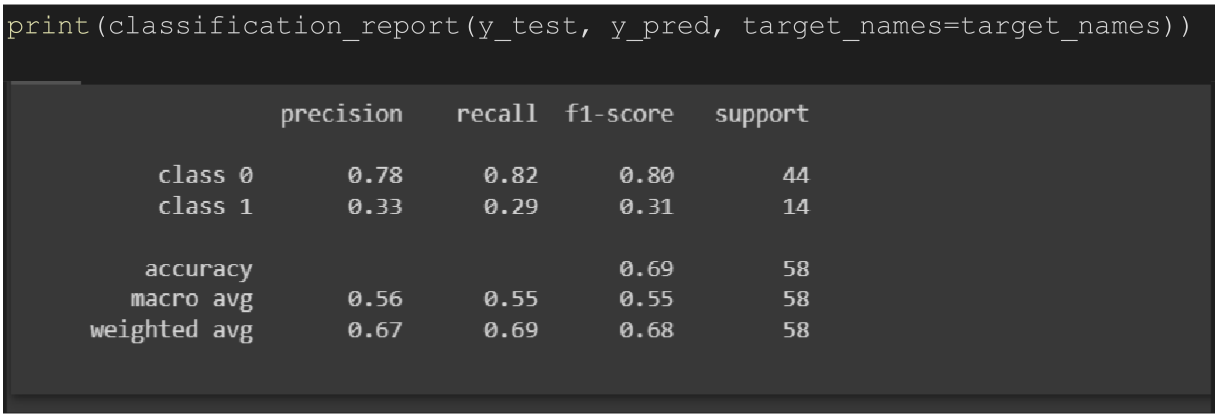
```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/dep
  warnings.warn(msg, category=FutureWarning)
```

```
from sklearn import metrics
print("Accuracy: ",metrics.accuracy_score(y_test,y_pred))
```



```
Accuracy:  0.6896551724137931
```

```
from sklearn.metrics import classification_report
target_names = ['class 0', 'class 1']
```



```
print(classification_report(y_test, y_pred, target_names=target_names))
```

```
              precision    recall  f1-score   support

     class 0       0.78      0.82      0.80        44
     class 1       0.33      0.29      0.31        14

    accuracy                           0.69        58
   macro avg       0.56      0.55      0.55        58
weighted avg       0.67      0.69      0.68        58
```

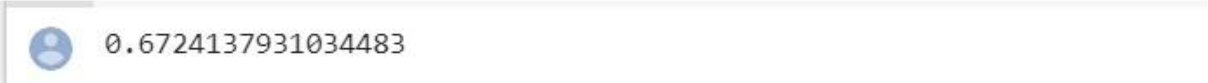
- AdaBoost:

Code:

```
from pandas.core.common import random_state
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
dtc=DecisionTreeClassifier(max_depth=1)
Ada_clf=AdaBoostClassifier(n_estimators=20,base_estimator=dtc,random_state=40)
Ada_clf.fit(X_train, y_train)
```

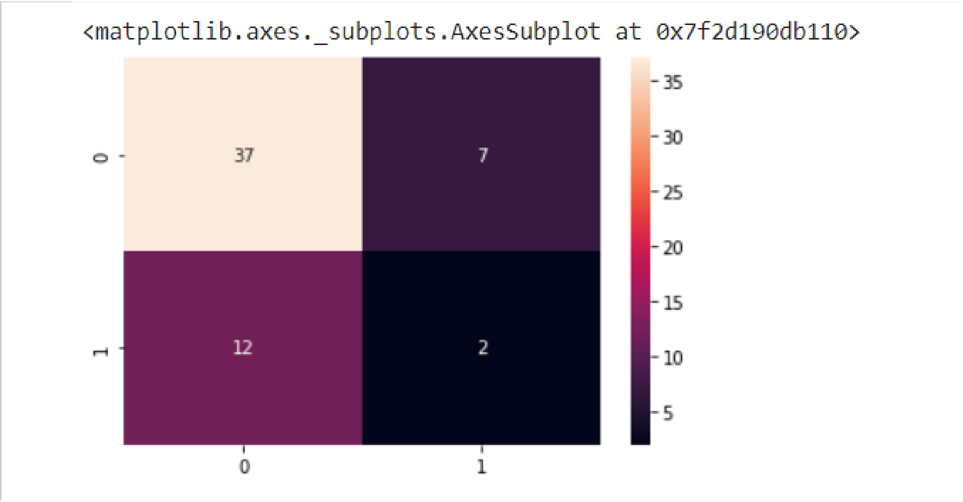
```
y_pred=Ada_clf.predict(X_test)
```

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
accuracy_score(y_test,y_pred)
```



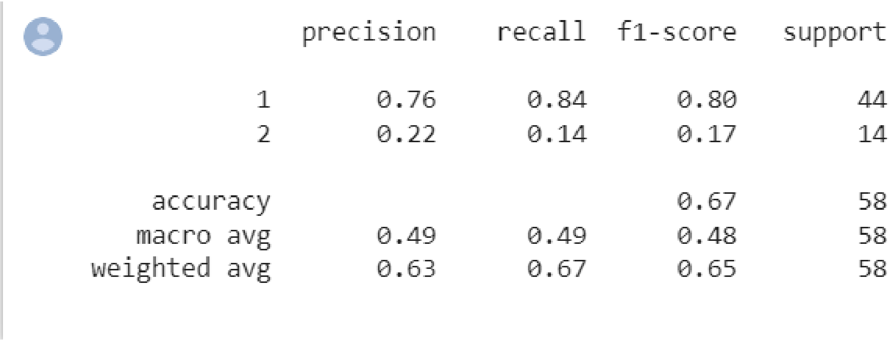
```
0.6724137931034483
```

```
import seaborn as sns

cm = confusion_matrix(y_test,y_pred)
sns.heatmap(cm,annot=True)
```



```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2d190db110>
```

```
print(classification_report(y_test,y_pred))
```



```
              precision    recall  f1-score   support

           1       0.76      0.84      0.80        44
           2       0.22      0.14      0.17        14

    accuracy                           0.67        58
   macro avg       0.49      0.49      0.48        58
weighted avg       0.63      0.67      0.65        58
```

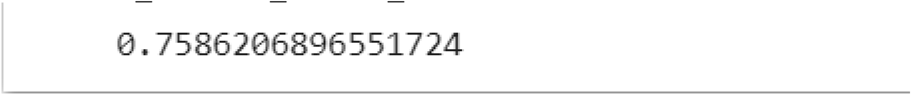
```
#HyperParameter Tuning using GridSearchCV to find the optimal parameter
s
from sklearn.model_selection import GridSearchCV
model=AdaBoostClassifier()
grid=GridSearchCV(estimator=model,param_grid={'n_estimators':range(1,50
)})
grid.fit(X_train,y_train)
```

```
grid.best_params_
```

```
y_pred=grid.predict(X_test)
```

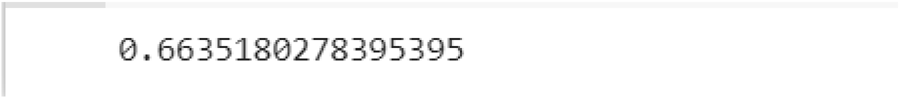
```
from sklearn.ensemble import AdaBoostClassifier
Ada_clf = AdaBoostClassifier(n_estimators=1,random_state=443,learning_r
ate=1)
Ada_clf.fit(X_train,y_train)
```

```
y_pred=Ada_clf.predict(X_test)

accuracy_score(y_test,y_pred)
```



```
0.7586206896551724
```

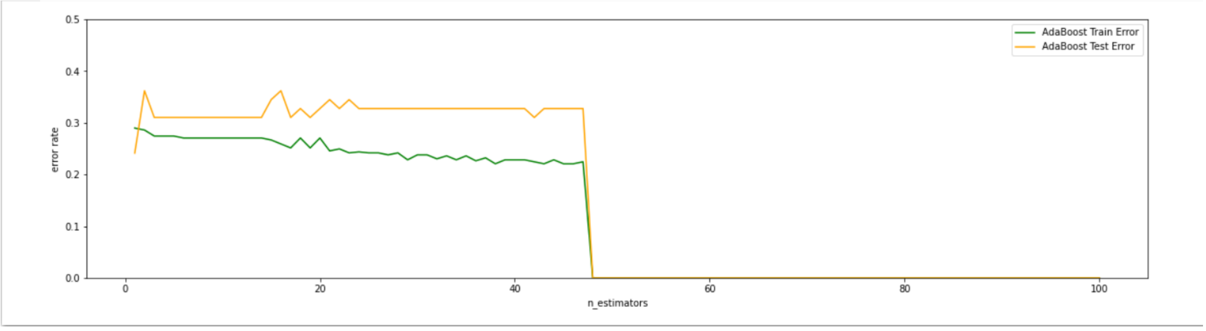
```
from sklearn.metrics import log_loss
loss = log_loss(y_test, Ada_clf.predict_proba(X_test), eps=1e-15)

loss
```



```
0.6635180278395395
```

```
from sklearn.metrics import zero_one_loss
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

n_estimators = 100

ada_real_err = np.zeros((n_estimators,))
for i, y_pred in enumerate(Ada_clf.staged_predict(X_test)):
    ada_real_err[i] = zero_one_loss(y_pred, y_test)

ada_real_err_train = np.zeros((n_estimators,))
for i, y_pred in enumerate(Ada_clf.staged_predict(X_train)):
    ada_real_err_train[i] = zero_one_loss(y_pred, y_train)


fig = plt.figure(figsize=(20,5))
ax = fig.add_subplot(111)
ax.plot(
    np.arange(n_estimators) + 1,
    ada_real_err_train,
    label="AdaBoost Train Error",
    color="green",
)
ax.plot(
    np.arange(n_estimators) + 1,
    ada_real_err,
    label="AdaBoost Test Error",
    color="orange",
)
```

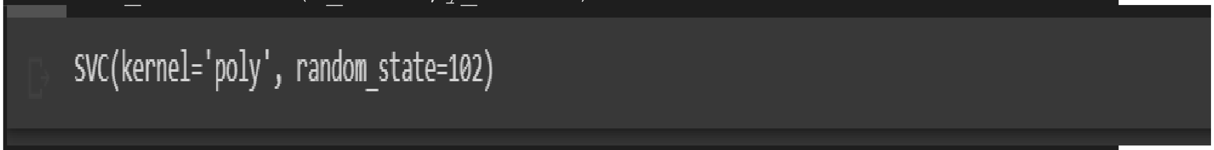
```
ax.set_ylim((0.0, 0.5))
ax.set_xlabel("n_estimators")
ax.set_ylabel("error rate")

leg = ax.legend(loc="upper right", fancybox=True)
leg.get_frame().set_alpha(0.7)

plt.show()
```



- Support Vector Machine:

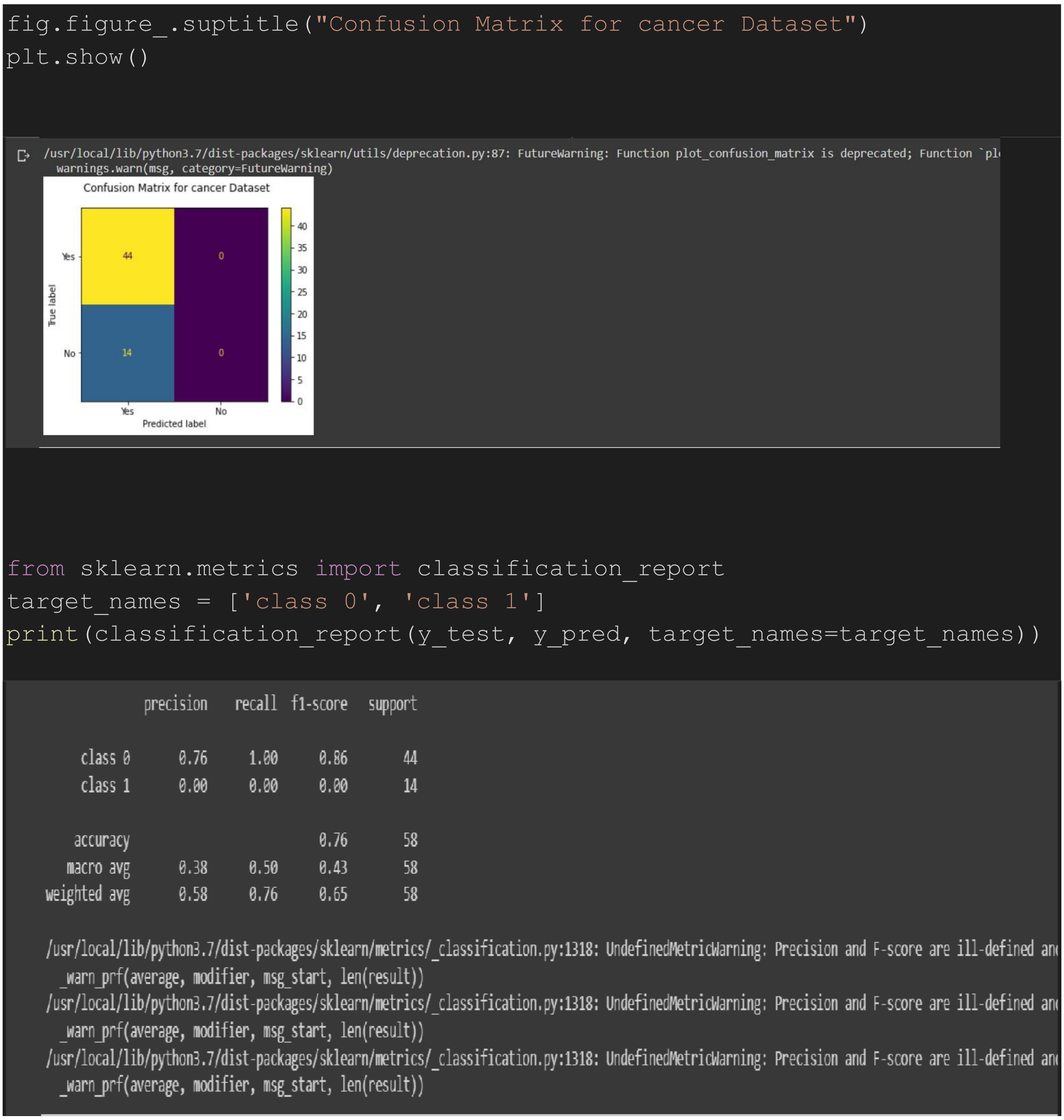
Code:

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_train = pca.fit_transform(X_train)
X_test = pca.fit_transform(X_test)
explained_variance=pca.explained_variance_ratio_


# Fitting SVM to the Training set

from sklearn.svm import SVC
classifier = SVC(kernel = 'poly', random_state = 102)
trained_model=classifier.fit(X_train,y_train)
trained_model.fit(X_train,y_train )
```



```
SVC(kernel='poly', random_state=102)
```

```
# Predicting the Test set results

y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import plot_confusion_matrix
import matplotlib.pyplot as plt
fig=plot_confusion_matrix(trained_model, X_test, y_test,display_labels=
["Yes","No"])
```



```
fig.figure_.suptitle("Confusion Matrix for cancer Dataset")
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is deprecated; Function `pl
  warnings.warn(msg, category=FutureWarning)
```

```
from sklearn.metrics import classification_report
target_names = ['class 0', 'class 1']
print(classification_report(y_test, y_pred, target_names=target_names))
```

```
              precision    recall  f1-score   support

     class 0       0.76      1.00      0.86        44
     class 1       0.00      0.00      0.00        14

    accuracy                           0.76        58
   macro avg       0.38      0.50      0.43        58
weighted avg       0.58      0.76      0.65        58

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetricWarning: Precision and F-score are ill-defined an
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetricWarning: Precision and F-score are ill-defined an
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetricWarning: Precision and F-score are ill-defined an
  _warn_prf(average, modifier, msg_start, len(result))
```

CONCLUSION:

Accuracy for Decision Tree Classifier is 67 % and accuracy for Random Forest is 69% and for AdaBoost (after hyper parameter tuning) is 75.8% and SVM is giving accuracy of 76%. On the whole SVM is best model for the liver disease prediction dataset. This is because of the kernel function being used for classifying the data (as it finds the best hyperplane).

GITHUB LINK:
https://github.com/ayughosh/Machine-Learning