In [4]:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Oct 25 18:00:09 2021

@author: stephaniewatkins
"""

import pandas as pd
import matplotlib.pyplot as plt

#data exploration
bc=pd.read_csv('~/Desktop/DANN862/breastcancer.csv', sep=',')
bc.head()
bc.columns
bc.shape
print(bc.isnull().sum())
bc.describe()
bc.info()
print(bc.describe())
print(bc.corr())
bc.describe()
plt.style.use('classic')
colormap=bc.Classification.factorize()[0]
pd.plotting.scatter_matrix(bc, c = colormap, diagonal = 'kde')

#2
import warnings #because F-test was showing " 0" as warning for linear model
warnings.filterwarnings('ignore')
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
x = bc.iloc[:,0:9]
y = bc.Classification
xtrain, xtest, ytrain, ytest = train_test_split(x,y,test_size=0.3, random_state=
svm_poly = svm.SVC(kernel = 'poly' , degree=2)
svm_poly.fit(xtrain, ytrain)
svm_poly_pred_train = svm_poly.predict(xtrain)
svm_poly_pred_test = svm_poly.predict(xtest)
print('SVM ploy train accuracity is ', accuracy_score(svm_poly_pred_train, ytrai
print(classification_report(svm_poly_pred_train,ytrain))
print('SVM poly bow test accuracy is ', accuracy_score(svm_poly_pred_test,ytest)
print(classification_report(svm_poly_pred_train,ytrain))


svm_rbf = svm.SVC(kernel = 'rbf')
svm_rbf.fit(xtrain,ytrain)
svm_rbf_pred_train = svm_rbf.predict(xtrain)
svm_rbf_pred_test = svm_rbf.predict(xtest)
print('SVM rbf train accuracy is ', accuracy_score(svm_rbf_pred_train, ytrain))
print(classification_report(svm_rbf_pred_train,ytrain))
print('SVM rbf test accuracy is ', accuracy_score(svm_rbf_pred_test, ytest))
print(classification_report(svm_rbf_pred_test,ytest))

svm_lin = svm.SVC(kernel = 'linear')
svm_lin.fit(xtrain, ytrain)
svm_lin_pred_train = svm_lin.predict(xtrain)
svm_lin_pred_test = svm_lin.predict(xtest)
```

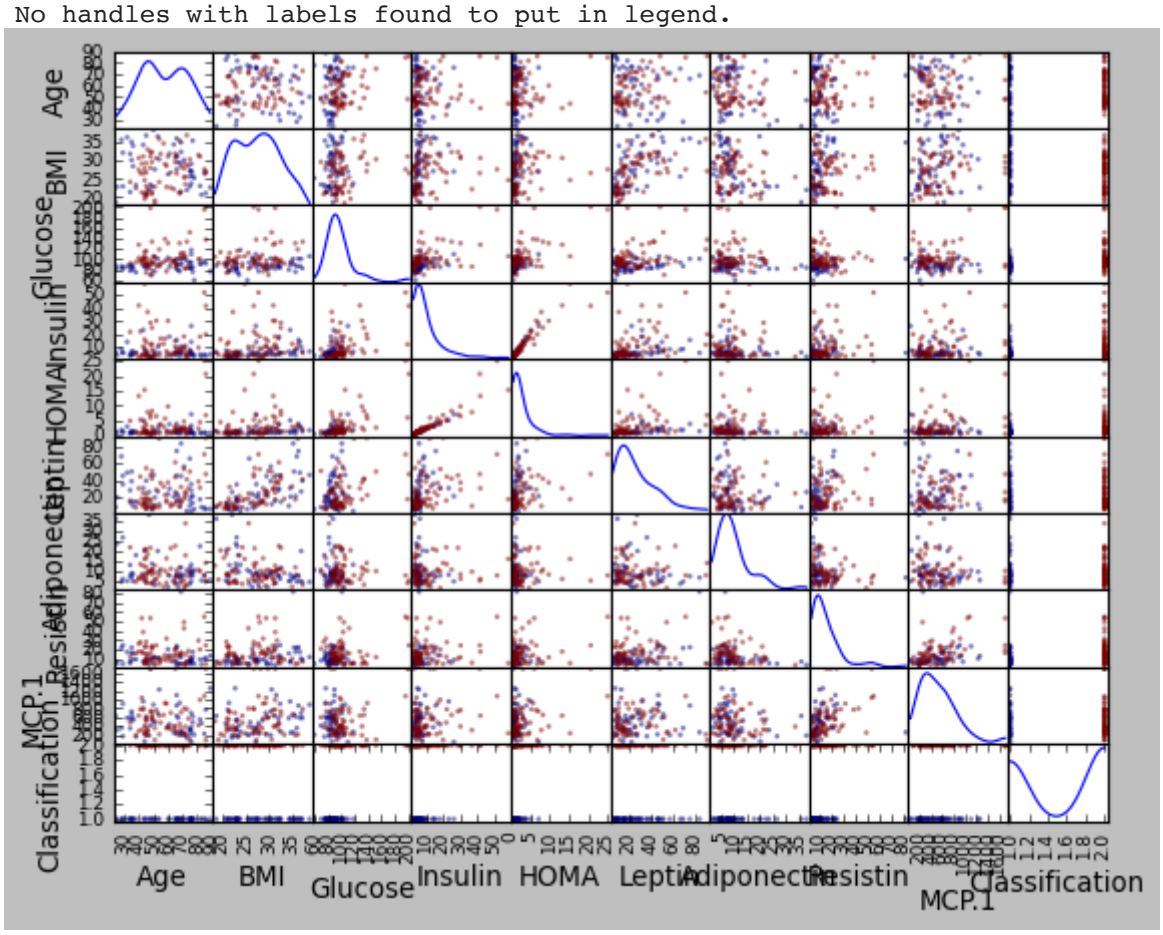
```
print('SVM linear train accuracy is ', accuracy_score(svm_lin_pred_train, ytrain
print(classification_report(svm_lin_pred_train,ytrain))
print('SVM linear test accuracy is ', accuracy_score(svm_lin_pred_test, ytest))
print(classification_report(svm_lin_pred_test,ytest))

#3

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score
RF = RandomForestClassifier(n_estimators = 100, random_state = 0)
RF.fit(xtrain,ytrain)
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
max_depth=None, max_features='auto', max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0, n_estimators=98, n_jobs=None,
oob_score=False, random_state=0, verbose=0, warm_start=False)
RF_pred = RF.predict(xtest)
accuracy_score(RF_pred,ytest)
print(classification_report(ytest,RF_pred))
pd.DataFrame({'feature':bc.columns[1:10],'importance':RF.feature_importances_})
n_estimator = range(2,100,2)
accuracy = []
for i in n_estimator:
    RF = RandomForestClassifier(n_estimators=i, random_state =0)
    scores = cross_val_score(RF,xtrain, ytrain)
    accuracy.append(scores.mean())

plt.figure()
plt.plot(n_estimator, accuracy)
plt.title('Ensemble Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Number of base estimators in ensemble')

import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10,10))
# Creating a bar plot
RF.fit(xtrain,ytrain)
sns.barplot(x=bc.columns[1:10], y=RF.feature_importances_)
# Add labels to your graph
plt.xlabel('Feature Importance Score')
plt.ylabel('Features')
plt.title("Visualizing Features")
plt.legend()
plt.show()
#BMI isbest n-estimator


#4
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
n_estimator = range(1, 50, 1)
accuracy = []
for i in n_estimator:
    ada = AdaBoostClassifier(n_estimators=i, learning_rate = 0.005,
random_state=21)
    scores = cross_val_score(ada, xtrain, ytrain)
    accuracy.append(scores.mean())

plt.figure()
plt.plot(n_estimator, accuracy)
```

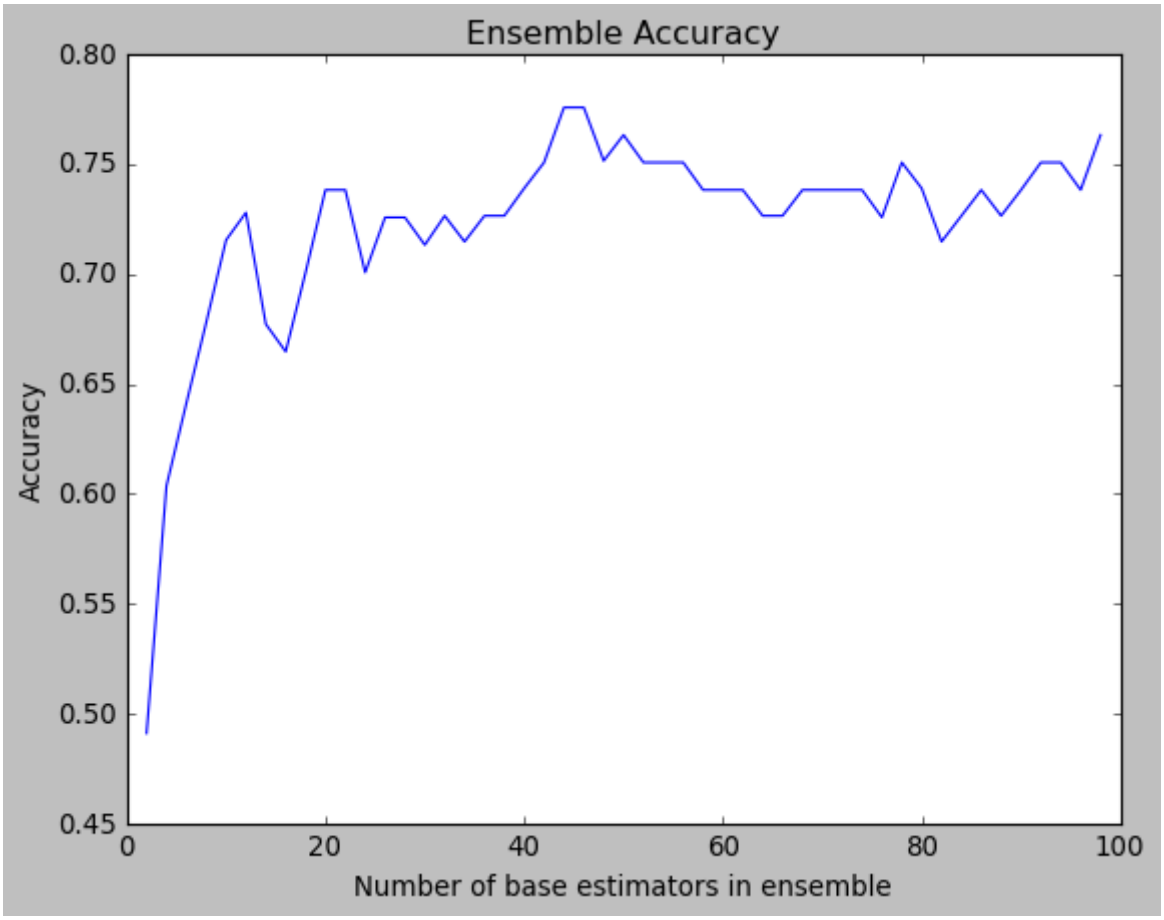
```
plt.title('Adaboost Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Number of base estimators')
```

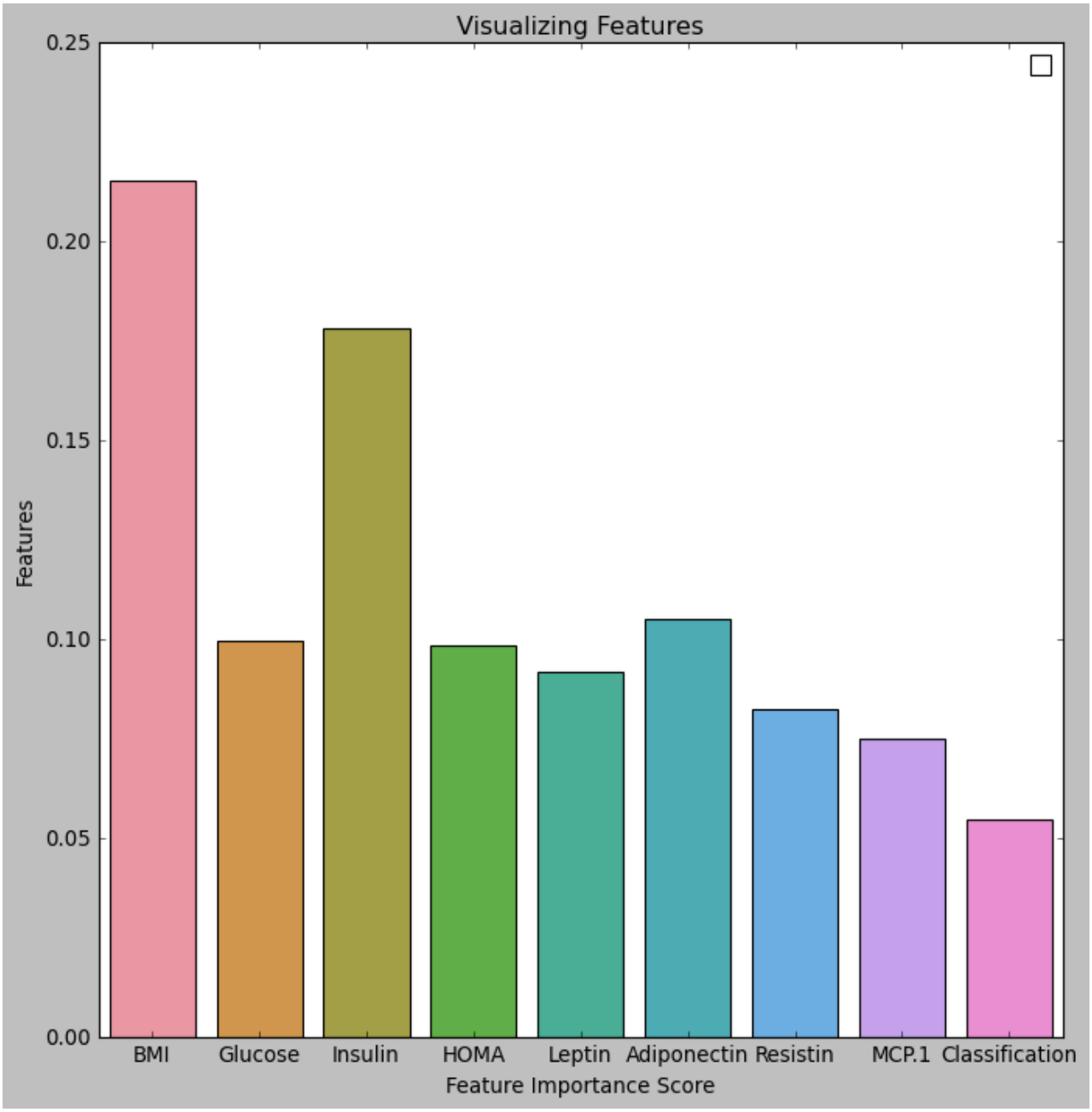
```
/Users/stephaniewatkins/opt/anaconda3/lib/python3.8/site-packages/ipykernel/ipke
rnel.py:287: DeprecationWarning: `should_run_async` will not call `transform_cel
l` automatically in the future. Please pass the result to `transformed_cell` arg
ument and any exception that happen during thetransform in `preprocessing_exc_tu
ple` in IPython 7.17 and above.
  and should_run_async(code)
Age               0
BMI               0
Glucose           0
Insulin           0
HOMA              0
Leptin            0
Adiponectin       0
Resistin          0
MCP.1             0
Classification    0
dtype: int64
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 116 entries, 0 to 115
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             116 non-null    int64
 1   BMI             116 non-null    float64
 2   Glucose         116 non-null    int64
 3   Insulin         116 non-null    float64
 4   HOMA            116 non-null    float64
 5   Leptin          116 non-null    float64
 6   Adiponectin     116 non-null    float64
 7   Resistin        116 non-null    float64
 8   MCP.1           116 non-null    float64
 9   Classification  116 non-null    int64
dtypes: float64(7), int64(3)
memory usage: 9.2 KB
              Age         BMI     Glucose     Insulin        HOMA      Leptin  \
count  116.000000  116.000000  116.000000  116.000000  116.000000  116.000000
mean    57.301724   27.582111   97.793103   10.012086    2.694988   26.615080
std     16.112766    5.020136   22.525162   10.067768    3.642043   19.183294
min     24.000000   18.370000   60.000000    2.432000    0.467409    4.311000
25%     45.000000   22.973205   85.750000    4.359250    0.917966   12.313675
50%     56.000000   27.662416   92.000000    5.924500    1.380939   20.271000
75%     71.000000   31.241442  102.000000   11.189250    2.857787   37.378300
max     89.000000   38.578759  201.000000   58.460000   25.050342   90.280000

       Adiponectin    Resistin        MCP.1  Classification
count   116.000000  116.000000   116.000000      116.000000
mean     10.180874   14.725966   534.647000        1.551724
std       6.843341   12.390646   345.912663        0.499475
min       1.656020    3.210000    45.843000        1.000000
25%       5.474283    6.881763   269.978250        1.000000
50%       8.352692   10.827740   471.322500        2.000000
75%      11.815970   17.755207   700.085000        2.000000
max      38.040000   82.100000  1698.440000        2.000000
                    Age       BMI   Glucose   Insulin      HOMA    Leptin  \
Age            1.000000  0.008530  0.230106  0.032495  0.127033  0.102626
BMI            0.008530  1.000000  0.138845  0.145295  0.114480  0.569593
Glucose        0.230106  0.138845  1.000000  0.504653  0.696212  0.305080
Insulin        0.032495  0.145295  0.504653  1.000000  0.932198  0.301462
HOMA           0.127033  0.114480  0.696212  0.932198  1.000000  0.327210
```

```
Leptin          0.102626  0.569593  0.305080  0.301462  0.327210  1.000000
Adiponectin    -0.219813 -0.302735 -0.122121 -0.031296 -0.056337 -0.095389
Resistin        0.002742  0.195350  0.291327  0.146731  0.231101  0.256234
MCP.1           0.013462  0.224038  0.264879  0.174356  0.259529  0.014009
Classification -0.043555 -0.132586  0.384315  0.276804  0.284012 -0.001078

                Adiponectin  Resistin     MCP.1  Classification
Age               -0.219813  0.002742  0.013462       -0.043555
BMI               -0.302735  0.195350  0.224038       -0.132586
Glucose           -0.122121  0.291327  0.264879        0.384315
Insulin           -0.031296  0.146731  0.174356        0.276804
HOMA              -0.056337  0.231101  0.259529        0.284012
Leptin            -0.095389  0.256234  0.014009       -0.001078
Adiponectin        1.000000 -0.252363 -0.200694       -0.019490
Resistin          -0.252363  1.000000  0.366474        0.227310
MCP.1             -0.200694  0.366474  1.000000        0.091381
Classification    -0.019490  0.227310  0.091381        1.000000
SVM ploy train accuracity is  0.6049382716049383
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         0
           2       1.00      0.60      0.75        81

    accuracy                           0.60        81
   macro avg       0.50      0.30      0.38        81
weighted avg       1.00      0.60      0.75        81

SVM poly bow test accuracy is  0.42857142857142855
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         0
           2       1.00      0.60      0.75        81

    accuracy                           0.60        81
   macro avg       0.50      0.30      0.38        81
weighted avg       1.00      0.60      0.75        81

SVM rbf train accuracy is  0.6049382716049383
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         0
           2       1.00      0.60      0.75        81

    accuracy                           0.60        81
   macro avg       0.50      0.30      0.38        81
weighted avg       1.00      0.60      0.75        81

SVM rbf test accuracy is  0.42857142857142855
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         0
           2       1.00      0.43      0.60        35

    accuracy                           0.43        35
   macro avg       0.50      0.21      0.30        35
weighted avg       1.00      0.43      0.60        35

SVM linear train accuracy is  0.7901234567901234
              precision    recall  f1-score   support

           1       0.75      0.73      0.74        33
           2       0.82      0.83      0.82        48

    accuracy                           0.79        81
   macro avg       0.78      0.78      0.78        81
```

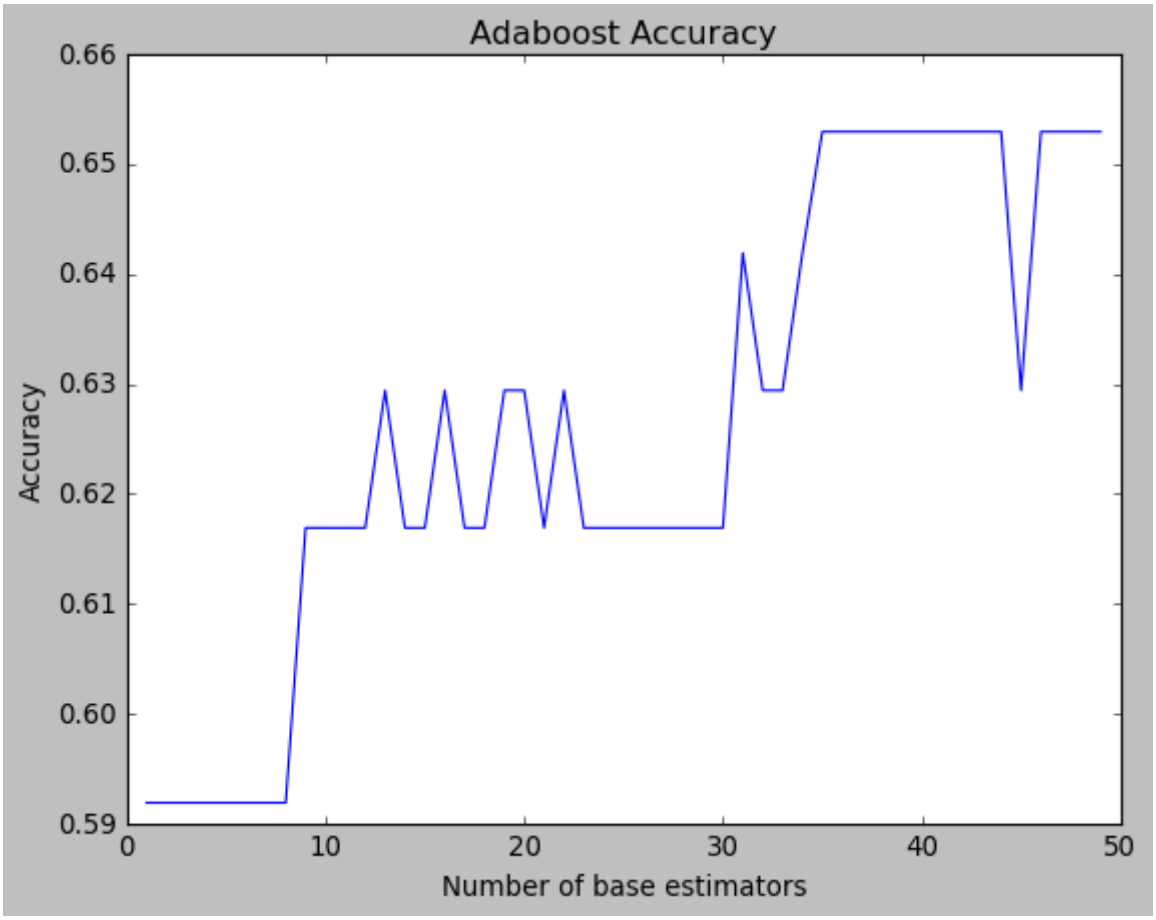
```
weighted avg       0.79      0.79      0.79         81

SVM linear test accuracy is  0.6571428571428571
              precision    recall  f1-score   support

           1       0.60      0.75      0.67         16
           2       0.73      0.58      0.65         19

    accuracy                           0.66         35
   macro avg       0.67      0.66      0.66         35
weighted avg       0.67      0.66      0.66         35

              precision    recall  f1-score   support

           1       0.79      0.55      0.65         20
           2       0.57      0.80      0.67         15

    accuracy                           0.66         35
   macro avg       0.68      0.68      0.66         35
weighted avg       0.69      0.66      0.66         35

No handles with labels found to put in legend.
```




Ensemble Accuracy
Accuracy
Number of base estimators in ensemble
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0
20
40
60
80
100



Out[4]: Text(0.5, 0, 'Number of base estimators')


Adaboost Accuracy
Accuracy
Number of base estimators
0.66
0.65
0.64
0.63
0.62
0.61
0.60
0.59
0
10
20
30
40
50