AdaHR

December 30, 2018

###
Classification of Human Resources Data using AdaBoost Classifier in Python

Burak Koryan | burak@koryan.ca | http://koryan.ca | Date : December 20 2018

0.0.1 Objective:

The aim of this project is to investigate accuracy of AdaBoost classifier on a dataset when classifier parameters are changed and classifying chosen dataset with AdaBoost.

0.0.2 Introduction:

One of my favourite classifiers is AdaBoost and in order to investigate this classifier further,I chose to use it to classify an human resources dataset I found on kaggle.com [1]

The following sentences about the Adaboost algorithm are taken from Wikipedia.org: "AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers " [2].AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms[2].AdaBoost (with decision trees as the weak learners) is often referred to as the best out-of-the-box classifier[2]."

According to SciKit Learn,the Adaboost classifier is explained as follows: "An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.[4]"

The dataset used in classification has multi-column data about employees of a company.The dataset has detailed information,such as pay rate,age of employees,state the employee is from.However,in order to make classification easier,only three parameters were selected from the dataset : age,pay rate,and sex of 310 employees in total.In line 3 of the source code below,what the dataset looks like can be seen.

In the next section below,the source code is shown and its output is printed step by step.Multiple plots of the classifier output can be seen when the number of trees changed from 1 to 30 step by step in order to examine the classifier output'plots in detail.To support this,classifier accuracy and the confusion matrix of related plot is given as well.

```
In [6]: # Importing sklearn libraries

        import numpy as np
        import matplotlib.pyplot as plt
```

```
import pandas as pd
import numpy as np
import random
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.cross_validation import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import confusion_matrix
from matplotlib.colors import ListedColormap
from numpy import array
```

In [7]:
```
# Import the dataset into "dataset" variable from the CSV file and assigning "age" and
# "pay rate" data into X variable,"sex" data into Y variable

dataset = pd.read_csv('HRDataset.csv')
X = dataset.iloc[1:310,[8,9]].values
Y = dataset.iloc[1:310,13].values
```

In [3]:
```
# Splitting dataset into test and training sets.The test set size is 40% of total numb
# In this case the number of elements in the CSV file is 310.This means that the test

X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size = 0.3,random_state = 5)

print("\nfirst 5 elements of X_train:")
print(X_train[0:5])

print("\nfirst 5 elements of X_test:")
print(X_test[0:5])

print("\nfirst 5 elements of Y_test:")
print(Y_test[0:5])

print("\nfirst 5 elements of Y_train:")
print(Y_train[0:5])
```

```
first 5 elements of X_train:
[[33. 22.]
 [39. 19.]
 [30. 43.]
 [28. 57.]
 [43. 24.]]

first 5 elements of X_test:
[[47.  56. ]
 [28.  45. ]
```

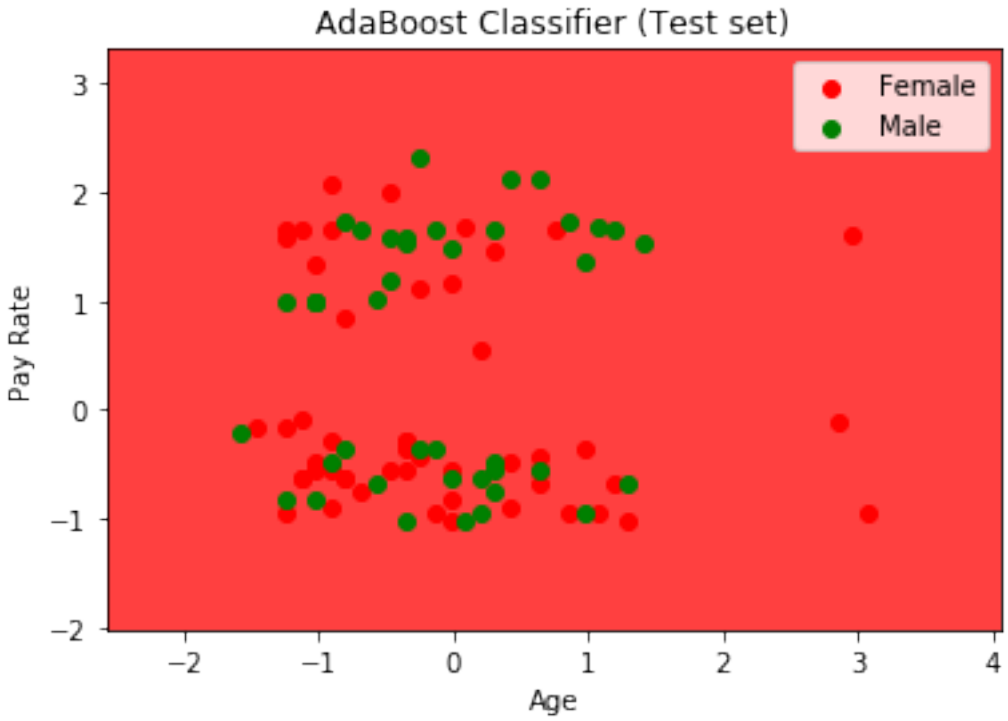
```
 [30.  45. ]
 [35.  53.8]
 [50.  55. ]]

first 5 elements of Y_test:
['Male' 'Male' 'Male' 'Male' 'Male']

first 5 elements of Y_train:
['Female' 'Female' 'Male' 'Female' 'Female']
```

```
In [4]: # Fit and transform data in X_train and X_test
        print("\nfirst 5 elements of X_train before fitting and transforming:")
        print(X_train[0:5])
        print("\nfirst 5 elements of X_test before fitting and transforming:")
        print(X_test[0:5])

        sc = StandardScaler()
        X_train = sc.fit_transform(X_train)
        X_test = sc.transform(X_test)

        print("\nfirst 5 elements of X_train after fitting and transforming:")
        print(X_train[0:5])
        print("\nfirst 5 elements of X_test after fitting and transforming:")
        print(X_test[0:5])
```

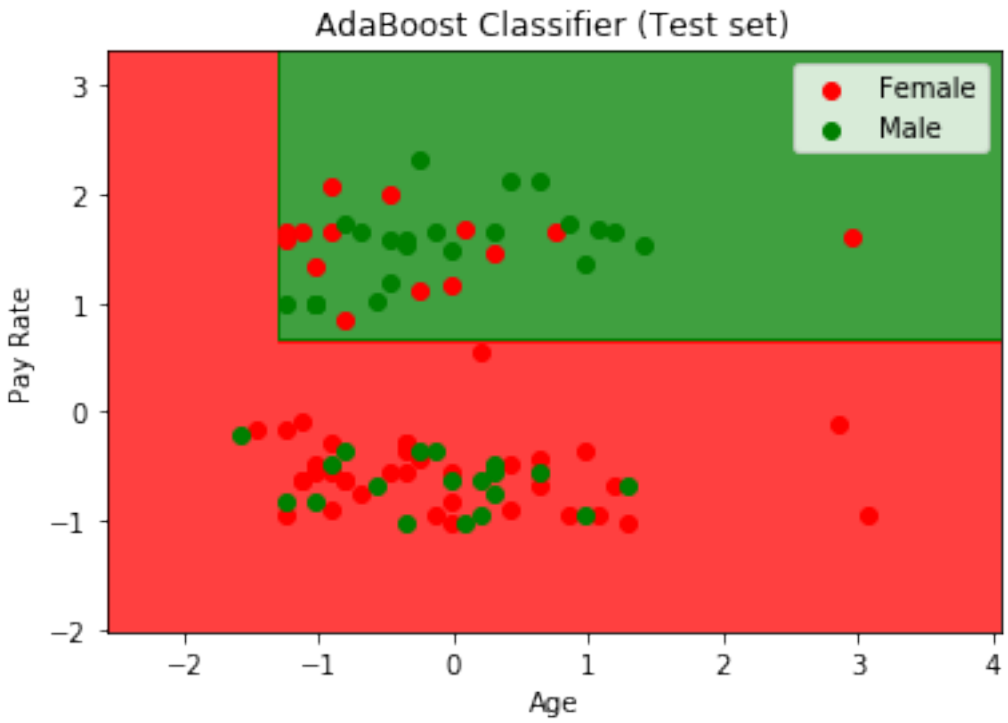
```
first 5 elements of X_train before fitting and transforming:
[[33. 22.]
 [39. 19.]
 [30. 43.]
 [28. 57.]
 [43. 24.]]

first 5 elements of X_test before fitting and transforming:
[[47.  56. ]
 [28.  45. ]
 [30.  45. ]
 [35.  53.8]
 [50.  55. ]]

first 5 elements of X_train after fitting and transforming:
[[-0.68841585 -0.55268387]
 [-0.02557265 -0.75362375]
 [-1.01983745  0.85389522]
 [-1.24078519  1.79161462]
 [ 0.41632281 -0.41872396]]
```

```
first 5 elements of X_test after fitting and transforming:
[[ 0.85821828  1.72463466]
 [-1.24078519  0.98785514]
 [-1.01983745  0.98785514]
 [-0.46746812  1.57727876]
 [ 1.18963988  1.65765471]]


In [5]: for num in range(1,30):
            random.seed(1000)
            classifier = AdaBoostClassifier(n_estimators = num,learning_rate = 0.9)
            classifier.fit(X_train,Y_train)
            y_pred = classifier.predict(X_test)


            X_set, y_set = X_test, Y_test

            X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].ma
                                 np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].ma


            class_pred = classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.sh

            for x in range(0,664):
                for y in range(0,535):
                    if class_pred[y,x] == 'Male':
                        class_pred[y,x] = '1'
                    else:
                        class_pred[y,x] = '0'

            plt.contourf(X1, X2, class_pred,alpha = 0.75, cmap = ListedColormap(('red', 'green

            plt.xlim(X1.min(), X1.max())
            plt.ylim(X2.min(), X2.max())
            for i, j in enumerate(np.unique(y_set)):
                plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                            c = ListedColormap(('red', 'green'))(i), label = j)
            plt.title('AdaBoost Classifier (Test set)')
            plt.xlabel('Age')
            plt.ylabel('Pay Rate')
            plt.legend()
            plt.show()
            cm = confusion_matrix(Y_test,y_pred)

            print("\nthe number of trees:")
            print(num)
            print("\n")
            print("Accuracy of classifier(%):")
```

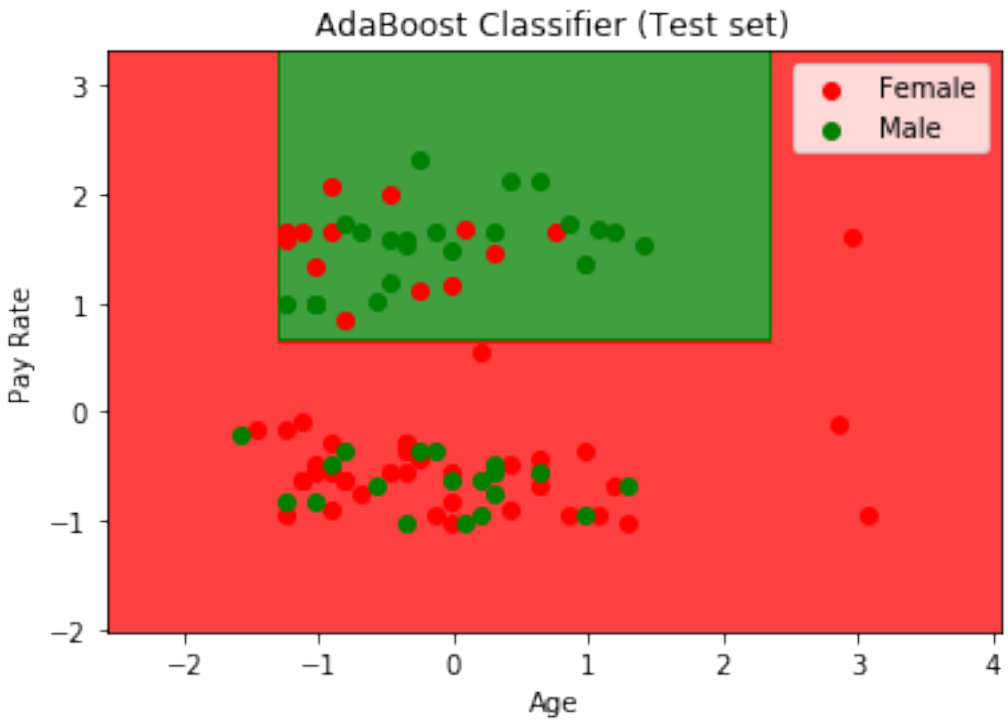
```
print(accuracy_score(Y_test, y_pred)*100)
print("\n")
print("Confusion matrix of the classification:")
print(cm)
```



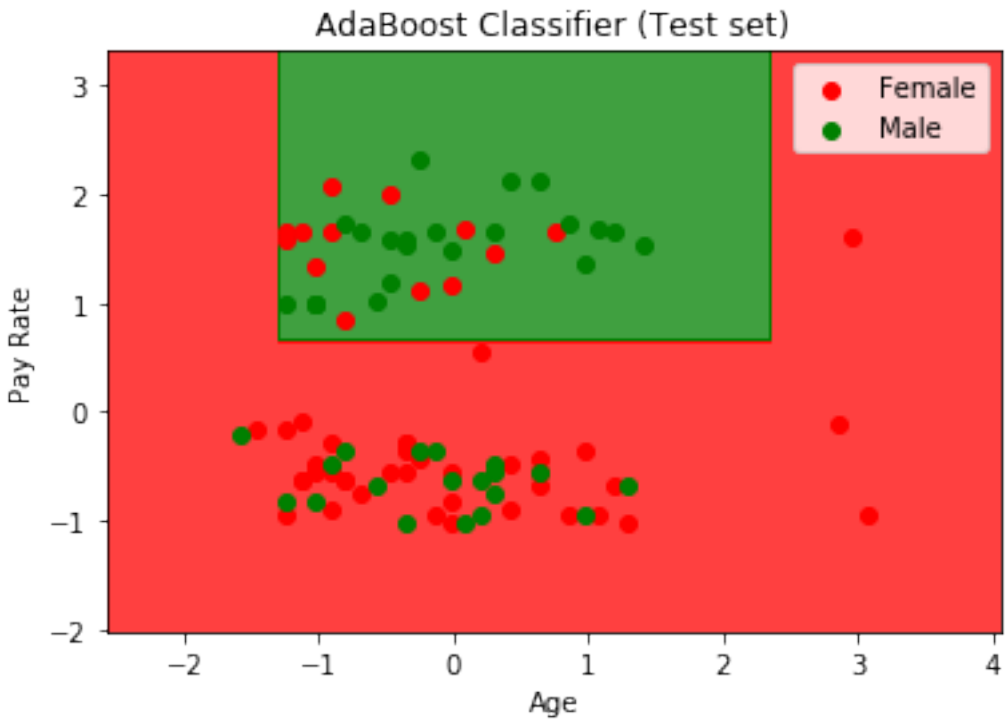

```
the number of trees:
1


Accuracy of classifier:
55.91397849462365


Confusion matrix of the classification:
[[52  0]
 [41  0]]
```



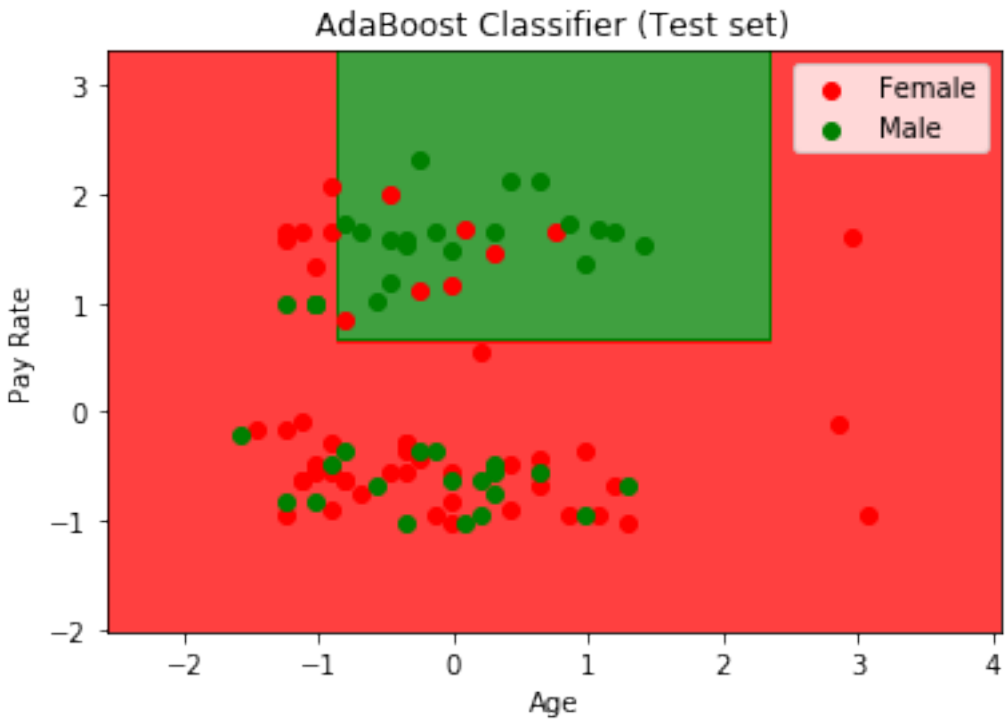

```
the number of trees:
2


Accuracy of classifier:
64.51612903225806


Confusion matrix of the classification:
[[38 14]
 [19 22]]
```



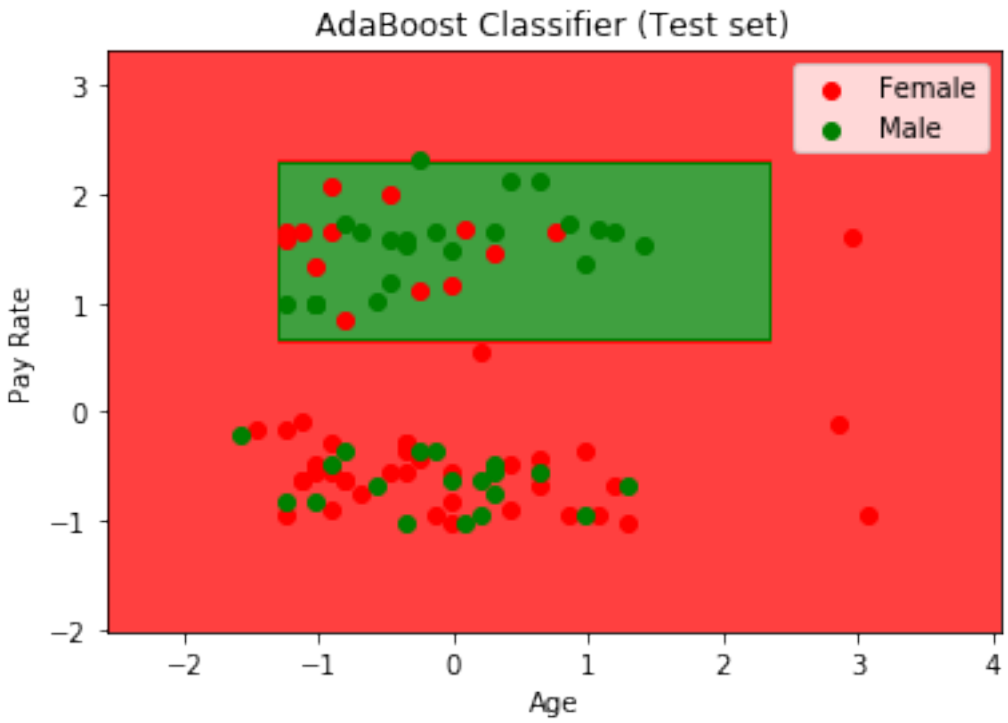

```
the number of trees:
3


Accuracy of classifier:
65.59139784946237


Confusion matrix of the classification:
[[39 13]
 [19 22]]
```



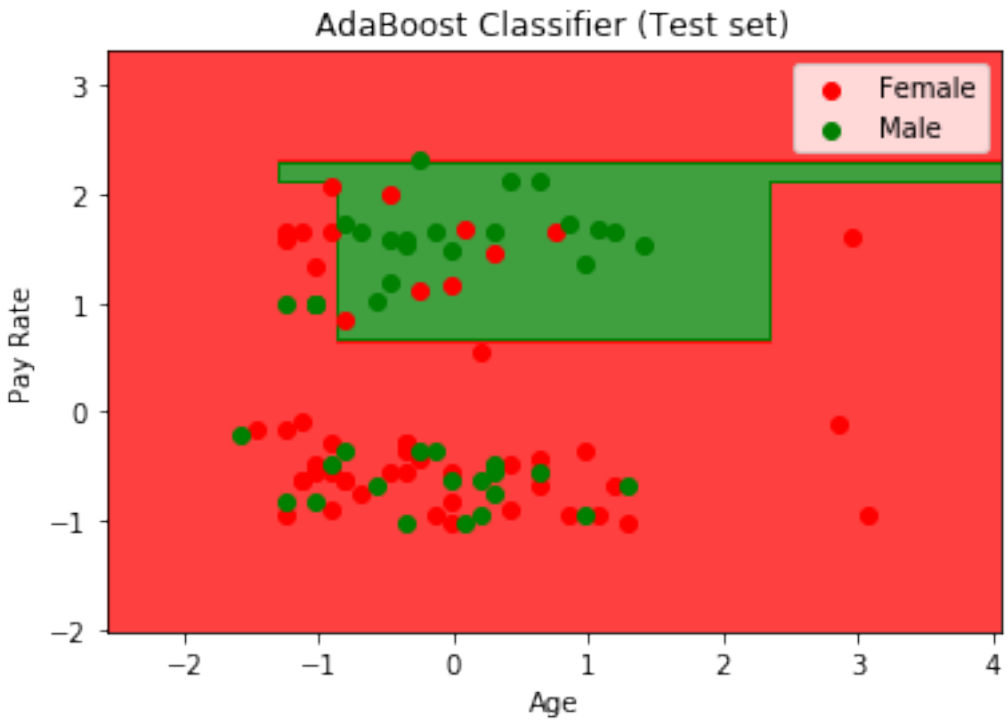

```
the number of trees:
4


Accuracy of classifier:
65.59139784946237


Confusion matrix of the classification:
[[39 13]
 [19 22]]
```



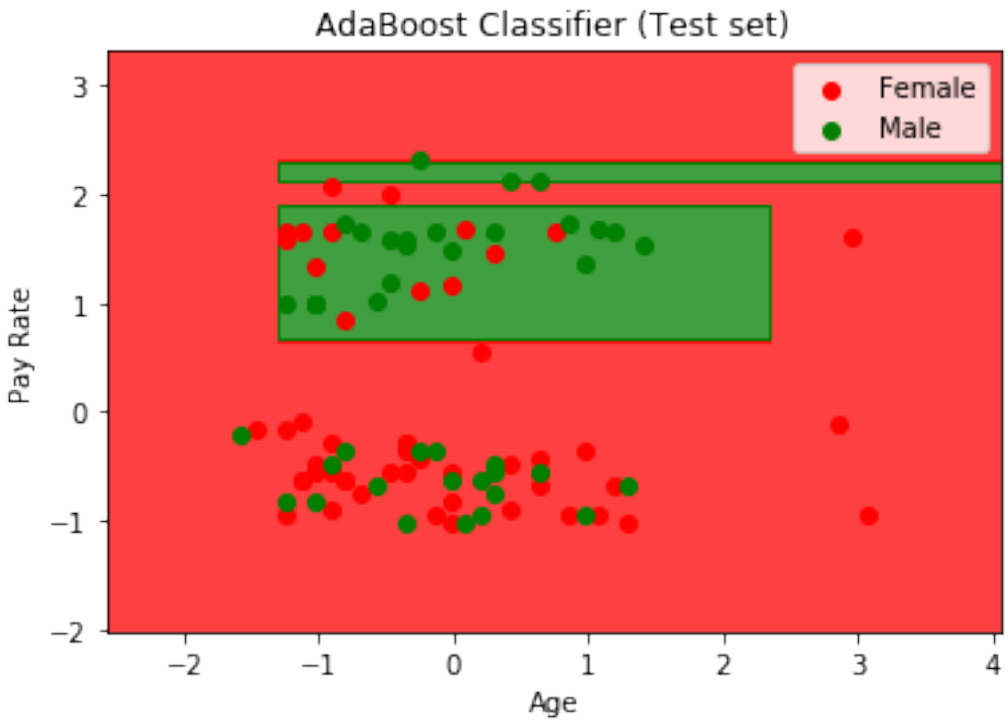

```
the number of trees:
5


Accuracy of classifier:
67.74193548387096


Confusion matrix of the classification:
[[45  7]
 [23 18]]
```



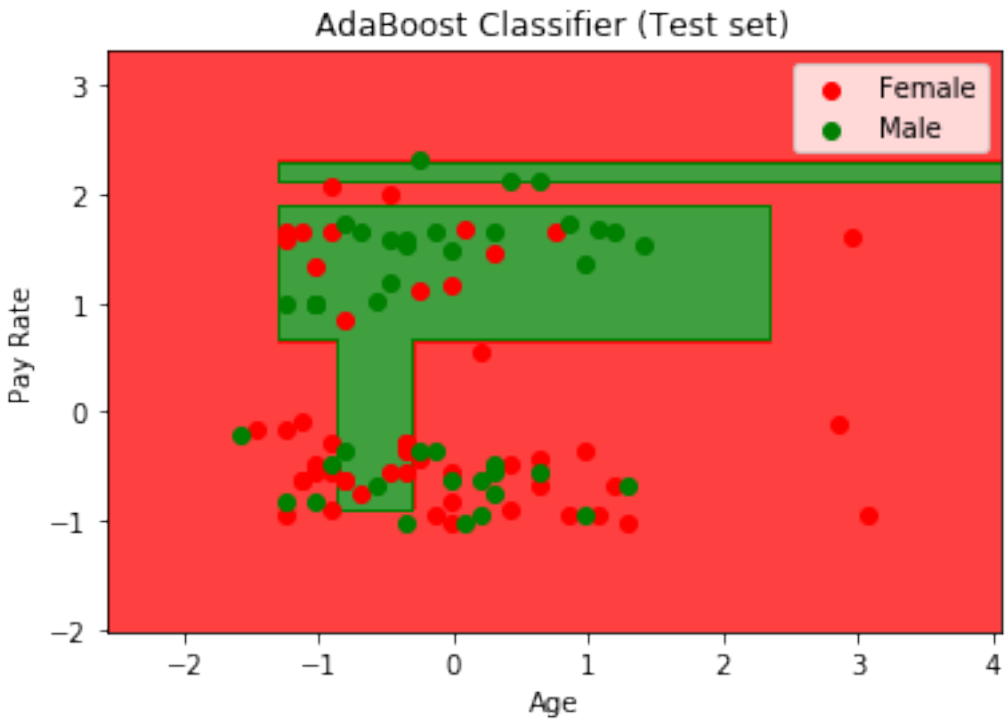

```
the number of trees:
6


Accuracy of classifier:
64.51612903225806


Confusion matrix of the classification:
[[39 13]
 [20 21]]
```



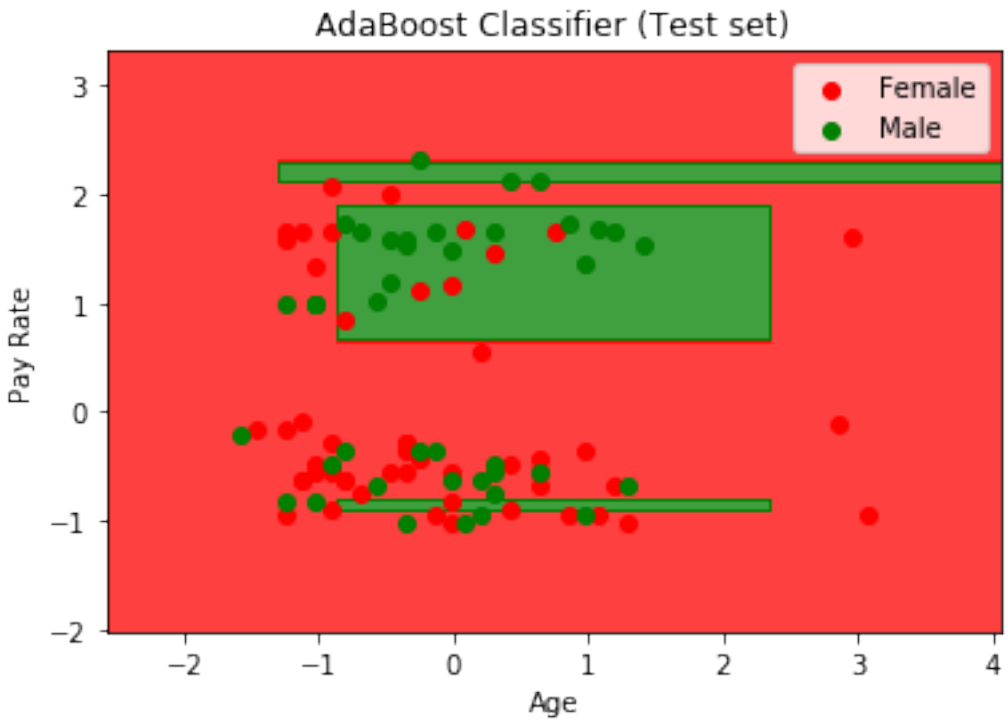

```
the number of trees:
7


Accuracy of classifier:
66.66666666666666


Confusion matrix of the classification:
[[45  7]
 [24 17]]
```



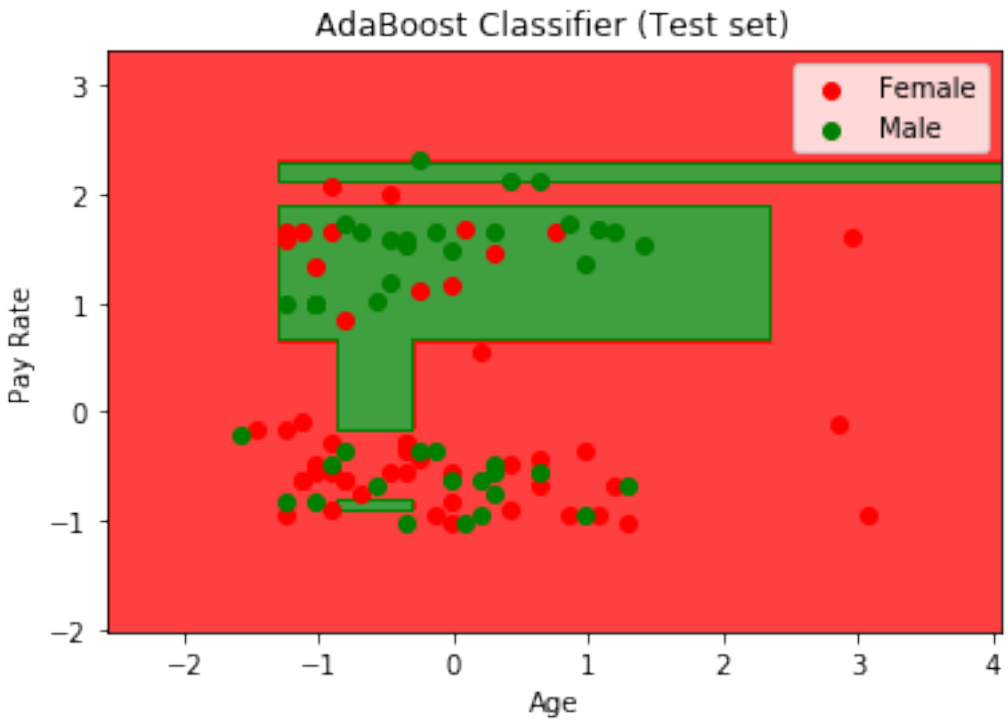

```
the number of trees:
8


Accuracy of classifier:
66.66666666666666


Confusion matrix of the classification:
[[41 11]
 [20 21]]
```



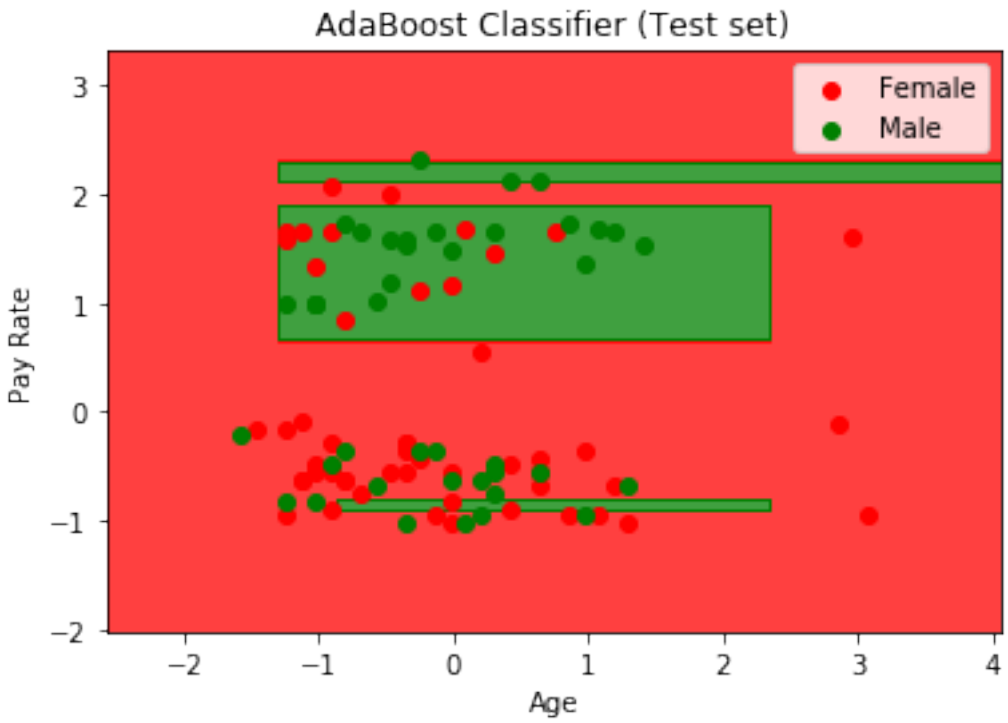

```
the number of trees:
9


Accuracy of classifier:
60.215053763440864


Confusion matrix of the classification:
[[33 19]
 [18 23]]
```



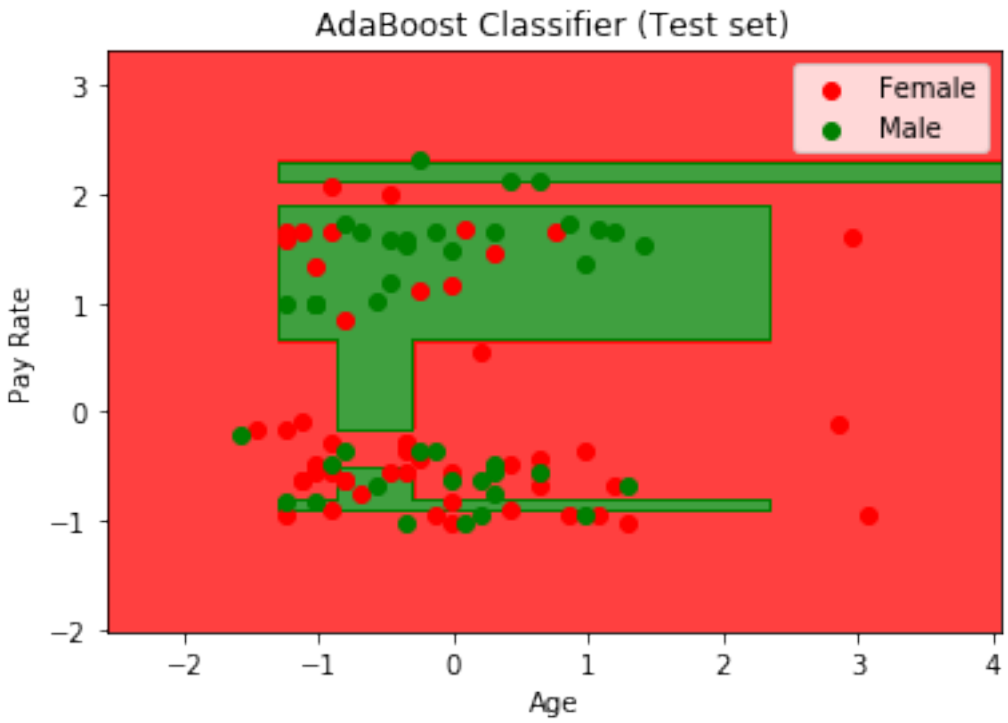

```
the number of trees:
10


Accuracy of classifier:
65.59139784946237


Confusion matrix of the classification:
[[44  8]
 [24 17]]
```



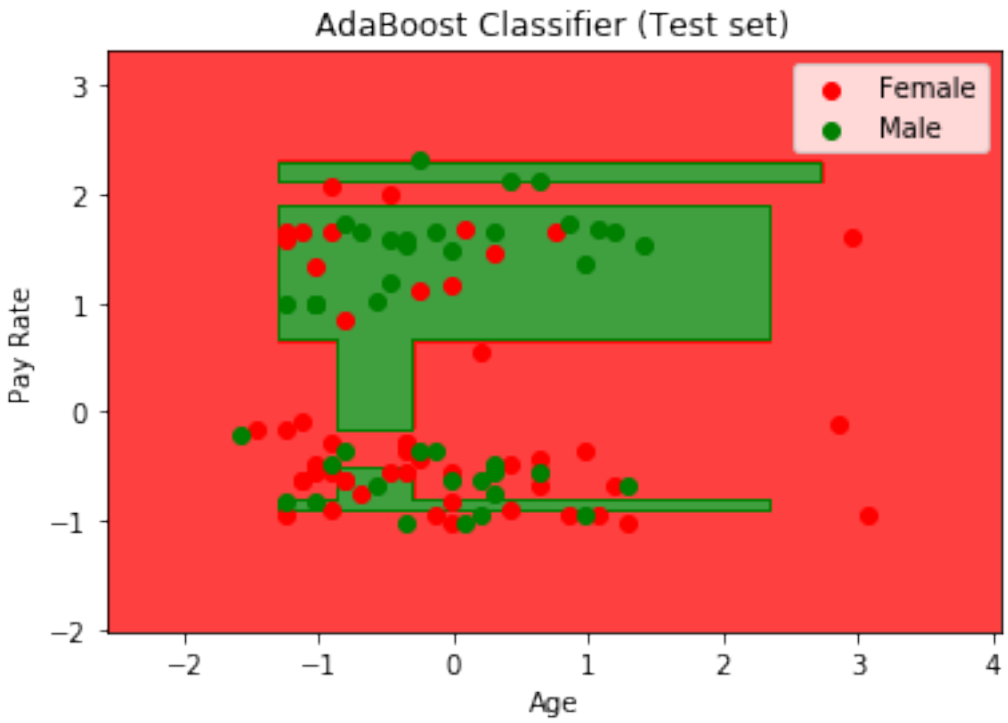

```
the number of trees:
11


Accuracy of classifier:
66.66666666666666


Confusion matrix of the classification:
[[41 11]
 [20 21]]
```



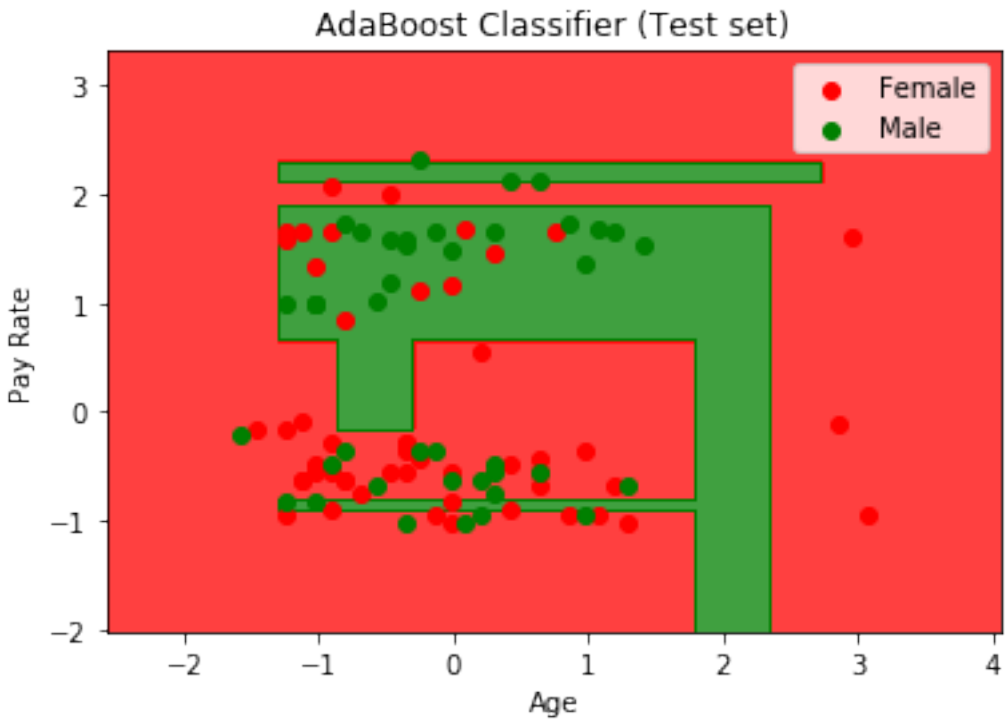

```
the number of trees:
12


Accuracy of classifier:
64.51612903225806


Confusion matrix of the classification:
[[39 13]
 [20 21]]
```



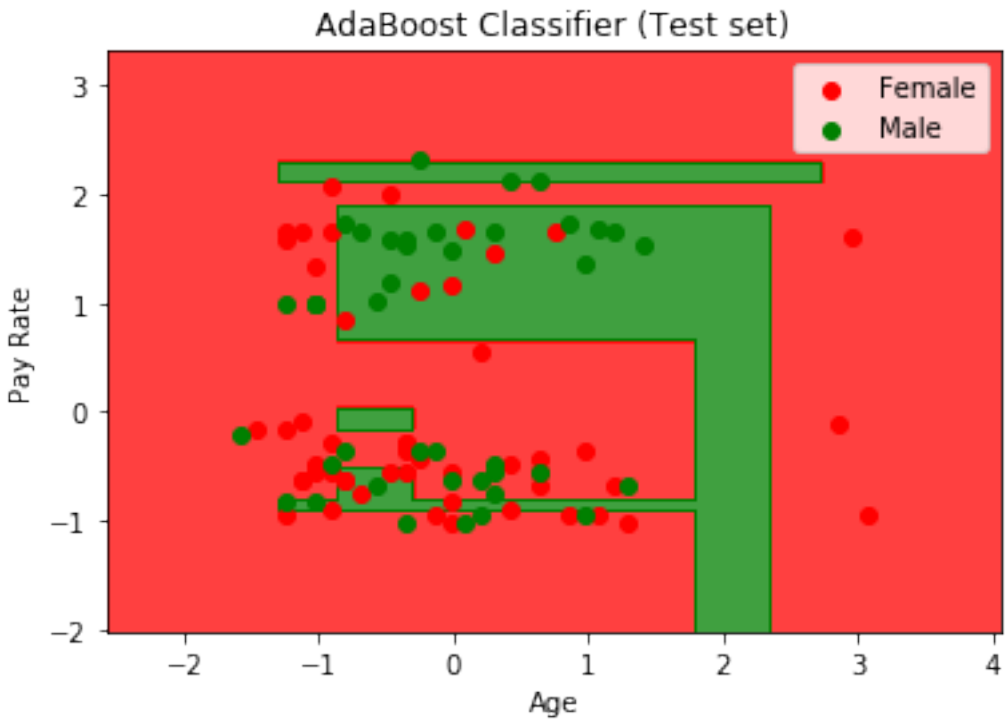

```
the number of trees:
13


Accuracy of classifier:
61.29032258064516


Confusion matrix of the classification:
[[33 19]
 [17 24]]
```



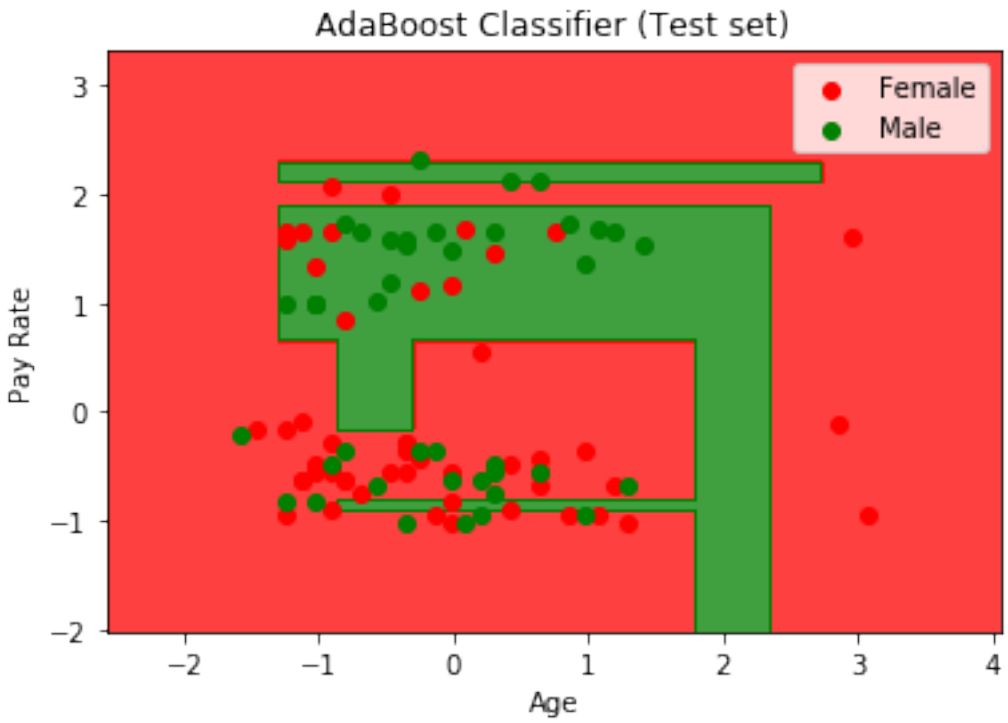

```
the number of trees:
14


Accuracy of classifier:
61.29032258064516


Confusion matrix of the classification:
[[33 19]
 [17 24]]
```



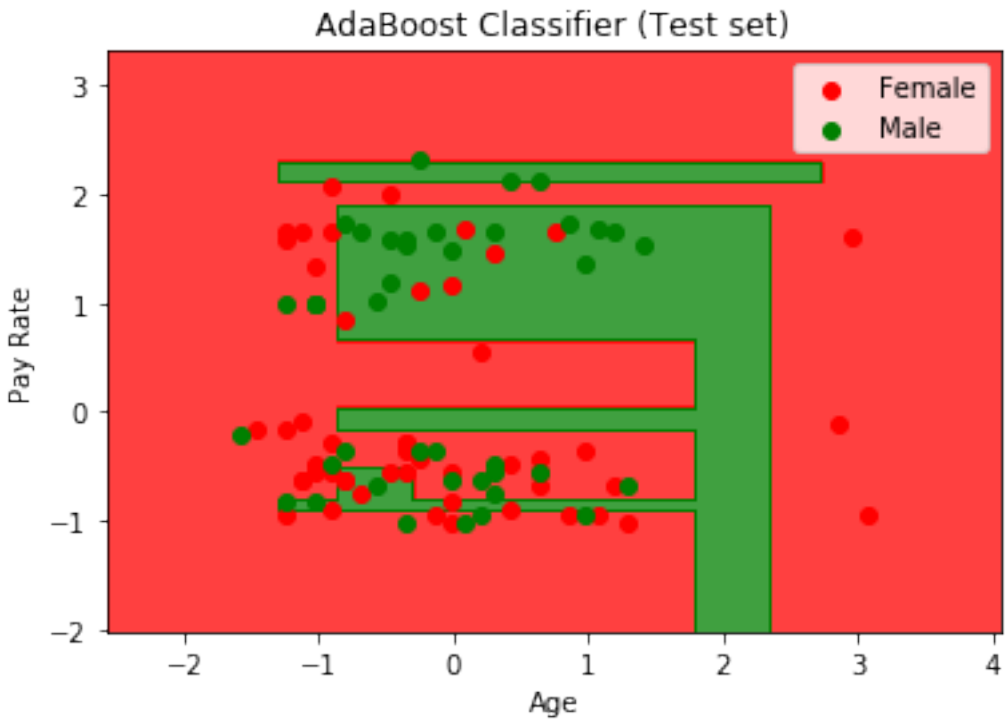

```
the number of trees:
15


Accuracy of classifier:
65.59139784946237


Confusion matrix of the classification:
[[38 14]
 [18 23]]
```



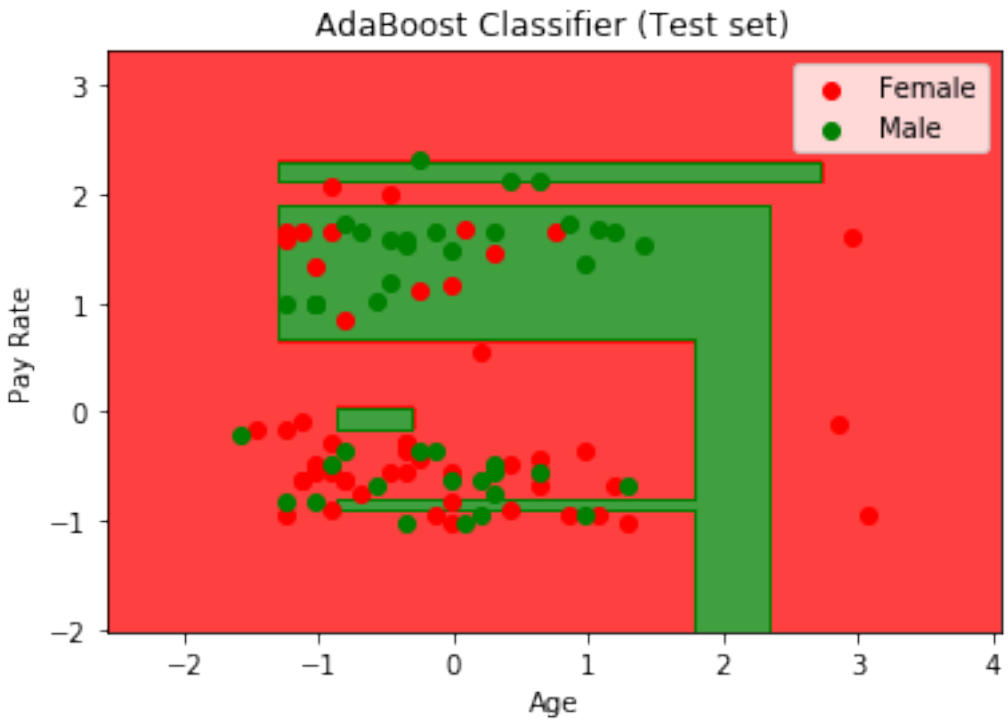

```
the number of trees:
16


Accuracy of classifier:
62.365591397849464


Confusion matrix of the classification:
[[38 14]
 [21 20]]
```



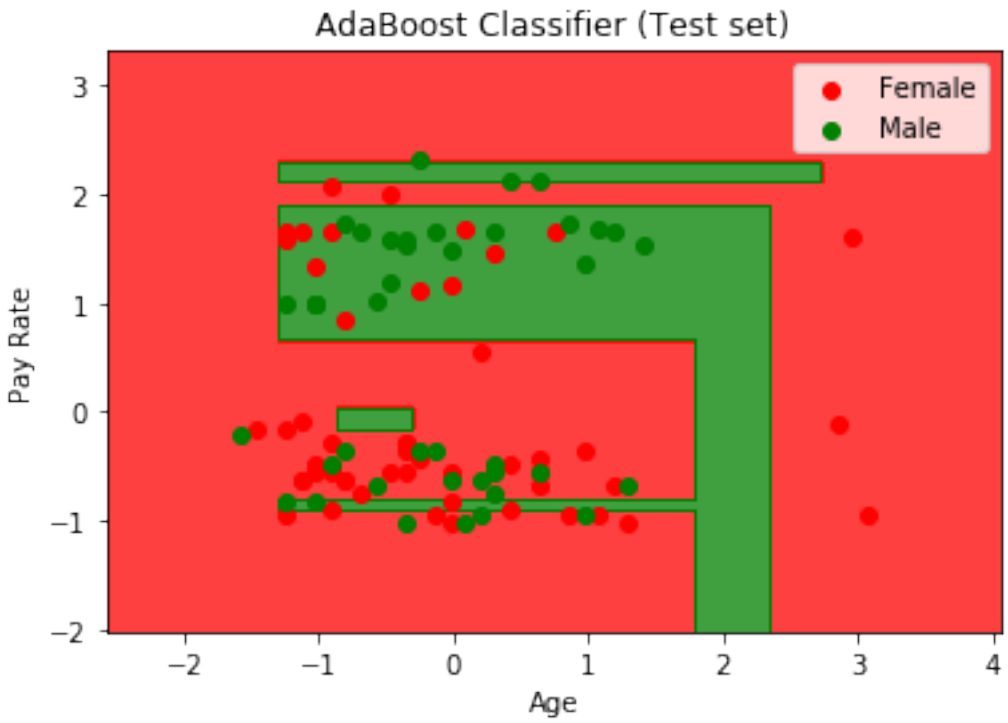

```
the number of trees:
17


Accuracy of classifier:
64.51612903225806


Confusion matrix of the classification:
[[39 13]
 [20 21]]
```



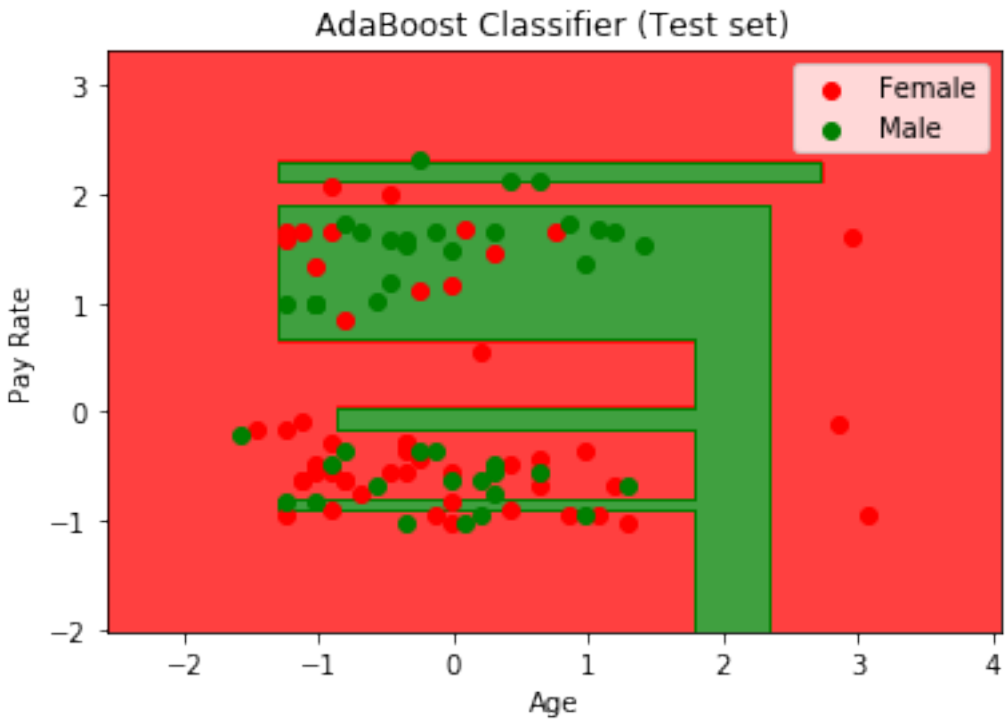

```
the number of trees:
18


Accuracy of classifier:
62.365591397849464


Confusion matrix of the classification:
[[38 14]
 [21 20]]
```



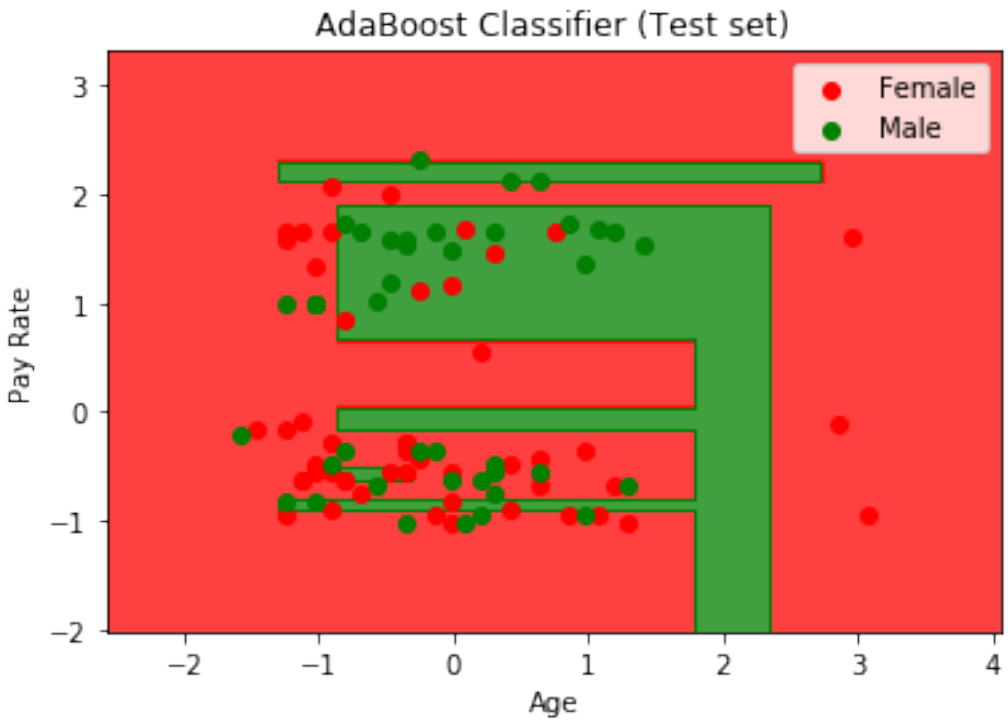

```
the number of trees:
19


Accuracy of classifier:
64.51612903225806


Confusion matrix of the classification:
[[39 13]
 [20 21]]
```



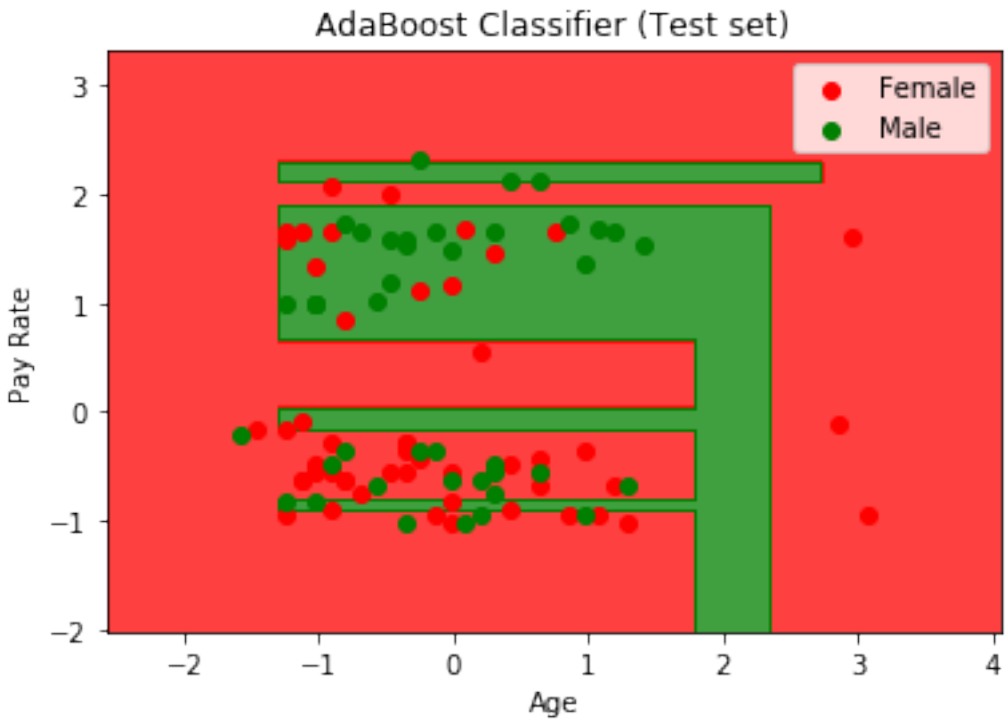

```
the number of trees:
20


Accuracy of classifier:
65.59139784946237


Confusion matrix of the classification:
[[38 14]
 [18 23]]
```



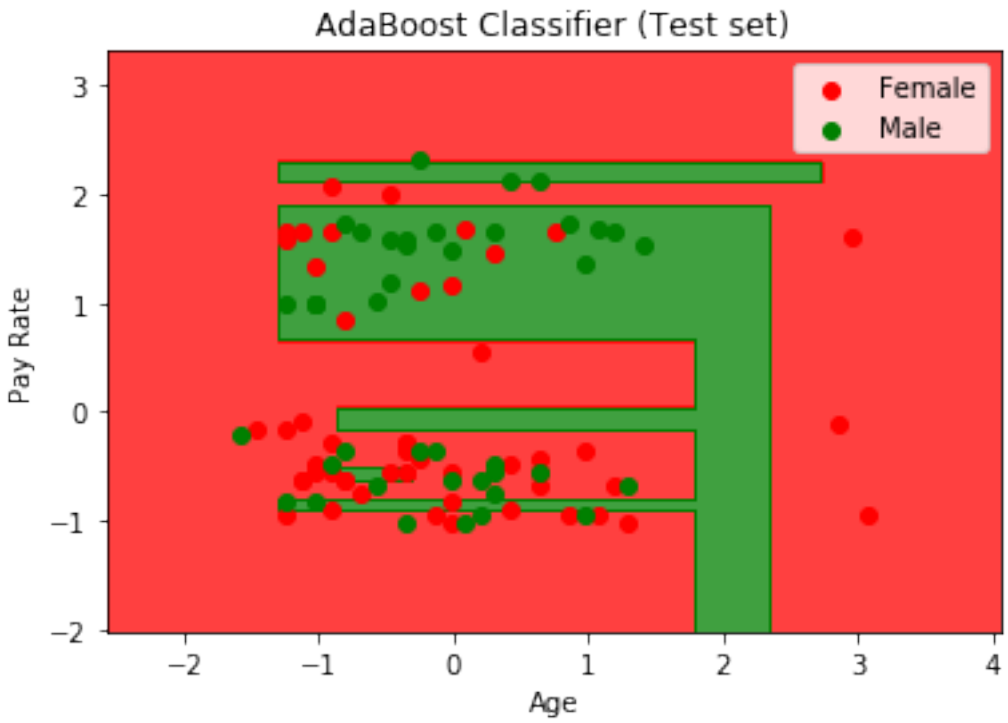

```
the number of trees:
21


Accuracy of classifier:
65.59139784946237


Confusion matrix of the classification:
[[38 14]
 [18 23]]
```



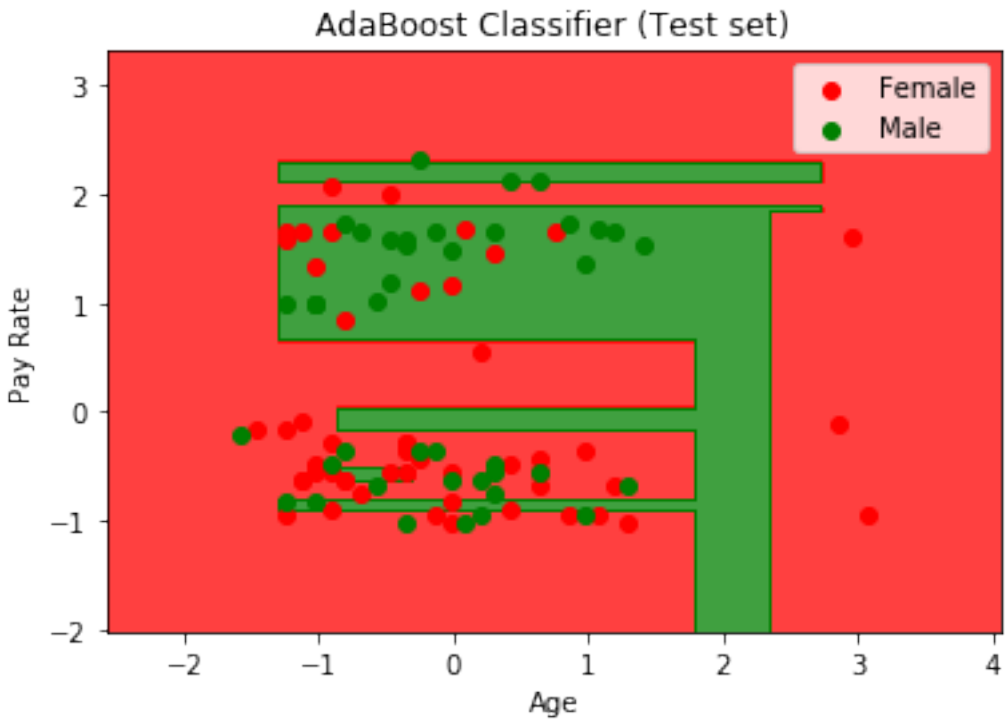

```
the number of trees:
22


Accuracy of classifier:
62.365591397849464


Confusion matrix of the classification:
[[39 13]
 [22 19]]
```



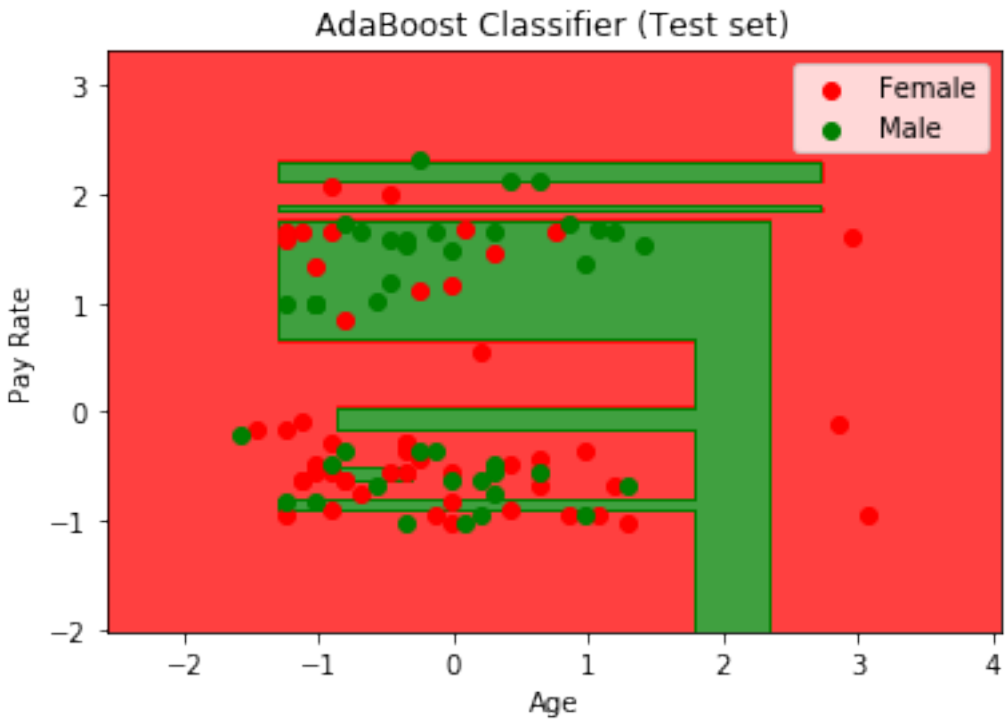

```
the number of trees:
23


Accuracy of classifier:
63.44086021505376


Confusion matrix of the classification:
[[36 16]
 [18 23]]
```



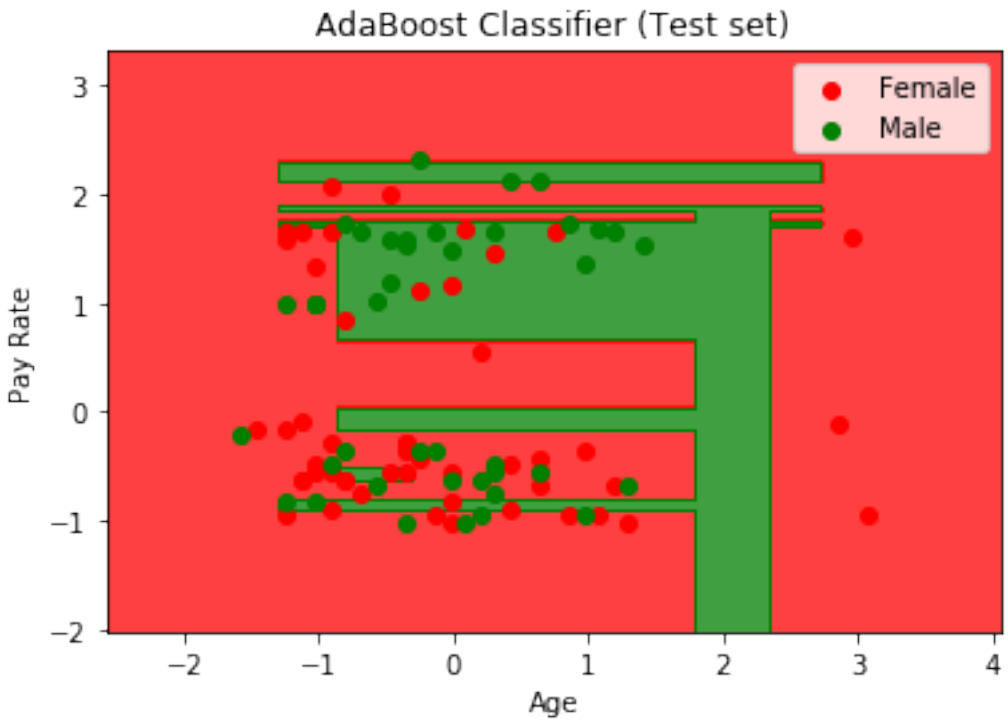

```
the number of trees:
24


Accuracy of classifier:
61.29032258064516


Confusion matrix of the classification:
[[34 18]
 [18 23]]
```



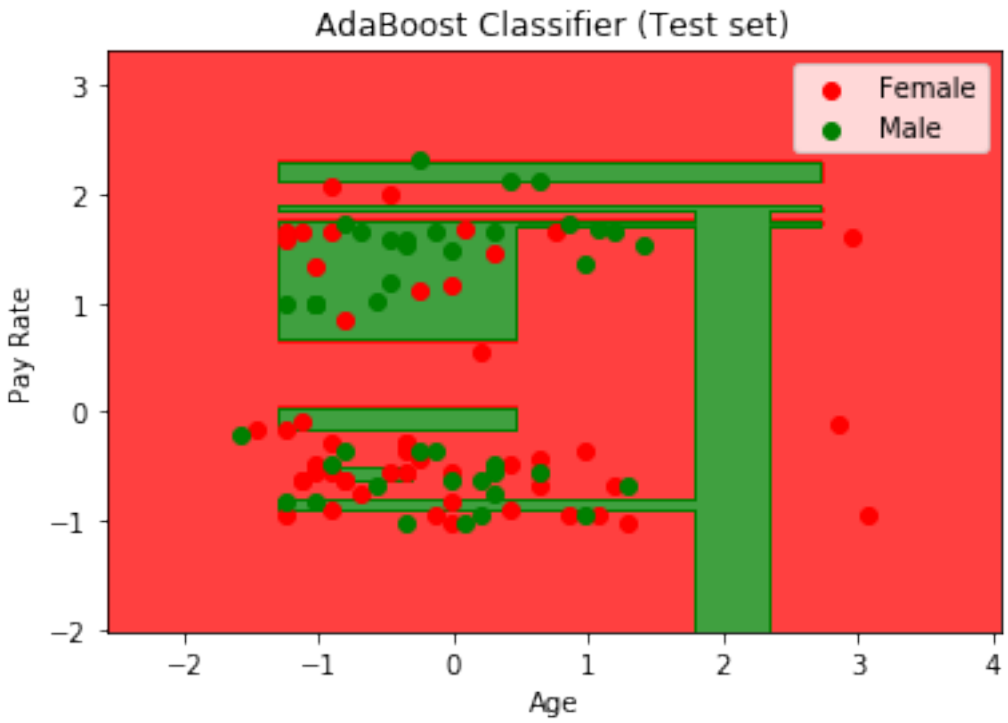

```
the number of trees:
25


Accuracy of classifier:
61.29032258064516


Confusion matrix of the classification:
[[34 18]
 [18 23]]
```



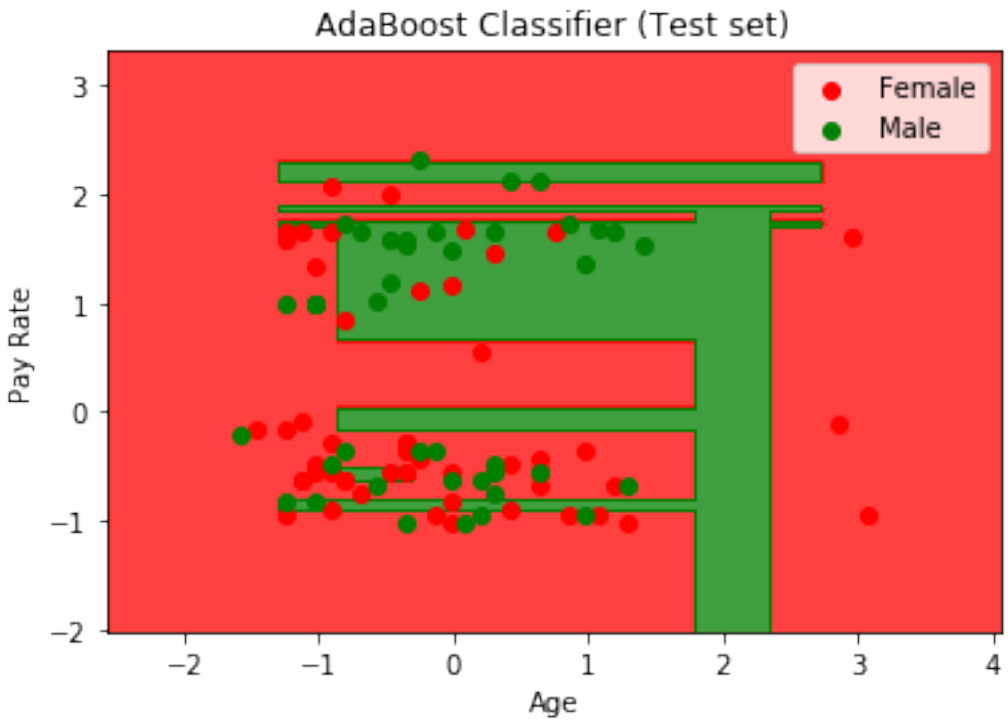

```
the number of trees:
26


Accuracy of classifier:
61.29032258064516


Confusion matrix of the classification:
[[34 18]
 [18 23]]
```




```
the number of trees:
27


Accuracy of classifier:
62.365591397849464


Confusion matrix of the classification:
[[39 13]
 [22 19]]
```




```
the number of trees:
28


Accuracy of classifier:
55.91397849462365


Confusion matrix of the classification:
[[33 19]
 [22 19]]
```




```
the number of trees:
29


Accuracy of classifier:
62.365591397849464


Confusion matrix of the classification:
[[39 13]
 [22 19]]
```

0.1 Discussion:

From examining the classification results above,it can be said that AdaBoost classifier gives the highest accuracy rate 67.74%,when the number of trees in the forest are five when the learning rate is 0.9, test size is 0.3(30%) and random_state is 5.When the number of trees is 1 and 29,AdaBoost classifier for this dataset gives the lowest accuracy rate as 55.91%.There is no direct relationship between the number of trees and the accuracy rate.In other words,when the number of trees in the forest are increased or decreased,it cannot be said that the accuracy changes accordingly.

Changes in the classifier accuracy can also be seen in the confusion matrices given classifier plots.For instance when the accuracy rate is highest,true positive is 45,false positive 7,false nega-

tive 23,and false negative 18.Similarly when the lowest accuracy rate is 55.91% (when the number of trees is 1),the number of true positives is 52,but the number of false negatives is 41.

Further investigation can be done on Adaboost classifier by changing the learning rate as well as test size in order to find out how the accuracy of the classifier changes by changing these parameters.

1 References:

[1] Dataset used from Kaggle.com : https://www.kaggle.com/rhuebner/human-resources-dataset [2] AdaBoost on Wikipedia: https://en.wikipedia.org/wiki/AdaBoost [3]AdaBoost classifier on SciKit: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html#sklea