INTRODUCTION

AdaBoosting is an approach in machine learning based on the idea of creating a highly accurate prediction rule by combining many relatively weak and inaccurate rules. This algorithm is based on Ensemble learning which is the process by which multiple classifiers are strategically generated and combined to solve a particular computational problem. It combines a series of low performing classifiers with the aim of creating an improved classifier. Ensembles offer more accuracy than individual or base classifier.

Ada-boost or Adaptive Boosting is one of ensemble boosting classifier proposed by Yoav Freund and Robert Schapire in 1996. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observation

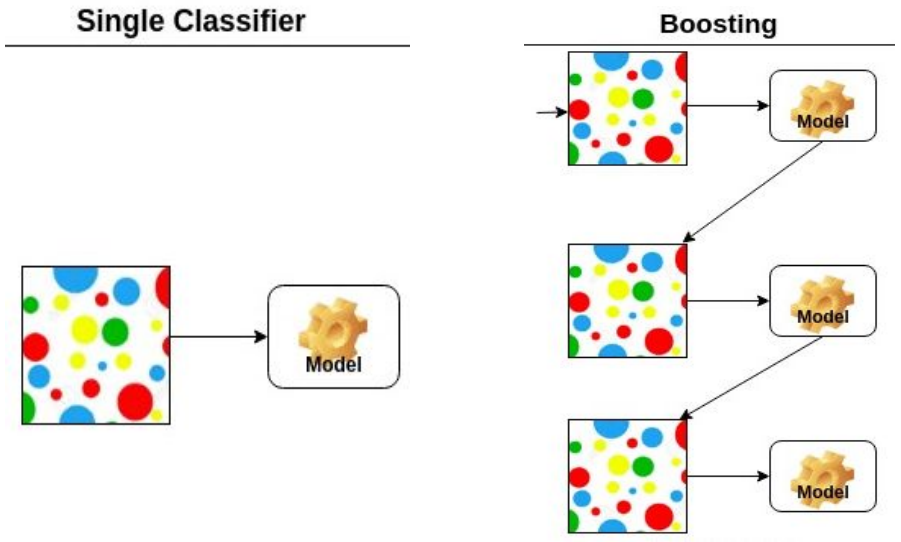


Fig - Ensemble Learning (Boosting)

ADVANTAGES :

- Keep Interpretation of model as it is i.e feature scaling is not required
- High Accuracy (As it tries to fit every data point)
- Fast execution
- Less no. of parameters to tune (Iterations, size of tree)

ALGO STEPS :

1. Initialize the observation weights $w_i = 1/N,\ i = 1, 2, \ldots, N$.
2. For $m = 1$ to M:
   (a) Fit a classifier $G_m(x)$ to the training data using weights w_i.
   (b) Compute
   $$\text{err}_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i}.$$
   (c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],\ i = 1, 2, \ldots, N$.
3. Output $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

Fig - Methodology

1. Each sample is weighted as 1/n. (where n is no of samples)

2. Then we make stumps for each dependent variable. (stump : A Decision tree of only root node and two leafs)

3. Calculate gini index for each stump and the stump with lowest gini index is chosen first. (**Gini Index** is calculated by subtracting the sum of the squared probabilities of each class from one.)

4. Then we calculate error for stump by adding weights associated with incorrectly classified samples. ($0 \leq$ error ≤ 1)

5. Amount of say or alpha is calculated by :
   $$\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$$

6. Using this formula, Amount of say is positive when error is less than 0.5 and negative when error is greater than 0.5 .

7. Now, incorrectly classified samples are emphasized for next stump by increasing their sample weights and decreasing for correctly classified samples using the formula
   $$w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],\ i = 1, 2, \ldots, N.$$

8. Now normalize the weights such that their sum is one.

9. Now make another stump using sample weights i.e more emphasis is given to incorrectly classified samples.

10. This step is repeated M times, where M is the no. of iterations.

11. Now the prediction is given by using the formula :

$$\text{Output } G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$$

DATASET :

The dataset is Social_networks_ads taken from kaggle.com which contains the following columns :

1. User_ID : Unique to every user
2. Gender : male/female
3. Age : depicts age of the user
4. Estimated Salary :The annual income of user
5. Purchased : 0/1 (0 for not purchased / 1 for purchased)

This is a categorical dataset to determine whether a user purchased a particular product. The size is 1000 x 5. It Describes whether the user has clicked the advertisement or not.

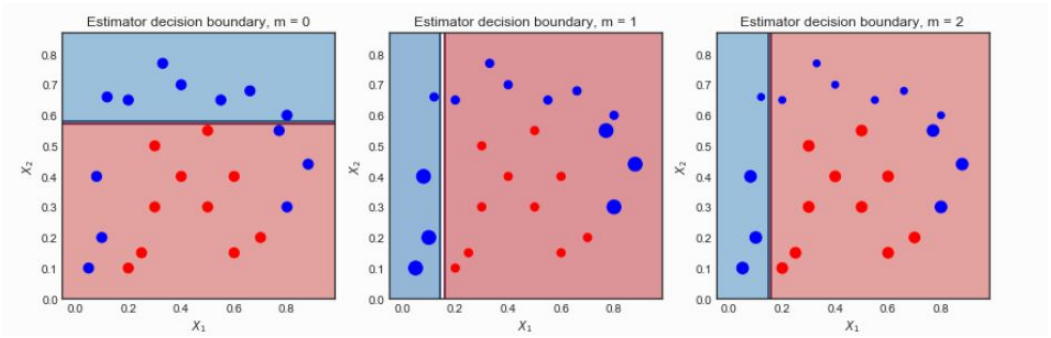
EXAMPLE ON TOY SET :

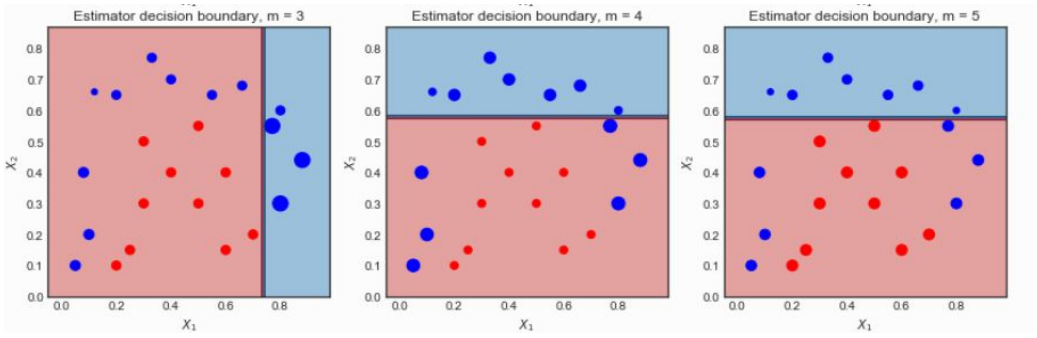
M = 10 (No. Of iterations = 10)
Size of scatter points is proportional to scale of sample_weights

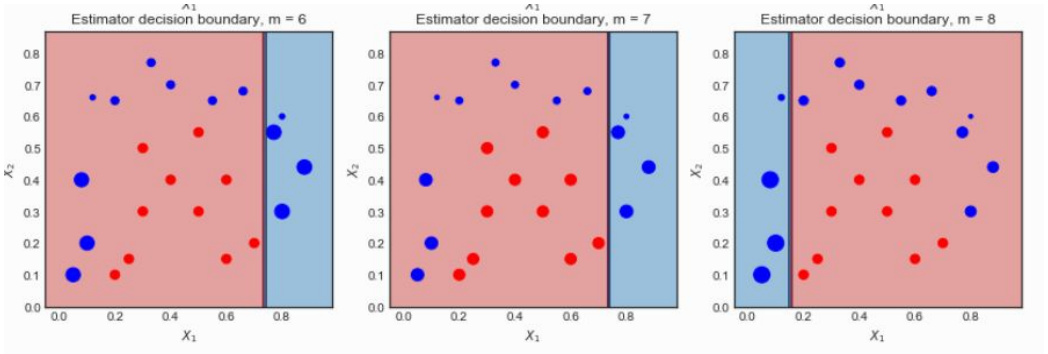
X1 = [.1,.2,.4,.8, .8, .05,.08,.12,.33,.55,.66,.77,.88,.2,.3,.4,.5,.6,.25,.3,.5,.7,.6]

X2 = [.2,.65,.7,.6, .3,.1,.4,.66,.77,.65,.68,.55,.44,.1,.3,.4,.3,.15,.15,.5,.55,.2,.4]

Y = [1,1,1,1,1,1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1]







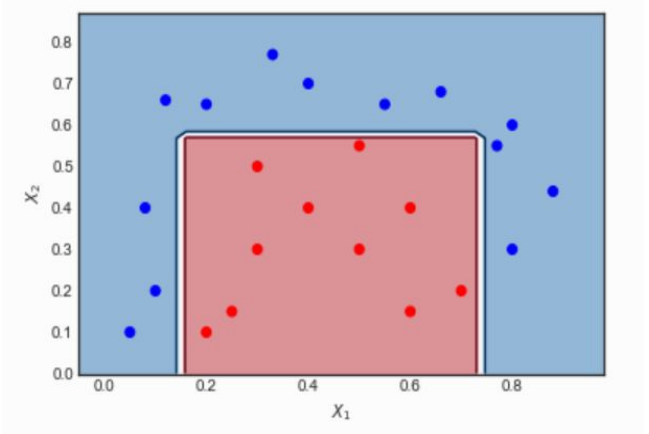
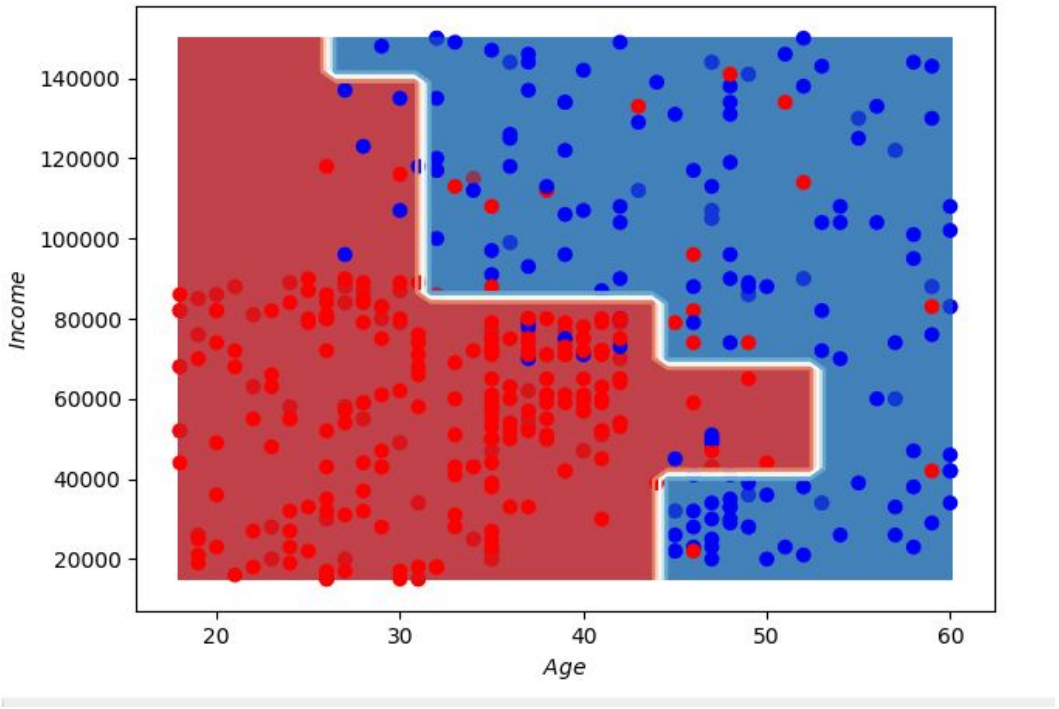


Fig - Final Decision boundary on toy dataset

RESULT :

Using Decision tree as estimator :

Training set :



Decision Boundary - Adaboost Training Set

Confusion matrix :

```
[[183  16]
 [ 16 105]]
```

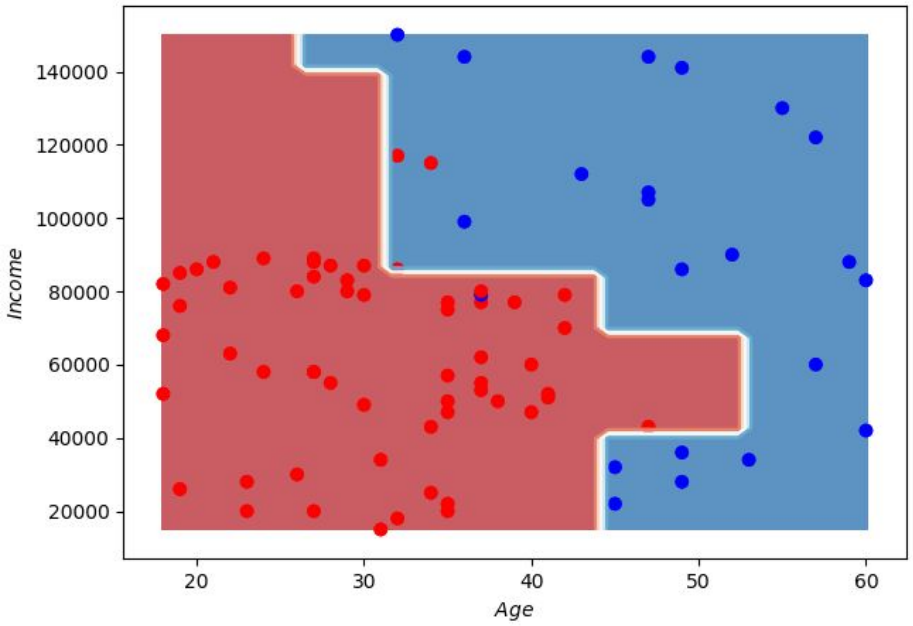
Accuracy : 90 %

Precision : 92 %

Sensitivity : 92 %

Specificity : 86.7 %

Test set :



Decision Boundary - Adaboost Test Set

Confusion matrix :

```
[[56  2]
 [ 1 21]]
```

Accuracy : 96.25 %

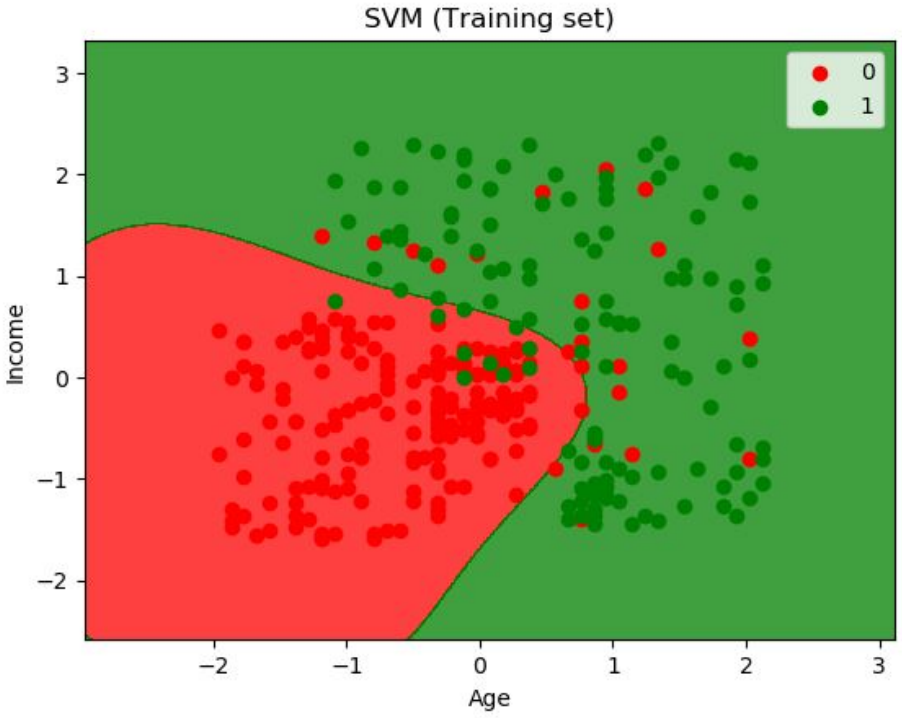
Precision : 96.6 %

Sensitivity : 98. 2 %

Specificity : 91.3 %

Using SVM :

Training Set :



Decision Boundary - SVM Training Set

Confusion matrix :

```
[[178  21]
 [ 10 111]]
```

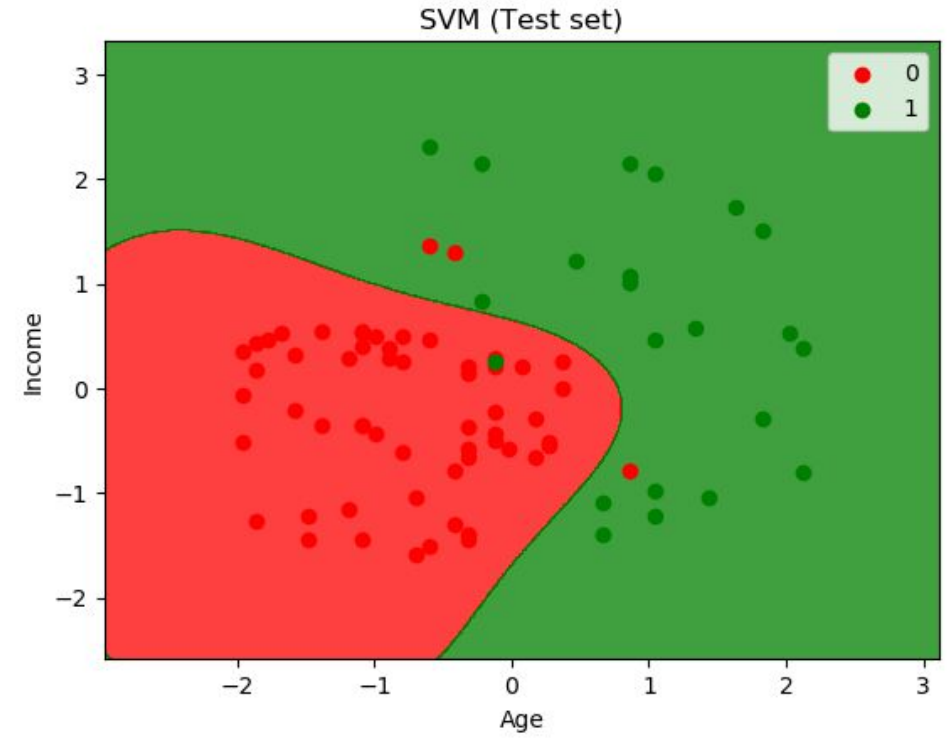
Accuracy : 90 %

Precision : 89.4 %

Sensitivity : 94.6 %

Specificity : 84 %

Test set :



Decision Boundary - SVM Test Set

Confusion matrix :

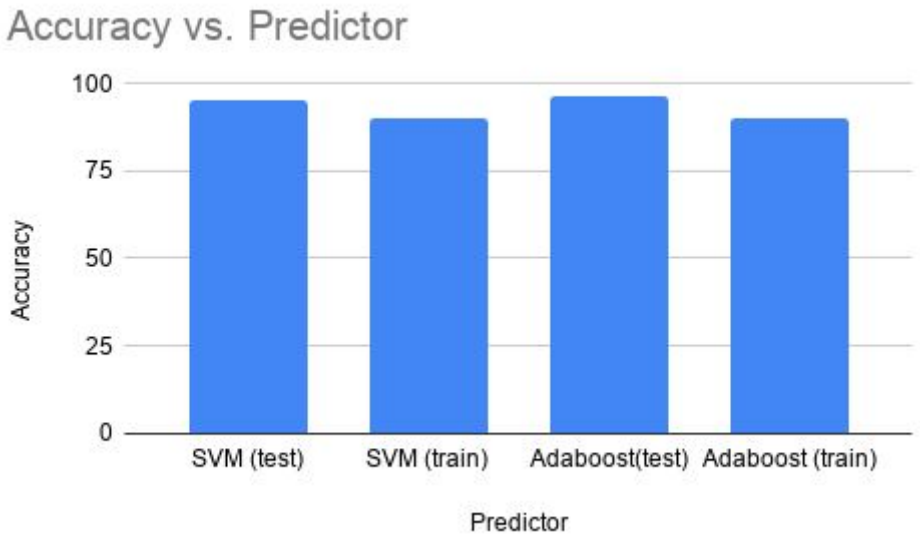
```
[[55  3]
 [ 1 21]]
```

Accuracy : 95 %

Precision : 94.8 %

Sensitivity : 98 %

Specificity : 87.5 %



We conclude that adaboost performs than SVM as seen by confusion matrix because in adaboost we try to fit each and every data point by iterating through the wrongly classified data points and consequently we get a prediction model which predicts the results accurately.

REFERENCES :

1. Intro to boosting, Journal of Japanese Society for Artificial Intelligence,, September, 1999.

2. J. Ross Quinlan : Programs for Machine Learning. Morgan Kaufmann, 1993

3. Robert E .Schapire,Yoav Freund, Peter Bartlett and Wee Sun Lee. Boosting the mar-gin : An explanation for the effectiveness of voting methods. The Annals of Statistics: October1998.

4. Robert E.Schapire. Using output codes to boost multi class learning problems in Machine Learning : Proceedings of the Fourteenth International Conference, 1997.

5. Robert E. Schapire and Yoram Singer. BoosTexter : A boosting - based system for text categorization.

6. Yoram Singer Improved boosting algorithm using condensed - rated predictions. In Proceedings of the Eleventh Annual Conference on Computational Learning Theory.

7. Adaboost Classifier Tutorial, https://www.datacamp.com/community/tutorials/adaboost-classifier-python

8. AdaBoost: Implementation and intuition, https://xavierbourretsicotte.github.io/AdaBoost.html

9. Boosting and AdaBoost for Machine Learning, https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/