Question 1.. Support Vector Machines (20 points) Given 10 points in Table 1, along with their classes and their Lagranian multipliers (αi), answer the following questions:

(a) What is the equation of the SVM hyperplane h(x)? Draw the hyperplane with the 10 points.

Ans:

$$w = \sum_{i>0} \alpha_i \, y_i x_i$$

=0.414*(4, 9) - 0.018(2.5, 1) + 0.018(3.5, 4) - 0.414(2, 2.1)

= > w = [0.846 0.3852]

Now calculating bias,

Intercept = 1-x.T * w

Solving this we get intercept = -3.501

Thus the equation for the HyperPlane would be as follows:

[0.846 0.3852] X -3.501 = 0

(b) What is the distance of x6 from the hyperplane? Is it within the margin of the classifier?

We can calculate this using the following formula:

$$\frac{yh(x)}{||w||}$$

Substituting the value of x_6 = (1.9 , 1.9) and its label y = - 1 in the above equation we get :

Distance = 1.1618, It will be not be within the classifier as it is >1.

(c) Classify the point z = (3; 3)T using h(x) from above:

Plugging the values of (3,3) in the above equation we see that the value if >0. Thus we classify the point as +1. This

2. Support Vector Machines (20 points) Create a binary (2 feature) dataset where the target (2-class) variable encodes the XOR function. Design and implement a SVM (with a suitable kernel) to learn a classifier for this dataset. For full credit, explain the kernel you selected, and the support vectors picked by the algorithm. Redo all the above with multiple settings involving more than 2 features. Ensure that your kernel is able to model XOR in all these dimensions. Now begin deleting the non-support vectors from your dataset and relearn the classifier. What do you observe? Does the margin increase or decrease? What will happen to the margin if the support vectors are removed from the dataset? Will the margin increase or decrease?(You can use packages/tools for implementing SVM classifiers.):

In the following question we are supposed to see which kernel is best for a random binary dataset by fitting different svm kernels on them.

We follow the procedure of creating a random dataset using numpy's random.randn function an then we use the logical Xor to encode it into XOR.

Here we use 3 test on three types of kernels:

1. Gaussian
2. Sigomoid
3. Polynomial with degree= 3

The data split for all the runs is 80-20. From the results we can see that the gaussian kernel performs the best on the test dataset.

Next we remove the Support vectors from the dataset and retrain it. This pushes the boundary backward in the case of the gaussian kernel.

Now we just use the support vectors on the dataset , this does change the boundary, looking at the plot it looks as if the boundary is flipped but the points are classifiied the same. But just using support vectors is a bad idea as it will cause overfitting.

3. Decision Trees (20 points) Please use the data set in Table 2 to answer the following questions:

(a) Show which attribute will be chosen at the root of the decision tree using information gain. Show all split points for all attributes. Please write down every step including the calculation of information gain of the attributes at each split.

We use the concept of entropy and information gain to decide the best feature to spilt on:

1. First we calculate the total entropy of the decision boundary, this is done by:

$$\frac{-4}{9}log_2(\frac{4}{9}) - \frac{5}{9}log_2(\frac{5}{9})$$

$$0.5199 + 0.4711 = 0.991$$

Now we calculate the Entropy of each class attribute, doing it for a_1:

$$[\frac{4}{9}(-\frac{3}{4}log_2(\frac{3}{4}) - \frac{1}{4}log_2(\frac{1}{4})) + \frac{5}{9}(-\frac{1}{5}log_2(\frac{1}{4}) - \frac{4}{5}log_2(\frac{4}{5}))]$$

We get the entropy as

$$0.76$$

Now we calcuate the Entropy for a_2:

$$-[\frac{5}{9}(\frac{2}{5}log_2\frac{2}{5}+\frac{3}{5}log_2\frac{3}{5})+\frac{4}{9}(\frac{1}{2}log_2\frac{1}{2}+\frac{1}{2}log_2\frac{1}{2})]$$

We get the entropy as :

$$0.98$$

Similarly:
We get the entropy of a_3:

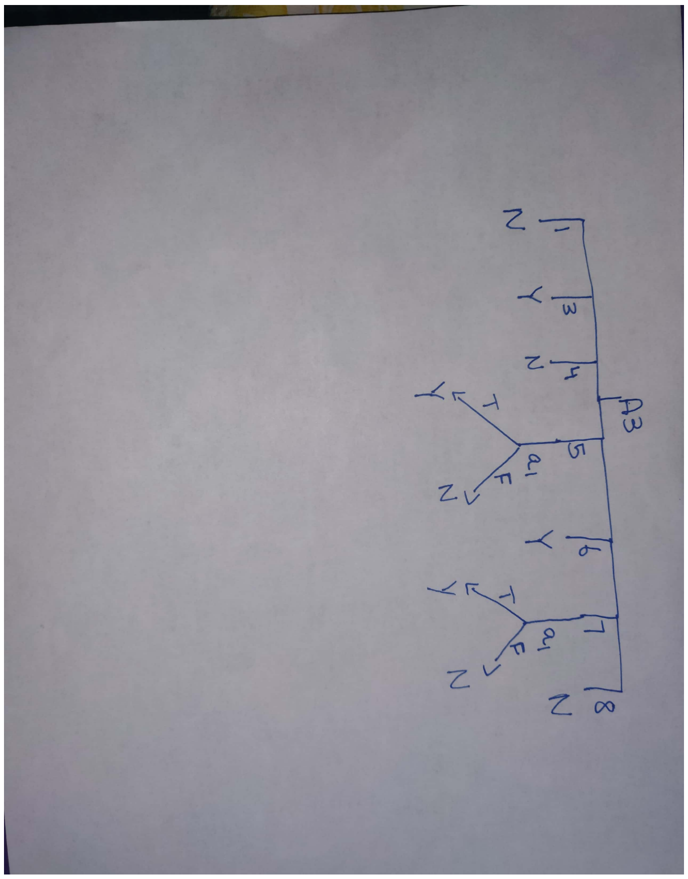
$$0.44$$

We now calculate the information gain for the 3 attributes:
I(a1) = 0.991 - 0.76 = 0.22
I(a2) = 0.991 - 0.98 = 0.007
I(a3) = 0.991 - 0.44 = 0.54

We thus split the the data at the root node on a3 as it has highest information gain.

(b) What happens if we use Instance as another attribute? Do you think this attribute should be used for a decision in the tree?

No this attribute should not be used because it has too many different values and will not yield any good information and will only complicate the process.

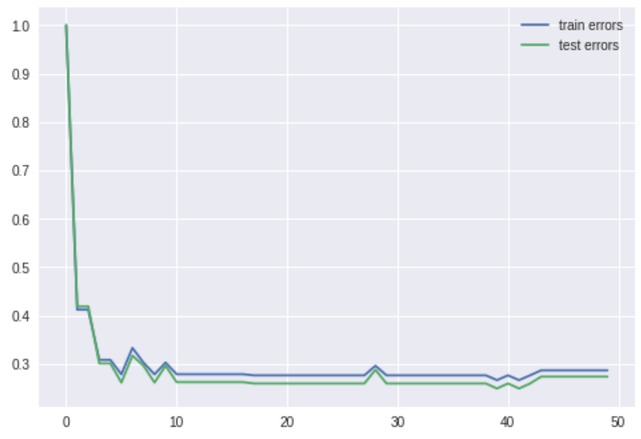


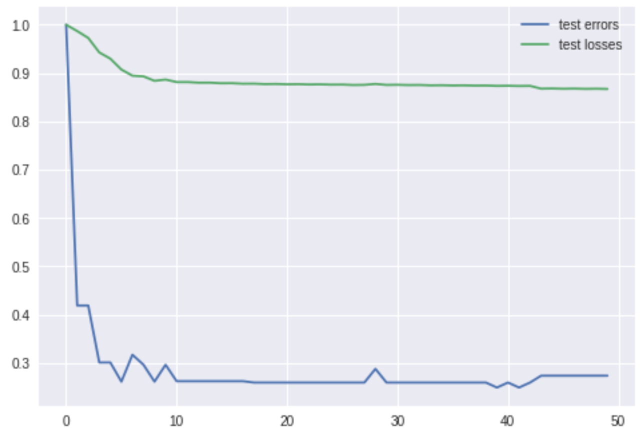
4. Boosting (20 points) Implement AdaBoost for the Banana dataset with decision trees of depth 3 as the weak classifiers (also known as \base classifiers"). You can use package/tools to implement your decision tree classifiers. The fit function of DecisionTreeClassifier in sklearn has a parameter: sample weight, which you can use to weigh training examples differently during various rounds of AdaBoost. Plot the train and test errors as a function of the number of rounds from 1 through 10. Give a brief description of your observations.

Here we implement the adaboost algorithm from scratch with help from sklearn decision tree.

The train test split is: Training : 80%, Testing:20%

Below is the Train error and test error plots along wiht test loss plots.





The training and test errors are:
train error: 0.08893606982778957
test error: 0.10660377358490569

5. Neural Networks (20 points) Develop a Neural Network (NN) model to predict class labels for the Iris data set. Report your training and testing accuracy from 5-fold cross validation. You can use packages such as TensorFlow.

Here we use the KERAS library which is built on top of tensorflow for faster implementation of Deep Neural Networks. Here we use Neural network of size 4-8-3.

This is interpreted as one hidden layers neural network with 4 input neurons, 8 hidden neurons and 3 output neurons.

We split the dataset into 70% training and 30 % testing. The training is done with 5 fold cross validation and the training accuracy for each of it is reported.

The testing accuracy observed is 97.03%