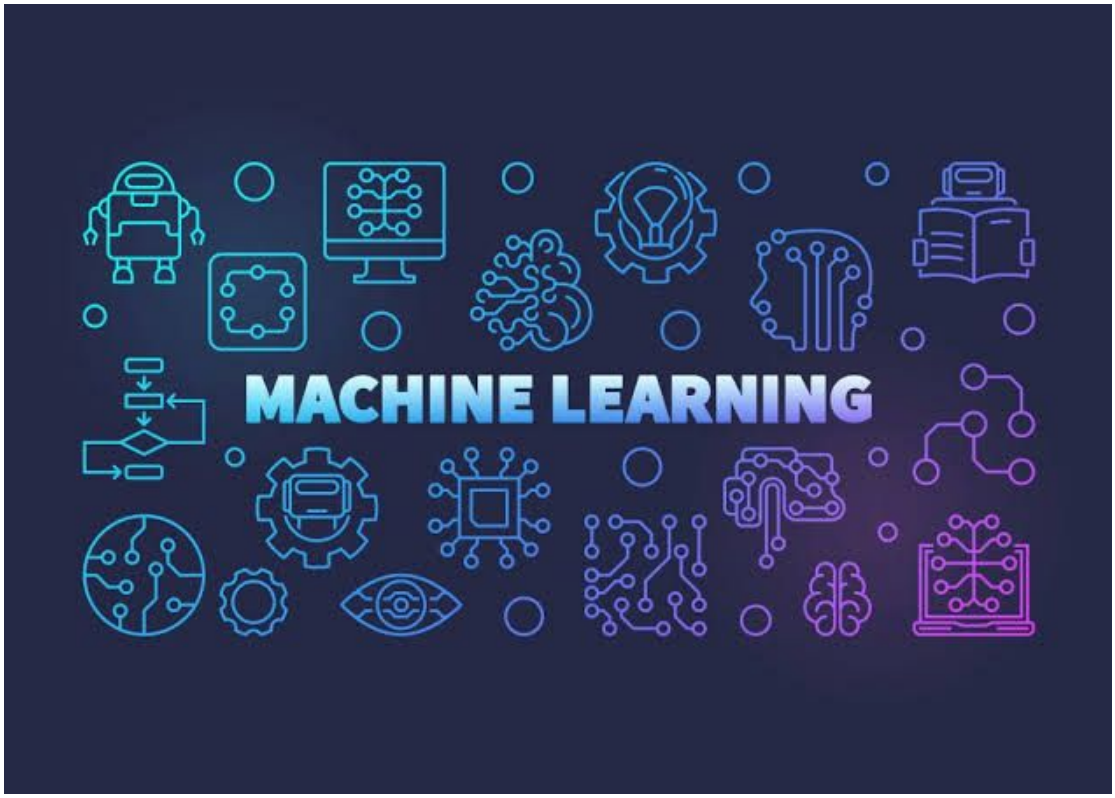
UE17CS303

Machine Learning Assignment

Mr. Channa Bankapur



Team Members

MADHAV MAHESH KASHYAP	PES1201700227
NISHANTH SHASTRY	PES1201700115
SHASHANK PRASAD	PES1201700667

Problem Statement

To classify stellar objects into spectrometric classes based on photometric data.

Research into Domain of Astronomical Classification

The dataset is a MultiClass Classification dataset.

It contains 23 Independent attributes each corresponding to some astronomical data observed.

The Dependent attribute is a 3 Class attribute which is an integer in the range of [0, 2].

Each of these classes corresponds to the Temperature classification of the star.

Light is made up of many different wavelengths. When we pass the light through some gas(such as in the outer layer of the Star), some of these wavelengths get absorbed.

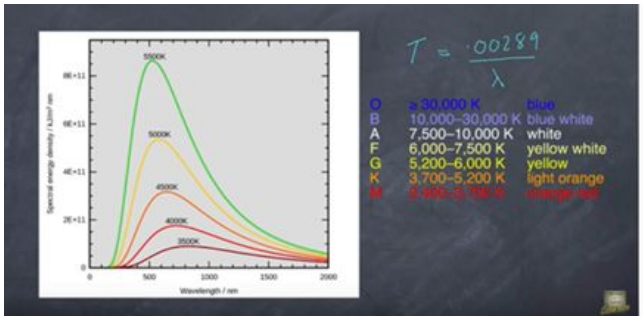
If we can find out which wavelengths are missing, we can pinpoint which elements the star is composed of.

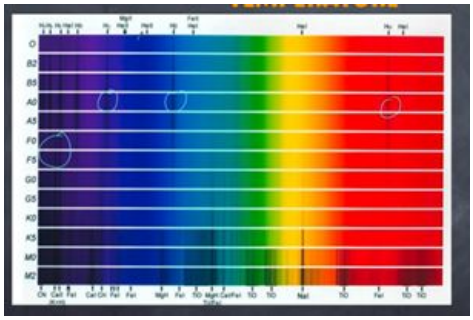
If we look at wavelength having highest intensity, we can determine the temperature of the star.

This method is called Light Spectroscopy.

These measurements of light wavelengths and magnitudes are the columns of this dataset.

After further research, the SLOAN Digital Sky Survey was very helpful in understanding the intricacies of the columns.





PreProcessing

The preprocessing steps include Z-score normalization of all of the 23 features of the dataset.

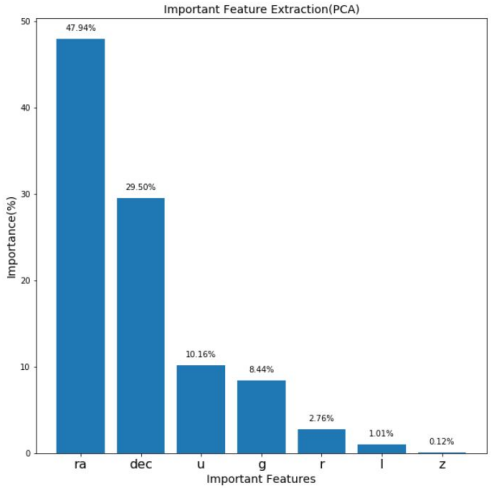
The normalization does not increase the accuracy, but improves the speed of calculation considerably. The speedup observed was almost 200%.

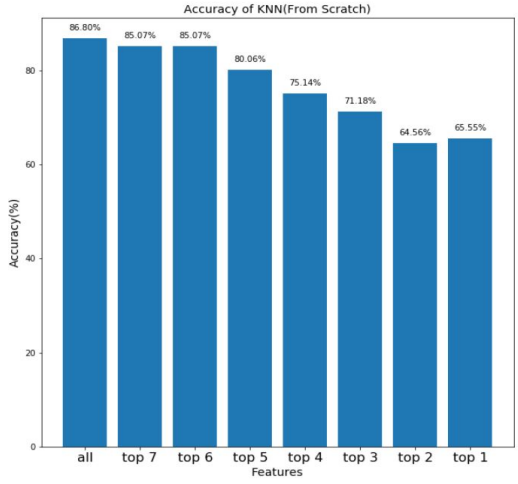
K-Nearest Neighbours (KNN)

The KNN algorithm is a supervised learning algorithm used for solving classification and regression problems. The KNN algorithm is an example of instance based learning and is a Non-parametric learning algorithm.

Pseudo code for KNN:

1) Load data
2) Initialize value of k
3) For getting predicted class, iterate from 1 to the total number of training data points
    - Calculate the distance between test data and each row of training data. We are using Euclidean distance for our algorithm.
    - Sort the calculated distances in ascending order.
    - Get the top k rows from the sorted array
    - Get the most frequent class of these rows
    - Return the predicted class





KNN with K = 11 has a maximum accuracy of 86.80%.

Using PCA for feature extraction, we have identified the top 10 features of our dataset, and implemented **knn using Top 6 features gets an accuracy of 85.07%.**

Thus, we see that using only top 6 features provides a satisfactory accuracy with a much faster computation speed.

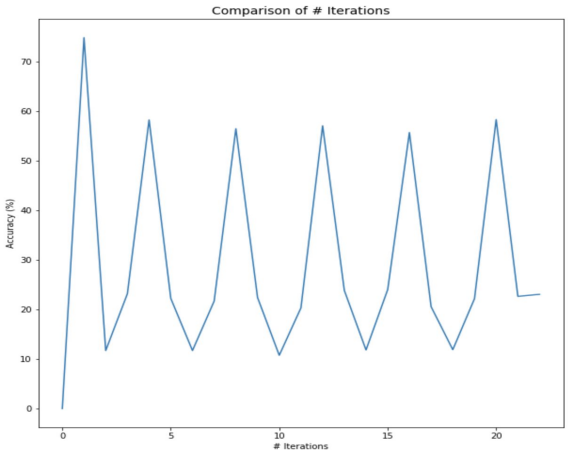
Neural Networks

An Artificial Neural Network is a supervised learning algorithm inspired by biological neural networks. We have implemented a 3 layer Neural Network containing a visible input layer with a sigmoid activation function, a hidden layer with a soft max activation function and a visible output layer. The architecture is of a simple Feed Forward(FF) type.

Pseudo code for ANN:

- Assign random weights to the links of all the nodes.
- Using the inputs and input to hidden layer link we find the activation rate of the hidden nodes.
- Using the activation rate of the hidden nodes and the linkages from hidden layer to the output layer, we find the activation rates of the output nodes.
- Find error rate at the output node and recalibrate all the linkages between the hidden nodes and the output nodes.
- Using the weights and error found at Output node, cascade down the error to hidden nodes.
- Recalibrate the weights between the hidden nodes and input nodes.
- Repeat process until convergence is achieved.
- This represents the final weights and activation rates for the output nodes, and is our final model.

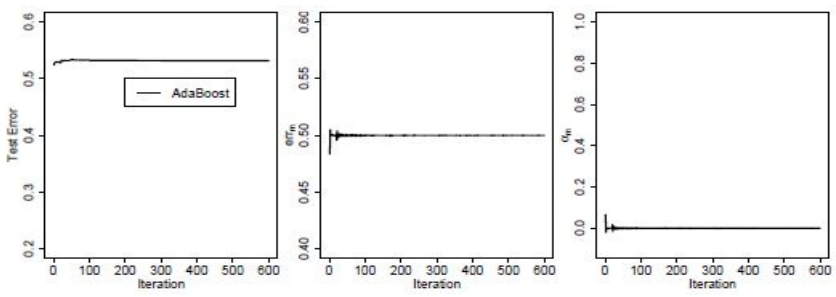
We successfully implemented ANN but it couldn't provide a satisfactory accuracy. We observed a **maximum accuracy of 59.03%**. The ideal graph is a smooth curve, but we were obtaining a wave like pattern. We couldn't figure out why but we think it has to do with either the unsuitability of SoftMax function for this dataset or the requirement of more hidden layers.



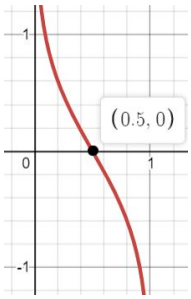
MultiClass Adaboost (SAMME algorithm)

Adaboost is one of the best ensemble methods for binary classification. But, the main limitation of traditional Adaboost is that it is applicable only for Binary classification problems. We usually work by converting the Multiclass problem into multiple Binary Classification problems. But, this method has a flaw.

Notice that in order for the misclassified training data to be boosted, it is required that the error of each weak classifier err(m) be less than ½, otherwise α(m) will be negative and the weights of the training samples will be updated in the wrong direction. For two-class classification problems, this requirement is about the same as random guessing, but when K >2, accuracy 1/2 is harder to achieve since the random guessing error rate=1 - 1/K. We can see that this causes the err(m) of each classifier to stagnate at 0.5; and this causes each of the α(m) = 0. This means that none of the classifiers have any weightage !



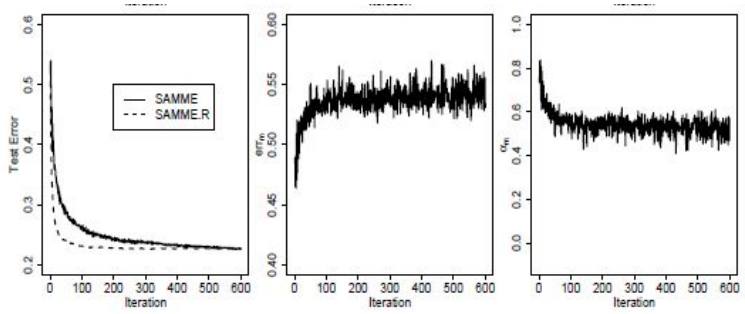
α(m) v/s iterations

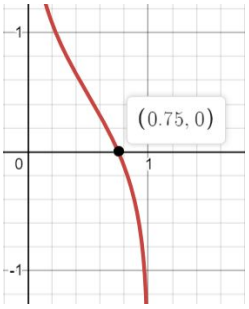


α(m) v/s err(m)

SAMME solves this problem by changing the α(m) updation formula to include an extra log(K -1) term.

1. *Initialize the observation weights* $w_i = 1/n,\ i = 1, 2, \ldots, n.$
2. *For* $m = 1$ *to* M*:*
   (a) *Fit a classifier* $T^{(m)}(\boldsymbol{x})$ *to the training data using weights* w_i.
   (b) *Compute*
   $$err^{(m)} = \sum_{i=1}^{n} w_i \mathbb{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right) / \sum_{i=1}^{n} w_i.$$
   (c) *Compute*
   $$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}} + \log(K - 1).$$
   (d) *Set*
   $$w_i \leftarrow w_i \cdot \exp\left(\alpha^{(m)} \cdot \mathrm{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right)\right),\ i = 1, \ldots, n.$$
   (e) *Re-normalize* w_i.
3. *Output*
   $$C(\boldsymbol{x}) = \arg\max_k \sum_{m=1}^{M} \alpha^{(m)} \cdot \mathrm{I}(T^{(m)}(\boldsymbol{x}) = k).$$





α(m) v/s iterations

α(m) v/s err(m) for K = 4

We can prove that the extra log (K -1) term is mathematically equal to the forward stagewise additive model using a multiclass exponential function. It also reduces to a normal AdaBoost when K=2.

●Study of Base Learners for MultiClass AdaBoost using Sklearn:

DT 1 level ： Top Accuracy = 90.3%

DT 2 levels ： Top Accuracy = 95.5%

DT 3 levels ： Top Accuracy = 92.3%

SVM RBF ： Top Accuracy = 86.7%

In general, we see that the more simplistic base models like Decision Stump outperform complex accurate models like RBF kernel SVM.

Using Base Learner = 2 level Decision Tree, Learning rate = 0.05 and number estimators = 100, Accuracy = 90.25%.

Precision Class 0 = 70.11% , Precision Class 0 = 97.18% , Precision Class 0 = 78.61%

Overall Precision = 81.96%

```
converted X_train
_______ ne  100 __lr  0.05
Accuracy
0.9024970273483948
_____________
              precision    recall  f1-score   support

           0     0.7011    0.7176    0.7093       170
           1     0.9718    0.9484    0.9599      1162
           2     0.7861    0.8400    0.8122       350

   micro avg     0.9025    0.9025    0.9025      1682
   macro avg     0.8197    0.8353    0.8271      1682
weighted avg     0.9058    0.9025    0.9038      1682
```

Summary of Results

Experimental Results using SKLearn Package

Method	Best Accuracy(after tuning hyper-parameters)
Decision Tree	0.87
Simple SVM	0.88
Weighted KNN k=13	0.88
SVM Polynomial Kernel Degree 3	0.89
SVM RBF Kernel	0.89
XGBoost learning rate = 0.55 , num_estimators = 1000	0.9143
Random Forests num_estimators = 150 , max_features = 0.5	0.917
Adaboost with Decision Tree level = 2	0.9548

From Scratch Implementation Results

Method	Best Accuracy
KNN	86.80%
ANN	59.03%
Multiclass Adaboost (SAMME)	90.25% Accuracy, 81.96% Precision

Analysis / Conclusion

Ensemble based learners such as AdaBoost provide state of the art results. The SAMME algorithm improves upon Binary Class Adaboost and allows Multiclass Classification.

We implemented this from scratch and got an Accuracy of 90.25% and Precision of 81.96%. Reference to the paper ()