ITCS 6156 Spring 2017
ASSIGNMENT : 3

SUPERVISED LEARNING

BOOSTING

Rahul Rachapalli
800968032
rrachapa@uncc.edu

Classification using Boosting

(Supervised Learning)

Boosting:

Boosting is a machine learning meta algorithm that is used to reduce the variance and bias of supervised learning algorithms. This algorithm is used to convert a weak learner algorithm into a strong learning algorithm. In most boosting algorithms, a weak learner is used to classify data into smaller sets and then a strong learner is applied on the classified data set

On a given data set, a lazy learning algorithm is used to classify and store the given data set. On the classified data set, a strong learning algorithm like decision trees or artificial neural networks to build several models for all groups of data rather than just one as in the strong learning algorithm.

As shown in Fig 1, the actual data is classified into various sub processes and then the best ones with minimal error are combined together to produce even better results and classify the data.

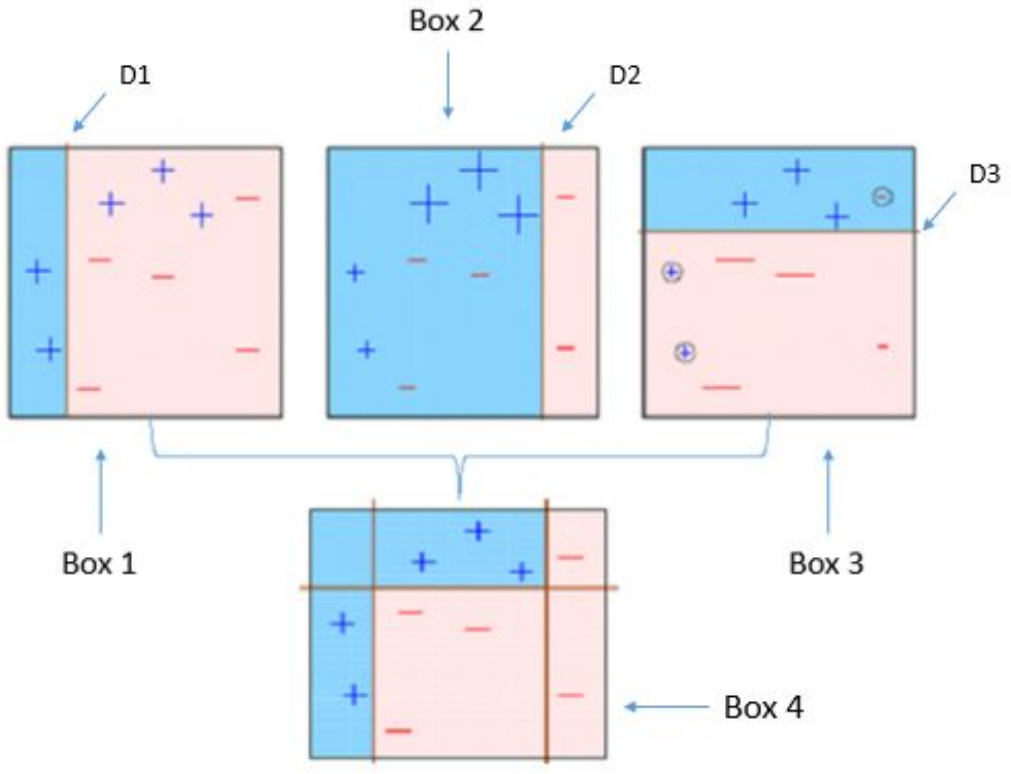

Fig 1: Adaboost Classification on training dataset

Algorithms

Adaboost:

Adaptive **Boost** is the full form of Adaboost. It is a machine learning algorithm that is used to construct a ”strong” classifier as linear combination of “simple” “weak” classifiers. The main idea in Adaboost is to assign initial weights to each data point that needs to be classified then apply a weak learner on this data set to classify data. The next step is to increase the weights of wrongly classified data and perform the weak learner operation on the previously classified data. This process is repeated until the data is separated completely. This algorithm is very sensitive to noise.

Random Forest :

Random forest is based on ensemble method of classifying the data using Decision tree classification algorithm. While building a tree, the algorithm may choose to select a random two features, and this continues until we reach a leaf node with a decision or all the data at the leaf nodes have a common property. The result is many trees that are all formed by random sets of features.

Bagging :

Bagging stands for **B**ootstrap **Agg**regation. It is a way to decrease the variance of your prediction by generating additional data for training from your original dataset using combinations with repetitions to produce multisets of the same cardinality/size as your original data. By increasing the size of your training set one can't improve the model predictive force, but just decrease the variance, narrowly tuning the prediction to expected outcome.

ExtraTrees :

ExtraTrees stands for Extremely Classified Trees. This algorithm builds an ensemble of unpruned decision or regression trees according to the classical top-down procedure. Its two main differences with other tree based ensemble methods are that it splits nodes by choosing cut-points fully at random and that it uses the whole learning sample (rather than a bootstrap replica) to grow the trees. The predictions of the trees are aggregated to yield the final prediction, by majority vote in classification problems and arithmetic average in regression problems.

Gradient Boosting :

Boosting is a two-step approach, where one first uses subsets of the original data to produce a series of averagely performing models and then "boosts" their performance by combining them together using a particular cost function (=majority vote). Unlike bagging, in the classical boosting the subset creation is not random and depends upon the performance of the previous models, every new subsets contains the elements that were misclassified by previous models. Adaboost uses increased weights to emphasize the misclassified data whereas

Gradient boosting uses Gradients to emphasize the misclassified data. Gradient boosting is typically used with decision trees (especially CART trees) of a fixed size as base learners.

Data Sets

A dataset is a collection of data represented in as a single statistical matrix. Given are two data sets

1. *Amazon reviews - Sentiment Analysis*
2. *Optical recognition of handwritten digits*

1. Amazon reviews sentiment analysis dataset

This dataset consists of reviews of Amazon products which is a subset of large Amazon review collection. The data set has 3 columns and the type of data is as follows:

Name of the product	-	*String / text*
Review of the product	-	*String / text*
Rating given by the user	-	*Integer set {1,2,3,4,5}*

2. Optical recognition of handwritten digits

This data set consists of preprocessed normalized bitmaps of handwritten digits from a preprinted form. From a total of 43 people, 30 contributed to the training set and different 13 to the test set. 32x32 bitmaps are divided into non-overlapping blocks of 4x4 and the number of on pixels are counted in each block. This generates an input matrix of 8x8 where each element is an integer in the range 0..16. This reduces dimensionality and gives invariance to small distortions.

The dataset has 65 columns with 64 columns being the 8X8 matrix integer values and the last column being the digit that the 64 values represented.

Amazon Dataset Analysis and Prediction

Assumptions:

The name of the product is ignored as it is independent of the rating and ratings cannot be predicted using the name of the product. Hence, the rating depends on the reviews only.

Preprocessing:

Step 1: The reviews are separated and each review is converted into an array of words.

Step 2: Each word in the array is compared with stop words that are defined. If the word is present in the list of stop words, it is ignored and we proceed to the next word.

Step 3: If the word is not present in the list of stop words, we lemmatize the word and then stem it. At the end of step 3, we get a list of words that are lemmatized and stemmed.

Step 4: The words that are lemmatized and stemmed are converted into a single string and the word frequency is counted.

Step 5: The top 500 words or the whole list of words are considered as the attributes to dive the data.

Step 6: A sparse matrix is built based on the above attributes counting the occurrences of attributes in each review.

Stop Words:

The words 'the','a','an,'is','it' etc., are present in almost all the sentences and do not contribute to the decision process. All such words are called stop words and it is important to remove the stop words in pre-processing stage. So, we used some libraries like nltk to get a list of some pre-defined stop words.

Lemmatizing[2]:

The process of grouping together different forms of word to analyse it as a single word.

Stemming[3]:

The process of converting words into their stem or its root form. For sing, sang, sung, singing are represented in their base word form like sing*

Prediction :

The rating of a review is predicted after the new review is pre-processed (lemmed and stemmed) and a sparse matrix is built for it using the same set of attributes that we decided to build the decision tree. The new sparse matrix is given as input to the neural network and the result is the predicted rating of the given review.

Results:

There are 500 columns and all of them are used as attributes to predict the results. The prediction probability for this dataset is **57.9955%** irrespective of the number of estimators.

Number of Estimators	Classifier	Prediction rate
50	Gradient Boosting	57.9955
	Adaboost	57.9955
	Bagging	57.9955
	Random Forest	57.9955
	Extra Trees	57.9955
100	Gradient Boosting	57.9955
	Adaboost	57.9955
	Bagging	57.9955
	Random Forest	57.9955
	Extra Trees	57.9955
200, 250, 300, 350, 400, 500, 600, 650, 750	Gradient Boosting	57.9955
	Adaboost	57.9955
	Bagging	57.9955
	Random Forest	57.9955
	Extra Trees	57.9955

Table 1 : Dependency of Number of Estimators on different classifiers for Amazon Data set

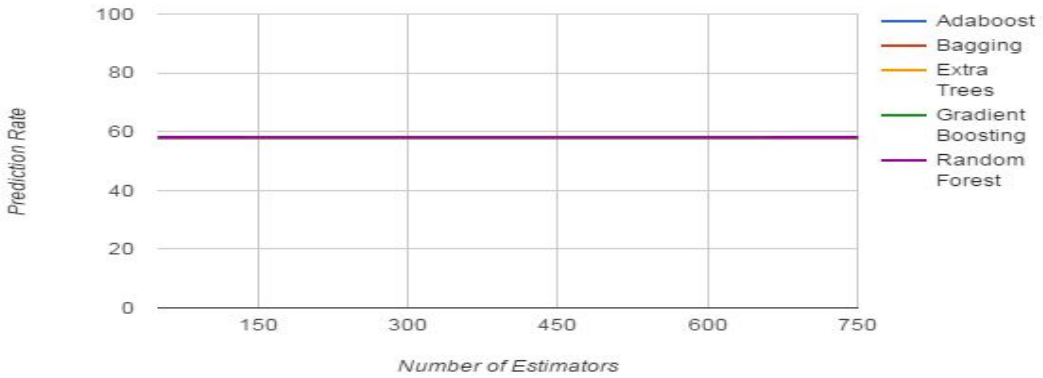

Fig 2 : Dependency of Number of Estimators on different classifiers for Amazon Data set

Digit Recognition Dataset Analysis and Prediction

Assumptions:

There are no missing values in the dataset and the given data is accurate.

Preprocessing:

The data is arranged in the form of a list, read from the file and is represented as a Data Frame.

Neural Network :

A Neural Network is built from the data frame that is obtained from the pre-processing step. This data frame itself acts as a matrix to build the decision tree.

Prediction :

A new dataset can be predicted using the decision tree built. The only step that is needed for it to be predicted is that there are **64** columns in the given integer data set. The pre-processing stage allows us to represent this data in the form of a list. The new list is input to the decision tree for prediction.

Results:

There are 64 columns and all of them are used as attributes to predict the results. The prediction probability for this dataset is **97.9967%** with **8** neighbors, **ball_tree** as the classifier and **50** leaf nodes.

Number of Estimators	Classifier	Prediction rate
50	Gradient Boosting	94.3239
	Adaboost	56.5943
	Bagging	93.1553
	Random Forest	96.9393
	Extra Trees	97.3289
100	Gradient Boosting	95.7151
	Adaboost	56.5943
	Bagging	93.3779
	Random Forest	96.995

	Extra Trees	97.5515
150	Gradient Boosting	95.8264
	Adaboost	56.5943
	Bagging	93.8787
	Random Forest	97.5515
	Extra Trees	97.7741
200	Gradient Boosting	96.1603
	Adaboost	56.5943
	Bagging	93.4335
	Random Forest	96.8837
	Extra Trees	97.7741
250	Gradient Boosting	96.1603
	Adaboost	56.5943
	Bagging	93.6004
	Random Forest	97.3845
	Extra Trees	97.6628
300	Gradient Boosting	96.1603
	Adaboost	56.5943
	Bagging	93.3779
	Random Forest	97.2732
	Extra Trees	97.7184
400	Gradient Boosting	96.1603
	Adaboost	56.5943
	Bagging	93.7117
	Random Forest	97.3289
	Extra Trees	97.6071

500	Gradient Boosting	96.1603
	Adaboost	56.5943
	Bagging	93.4891
	Random Forest	97.4402
	Extra Trees	97.6628
750	Gradient Boosting	96.1603
	Adaboost	56.5943
	Bagging	93.6561
	Random Forest	97.4958
	Extra Trees	97.941

Table 2 : Dependency of Number of Estimators on different classifiers for Digit Recogniser Data set

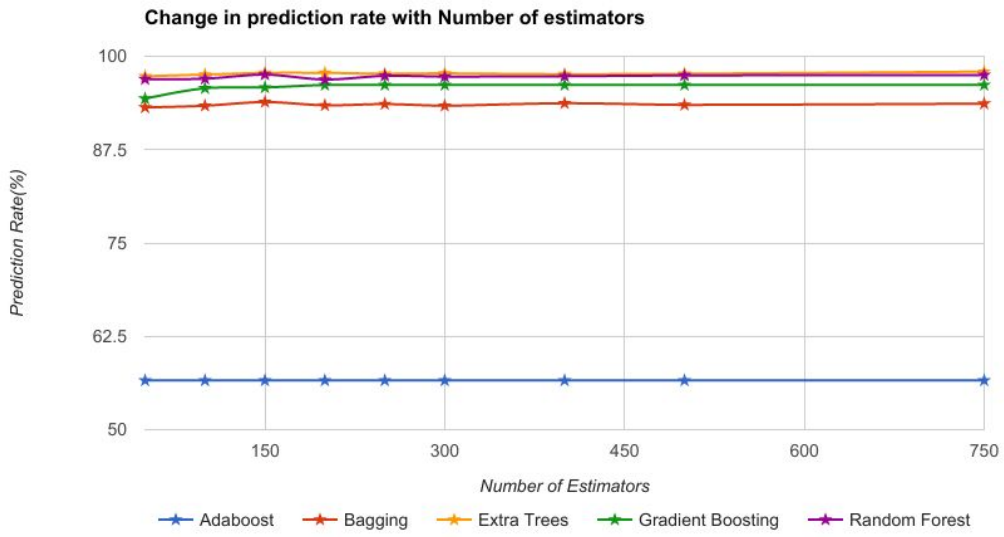

Fig 3 :Dependency of Number of Estimators on different classifiers for Digit Recogniser Data set

From the results, it is observed that the number of estimators affects the prediction rate for optical image classification. After analysis, **200 estimators** gives us the best results for almost all the classifiers. The Adaboost has a very low prediction rate irrespective of the number

of classifiers. **Random Forest** provides the best results for many cases and it has been implemented in the code.

Acknowledgement

This assignment is a joint work with Sujal Vijayaraghavan.

The data exploratory questions have been answered in Assignment 1 and have not been included again.

References

1. ITCS6156_SLProject.pdf
2. https://en.wikipedia.org/wiki/Lemmatisation
3. https://en.wikipedia.org/wiki/Stemming
4. http://documents.software.dell.com/Statistics/Textbook/Text-Mining
5. http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits
6. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
7. http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
8. https://en.wikipedia.org/wiki/AdaBoost
9. http://www.robots.ox.ac.uk/~az/lectures/cv/adaboost_matas.pdf
10. Extremely randomized trees by Pierre Geurts, Damien Ernst and Louis Wehenkel