Information – SL Assignment (Boosting)

OptDigits Dataset:

- Status - Completed.
- It was an easy implementation of Boosting on the OptDigits dataset. I have first given indices to all the attributes and labelled the class attribute as Target so that WEKA can understand it.
- AdaboostM1 in WEKA is a Classifier that Class for boosting a nominal class classifier using the Adaboost M1 method. Only nominal class problems can be tackled. Often dramatically improves performance, but sometimes overfits..
- The number of weak learners = 100 and the base estimators that I've tried on are IBk, J48, LMT, Decision Tree and RandomForest.
- The learning rate by default is 1.
- For J48 algorithm WEKA gives accuracy of around 96% and the confusion matrix for each class portrays that classes 3, 7 and 8 are falling behind the classification.

=== Detailed Accuracy By Class ===

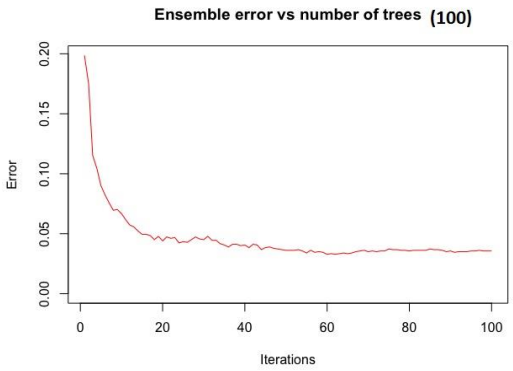
	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.983	0.004	0.967	0.983	0.975	0.972	1.000	0.999	0
	0.978	0.009	0.922	0.978	0.949	0.944	0.998	0.987	1
	0.972	0.002	0.983	0.972	0.977	0.975	1.000	0.997	2
	0.902	0.006	0.948	0.902	0.924	0.916	0.997	0.983	3
	0.934	0.004	0.960	0.934	0.947	0.941	0.996	0.979	4
	0.956	0.004	0.967	0.956	0.961	0.957	0.996	0.987	5
	0.978	0.002	0.983	0.978	0.981	0.978	0.999	0.994	6
	0.911	0.004	0.959	0.911	0.934	0.927	0.990	0.966	7
	0.943	0.006	0.943	0.943	0.943	0.936	0.994	0.976	8
	0.928	0.017	0.861	0.928	0.893	0.881	0.993	0.952	9
Weighted Avg.	0.948	0.006	0.949	0.948	0.948	0.943	0.996	0.982	

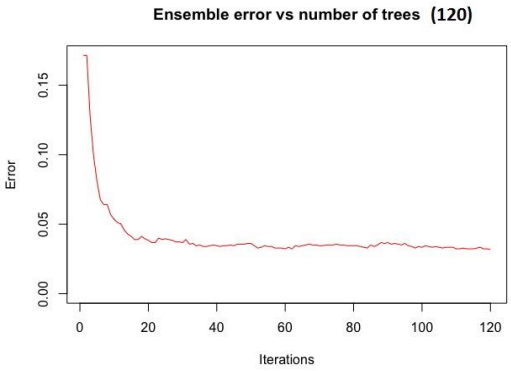
Let's take RandomForest which is an ensemble learner. It also gives accuracy of around 96% and this time we can say that the classes 3,7 and 8 were again classified a little accurately than compared to other classes.

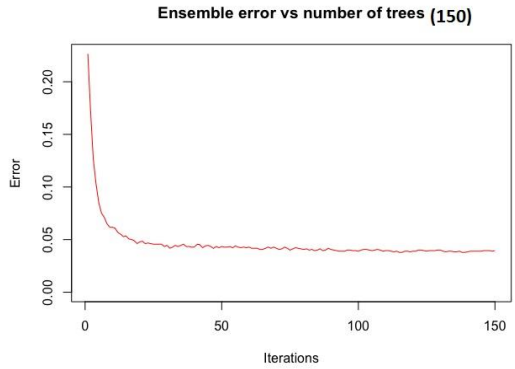
=== Detailed Accuracy By Class ===

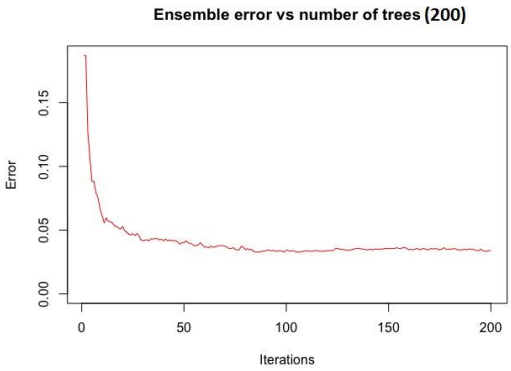
TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.994	0.001	0.994	0.994	0.994	0.994	1.000	1.000	0
0.995	0.005	0.958	0.995	0.976	0.973	1.000	0.999	1
0.983	0.000	1.000	0.983	0.991	0.991	1.000	0.999	2
0.967	0.002	0.978	0.967	0.973	0.969	0.999	0.992	3
1.000	0.002	0.984	1.000	0.992	0.991	1.000	1.000	4
0.978	0.003	0.973	0.978	0.975	0.973	0.999	0.995	5
0.989	0.001	0.994	0.989	0.992	0.991	1.000	0.999	6
0.916	0.001	0.988	0.916	0.951	0.946	0.999	0.992	7
0.954	0.006	0.943	0.954	0.949	0.943	0.997	0.986	8
0.944	0.010	0.914	0.944	0.929	0.921	0.998	0.982	9
Weighted Avg. 0.972	0.003	0.973	0.972	0.972	0.969	0.999	0.994	

- Let's take the number of trees as 150 , we get an accuracy of around 97%.
- We can also see that the mean squared error has been reducing as we keep increasing the trees. Accuracy increases as we increase pruning as well.
- We get the following graphs for different number of trees.









Let's take IBK which is an ensemble learner. It also gives accuracy of around 97% and for k=30 neighbours. It gives the following results.

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	0
0.995	0.008	0.933	0.995	0.963	0.959	0.993	0.928	1
0.989	0.000	1.000	0.989	0.994	0.994	0.994	0.990	2
0.978	0.002	0.978	0.978	0.978	0.976	0.988	0.959	3
0.983	0.002	0.983	0.983	0.983	0.982	0.991	0.969	4
0.984	0.001	0.989	0.984	0.986	0.985	0.991	0.974	5
1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	6
0.983	0.001	0.989	0.983	0.986	0.984	0.991	0.974	7
0.937	0.003	0.970	0.937	0.953	0.948	0.967	0.915	8

	0.944	0.005	0.955	0.944	0.950	0.944	0.970	0.908	9
Weighted Avg.	0.979	0.002	0.980	0.979	0.979	0.977	0.989	0.962	

Amazon Reviews Dataset:

- Status – Completed.
- The challenging part dealing with Amazon Reviews Dataset is to understand how to classify text data.
- Implemented Bag of Words algorithm to the reviews column of the dataset. Cleaned the data from stopwords (using NlTK's corpus – stopwords), numbers and converted to a set of meaningful words.
- Count Vectorizer in scikit learn was used to create a bag of words model which will return binary value if that is word present in each review of the data set or not. The same will be iterative for all the words in the bag of words.
- Later I learned of something called TF-IDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval, text mining, and user modelling.
- Instead of using count vectorizer I used TF-IDF vectorizer in scikit-learn to give a vector for each review with which I performed decision tree classification.
- I used the Adaboost classifier from the scikit-learn's neural network library.
- The parameters for the default sklearn's ADB classifier were –base-estimator= decisiontree, learning rate=1, default estimators=100.
- To reduce the time to execute the AdaBoost algorithm on the entire data set, I've reduced the number of reviews to 8000 in train dataset and to 2000 reviews in test dataset.
- I've run different learning rates with different estimators on decision tree algorithm, please find all the results in Results word document.