IB2 Instance Based Learning Algorithm

July 25, 2017

I declare that all material in this assessment task is my own work except where there is clear acknowledgement or reference to the work of others and I have complied and agreed to the UIS Academic Integrity Policy at the University website URL: http://www.uis.edu/academicintegrity

Student Name: Eric Maxwell UID: 654720206 Date: July 25, 2017

1 Introduction

The IB2 instance based learning algorithm is based on the IB1 algorithm. The difference being, instead of saving all the instances for classification purposes, like IB1, IB2 only saves misclassified instances (Aha et al. 1991). The algorithm begins by saving the first instance in the data set, as there must be at least one instance to find the closest neighbor. IB2 then iterates through the data set classifing each instance based on the nearest neighbor. Each time an instance is misclassified, it is added to the list of instances which are used by the classifier. Misclassified instances are near the concept boundary (Aha et al. 1991). By only storing these instances, a boundry can be defined and the algorithm can offer a significant savings in storage requirements. The IB2 algorithm will be added to the Weka workbench. Then, the algorithm will be tested on LED data sets.

The LED data set is an artificial data set which represents a number displayed with LED's. There are seven attributes (LED's) which can be used to make the numbers 0 through 9, like a digital clock. The Weka Explorer workbench has an LED data generator. This generator is called LED24, and it contains 17 irrelevant attributes. However, Aha et al. (1991) used an LED data set that contained only the 7 relevant attributes. Two sets of data will be generated. The first will use

the LED24 generator. These data sets will be referred to as LED24. The LED24 generator source code will then be altered, so the LED data will contain only the 7 relevant attributes. These data sets will be referred to as LED7.

2 Methodology

2.1 Creating The IB2 Classifier and LED7 Generator

The IB2 classifier, with a storage requirement counter, and the LED7 generator need to be added to the Weka workbench using the following steps:

1. Unzip the Weka source files from the weka-scr.jar file, which is included in the Weka folder.
2. Locate the IB1 classifier file in the weka.classifiers folder.
3. Create a subclass of the IB1 classifier which implements the IB2 algorithm describe above (see attached for java code).
4. Save the IB2.java source code file in the weka.classifier folder.
5. Add the IB2 classifier to the GUI.
6. Locate the LED24 data generator in the weka.datagenerators.classifiers.classification folder.
7. Create a subclass of the LED24 data generator, which implements the LED7 data generator described above.
8. Save the LED7 data generator in the weka.datagenerators.classifiers.classification folder.
9. Add the LED7 generator to the GUI.
10. Create a new weka.jar file which includes the new IB2 algorithm.

2.2 Creating the LED Data Set

The LED24 and LED7 data generators were each used to produce a series of data sets. These data sets contain 1000 training instances and 300 testing instances at varing noise levels. The noise level begins at 2% and increases by 1% until it reaches 18%. The training instances were set to have a seed of 1, and the testing instances were set to a seed of 2 to help ensure randomness.

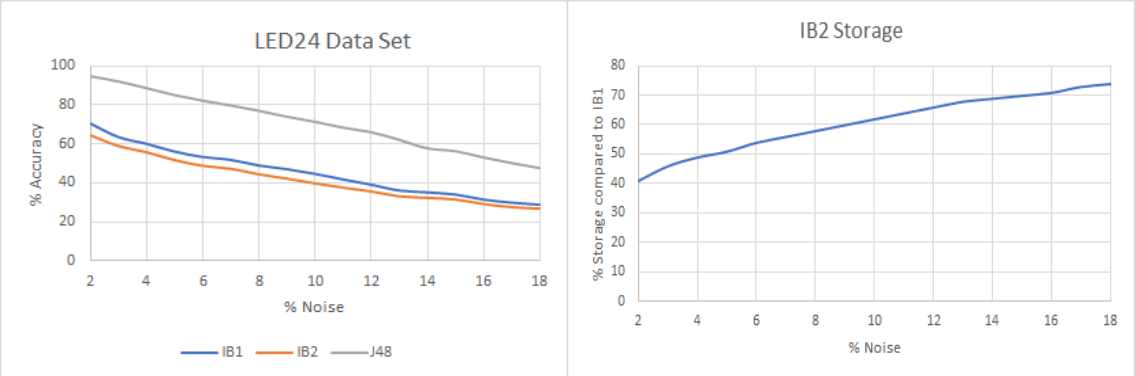
3 Results

3.1 IB2 Tested on Iris Data Sets

The IB2 classifier was first tested using the Iris data set with 10 fold cross validation repeated ten times on the Weka Experimentor. It had a slightly lower accuracy (91%) when compared to the IB1 classifier's accuracy (95%). However, it had a significanly reduced storage requirement using only 9% of the storage used by IB1. These numbers seem consistent with the results produced by Aha et al. (1991).

3.2 IB2, IB1 and J48 Tested on LED24 Data Sets

The IB2 classifier was then tested on the LED24 data sets, which contain irrelevant attributes, along with the IB1 and J48 classifiers. The IB2 classifier had an accuracy of 64% at 2% noise, which deteriorated to 45% accuracy at 10% noise and finally to 27% when the noise level reached 18%. The J48 algorithm had an accuracy of 95% at 2% noise, 71% at 10% noise and continued down to 47% at 18% noise. The storage requirements for the IB2 algorithm start at 41% at 2% noise, move up to 62% at 10% noise and end with 74% at 18% noise. See Below:



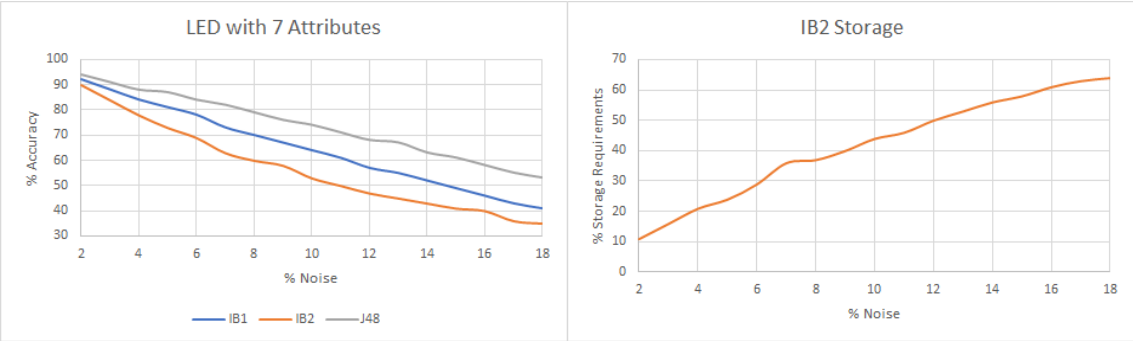
These numbers are low when compared to the results produced by Aha et al. (1991), which show IB2 with an accuracy of 62% and using 42% of the data set when the noise level is at 10%.

Aha et al. (1991) used an LED data set which contained only the seven relevant attributes and the class attribute. The LED24 data set used here contains 17 irrelevant attruibutes which are set at random. Even at the low level of 2% noise, IB2 has an accuracy of 64%. This indicates that these irrelevant attributes have a

negative impact on the IB2 algorithms accuracy, although the J48 classifier is only slightly affected, if at all. Both the noise and the irrelevant attributes are impacting the IB2 algorithm. Although the numbers dont quite match, this would agree with the results produced by Aha et al. (1991), which state that noise and imperfect attributes cause misclassified instances. These misclassified instances are saved, which increase storage requirments, and inturn misclassify other instances, which then must be saved.

3.3 IB2, IB1 and J48 Tested on LED7 Data Sets

Finally, the IB2, IB1 and J48 classifiers were tested on the LED7 data sets. These data sets contain only the seven necessary attributes and the class attribute. This should isolate the effect of noise on the classifiers. The IB2 algorithm had an accuracy of 90% with a storage requirement of 11% at the 2% noise level. IB2's performance deteriorated to 35% with a 64% storage requirement at the 18% noise level. J48 has an accuracy of 94% when the noise is at 2% and deteriorates to 53% as the noise increases to 18%. See Below:



The IB2 algorithm is much more comparable to the J48 algorithm when the noise is low for the LED7 data sets. The left figure shows the IB2 algorithm's accuracy dropping much faster than the other algorithms as the noise level increases. The right figure shows the storage requirements of the IB2 algorithm increasing as the noise level increases.

At 10% noise, IB2's accuracy is 53% with 44% storage requirements. These numbers are much more comparable to the results produced by Aha et al. (1991), which are 62% accuracy and 42% storage requirements at the 10% noise level for the IB2 algorithm.

4 Conclusion

These results tend to agree with the results produced by Aha et al. (1991). The IB2 algorithm can provide good accuracy at low noise levels with well defined attributes. It also offers a significant reduction in storage requirements when compared to the IB1 algorithm under these conditions. However, these benifits deteriorate quickly with noise or irrelevant attributes. As the noise level increases, the IB2 algorithm has a tendency to store noisy instances which inturn misclassify subsequent instances. This has a major impact on the accuracy and storage requirements of the IB2 algorithm. Noisy instances lead to misclassifications which lead to more data stored. The algorithm also showed deficiencies with data that has irrelevent or imperfect attributes. This occurs as a result of attributes not describing the target concept, which results in a large number of misclassified instances. They are poor clasifiers of similar instances. This causes IB2 to lose accuracy and increase storage requirements, similar to the noisy instances.

References

Aha, D. W., Kibler, D. & Albert, M. (1991), 'Instance-based learning algorithms', *Machine Learning* **6**, 37–66.