Data and Knowledge Engineering

Assessment

Data Pre-processing and Incremental Learning

Submitted by:

Johan Sebastian Ramirez Vallejo

1. Data Pre-processing

1. Assume that the Airlines dataset has some missing values. Out of the missing value imputation techniques (i.e., Mean Imputation, EMI and DMI) discussed in this subject, which technique do you prefer to use for the Airlines dataset? Why? [5 marks]

- I prefer to use the DMI technique because using decision trees allows us to find horizontal segments that likely find a high correlation among the attributes. Therefore, I can get better accuracy. This technic uses the EMI technic to achieve the prediction of the value, this is done to find the higher covariance within each segment to archive better imputation. EMI use the whole dataset to carry the calculation record by record which takes much processing time to predict each value. Mean Imputation technic finds the average in the whole dataset but the average is not a precise value and also just takes into account the numeric values. Therefore, the correlation in each segment is higher by DMI technic than with the whole dataset by EMI. Then, DMI can produce better results. The Airline dataset contains 539383 records therefore the processing time will be high.

2. Again, assume that the airline dataset has some missing values and some corrupt values. Do you prefer to handle the missing values before handling the corrupt values? Why? [5 marks]

- Choosing the best technic stands for how corrupt is the dataset or how many values are missing and the impact. To approach the airline dataset first, it is applied DMI technique to the whole dataset for missing values. Second, it is applied the Corrupt data detection technique and finally, it is applying the DMI again. I prefer to use this technique because it can get better accuracy but affect the processing time because more steps are done. the purpose of the dataset can increase certainty in business decision-making. Therefore, I chose accuracy over performance time. The processing time is inversely proportional to accuracy. Considering the purpose of the Airline dataset, delays cost a lot of money therefore, the model should target the highest accuracy.

2. Incremental Learning

The evaluation procedure of a learning algorithm determines which examples are used for training and which are used for testing the model output by the algorithm. In this case, the dataset airline was tested with **interleaved test-then-train** which takes each example to be used to test the model before it is used for training, therefore the accuracy result will be incrementally updated. The scheme is implemented in MOA using a landmark window model, which means that data from the stream is always considered (Bifet et al., 2018).

The dataset consists of 539383 instances and 8 attributes. Each instance indicates a historic flight record including the class value specifying if the flight was delayed. therefore, the task or output is to predict if a flight will be delayed. The dataset is used as steam (input) to evaluate the next algorithms. The accuracy and other measures are compared, but first, it is required to understand the algorithm:

Hoeffding tree: It is a very fast decision tree algorithm for streaming data, the algorithm waits for new instances, instead to reused the new instances. It builds the tree fast but requires a large amount of data to be reliable in their prediction(Bifet et al., 2018).

Hoeffding Adaptive Tree: it is an adaptative extension of the Hoeffding Tree as your name indicates that adapts to change by constructing tentative branches as preparation for change. It uses ADWIN which is a change detector and error estimator that guarantees performance and does not require parameters related to change control. It checks whether to substitute alternative subtrees(Bifet et al., 2018).

Leveraging bagging: it is also used in evolving data streams. It uses ADWIN (ADaptive WINdowing) for detection change. Thus, it improves the performance of a single tree and increases the accuracy and diversity of a single classifier. This is done by adding more randomization using Poisson distribution increasing resampling in the input and using detection codes in the output of the classifier (Bifet et al., 2010).

Adaptive random forests: it is a random forest for evolving streams. (Gomes et al., 2017).it increases diversity by selecting a random subset of features for node splits while using the Hoeffding tree in the forest. This is done using all Hoeffding tree attributes to find the best one for splitting. It uses an effective resampling base in the bagging algorithm and a bootstrap aggregation process(Alkazaz & Saado Kharouki, 2020). This means that also uses the Poisson distribution, thus as each new instance arrives it is used to train the model.

The task was set up as it is shown in the below figure, where the sampling frequency was changed to 10.000 to indicate that take a sample every 10.000 samples, t

Figure 1. Example moa setup

Mesures	Hoeffding Tree	Hoeffding Adaptive Tree	Leveraging Bagging	Adaptive Random Forest
Accuracy	65%	64%	63%	66%
Kappa	27%	25%	23%	30%
Kappa Temp	17%	14%	12%	19%
Time in seconds	8.34	16.19	426.95	363.5
Memory	15%	10%	30%	28%

Table 1. Algorithms MOA comparison

The comparison between algorithms shows that the Hoeffding tree algorithm was the fastest in building the model in almost 9 seconds followed by the Hoeffding adaptive tree in 16 seconds. In contrast, leveraging Bagging takes about 7 min and the adaptive random forest takes 6 min. This is a huge difference due to the use of ADWIN using the Poisson distribution to randomize the input.

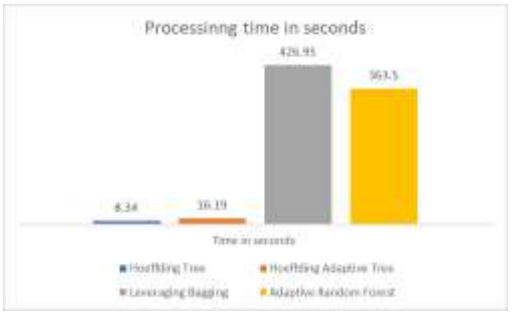

Figure 2. Processing time Comparison

The Kappa statistic stands out as the percentage that the classifier is always correct therefore if it is closer to k=1, however, the kappa for this algorithm is not expected according to the observed. Therefore, the best result was obtained by the ARF indicating that is always correct at 30%, followed by the holding tree at 27% and HAT at 25%, and the adaptive leveraging Bagging it is the lowest at 23%.

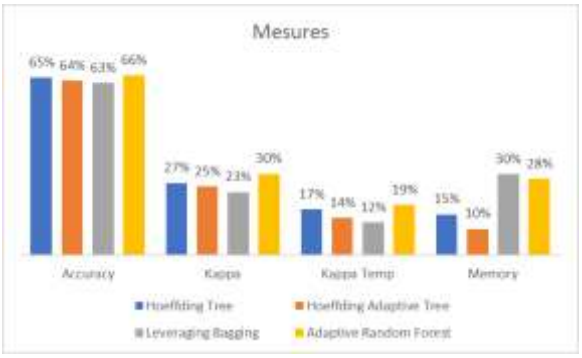

Figure 3. Measure Comparison

Surprisingly, the holding adaptive tree outperforms others over the use of memory during the processing time. It indicates that used 10% of memory. In contrast, to the leveraging bagging, the cost of memory was 30 %, ARF was 28% and Hoeffding Tree 15%.

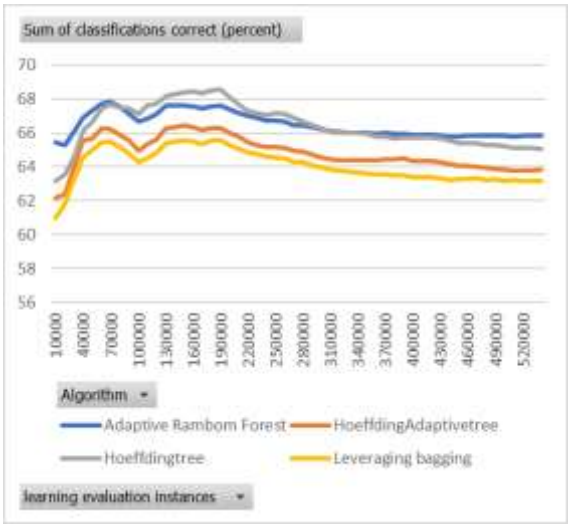

Figure 4. Learning Curve Comparison

Comparison of the learning curve of each algorithm adaptive random forest outperforms the others. its start point even is closer to the best accuracy. Therefore, it does not need a high bunch of data to obtain good accuracy, its behaviour moved between 65% to 68% so the learning curve was short while the holding tree 63% - 69% was the larger. Leveraging bagging learning curve was 61% - 65.4% and the holding adaptive tree was 62%-66.7%.

Streaming data can be processed faster with holding tree but Adaptative random forest can produce a better accuracy at any point because the learning curve is the shortest

3. References

Alkazaz, A., & Saado Kharouki, M. (2020). Evaluation of Adaptive random forest algorithm for classification of evolving data stream.

Bifet, A., Gavaldà, R., Holmes, G., & Pfahringer, B. (2018). *Machine learning for data streams: with practical examples in MOA*. MIT Press.

Bifet, A., Holmes, G., & Pfahringer, B. (2010). Leveraging bagging for evolving data streams. *Joint European conference on machine learning and knowledge discovery in databases* (pp. 135-150). Springer.

Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G., & Abdessalem, T. (2017). Adaptive random forests for evolving data stream classification. *Machine Learning, 106*(9), 1469-1495.