Journal of Machine Learning Research 1 (2020) 1-6 Submitted 9/13; Published 9/13

Winnow-2 and Naive Bayes

Brian Loughranǎ BLOUGHRAN618@GMAIL.COM
Department of Machine Learning
Johns Hopkins University
Baltimore, MA 21218, USA

Editor: Brian Loughran

Abstract

This paper describes two algorithms, winnow-2 and naive-bayes, as methods for creating models to predict against real-world data sets. Included as a part of the paper is a problem statement including assumptions made with the algorithms and projections on how each algorithm will perform. Also discussed is a description of the experimental approach, including assumptions made for each of the algorithms. Results for each algorithm are presented, and some discussion on the behavior of each algorithm is included. Finally, conclusions are drawn for the performance and bias for both winnow-2 and naive-bayes.

Keywords: winnow-2, naive-bayes

1. Problem Statement

Both winnow-2 and naive-bayes are linear classifiers, thus expectations for each should be similar. A multitiude of data sets are considered to test each algorithm, including breast-cancer-wisconsin, iris, glass, soybean-small and house-votes-84. Each of the algorithms work by using a subset of the data set to train and tune the data model then use that model to make predictions on a testing subset of the data. The predictions are compared to the actual results for the testing subset of the data set. Performance for each algorithm is given by the amount of correct/incorrect predictions utilizing a 0/1 loss function.

Some of the data sets are projected to behave more linearly, including iris, soybean-small and house-votes, thus both algorithms should work better against these data sets. Data sets that are projected to behave less linearly, including breast-cancer-wisconsin and glass, thus winnow-2 and naive-bayes are projected to work less well for these.

Another variable that may effect the performance of the winnow-2 and naive-bayes algorithms is the amount of data in the data set. The more data in the data set, the greater the amount of data that can be considered for building the data model (a 2:1 ratio of training data to test data is used in the winnow-2 and naive-bayes implementations).

It should be noted that winnow-2 is not a true learning algorithm, meaning that it is not recommended for any real data sets, and is reccomended more for theoretical applications. Thus it is expected that naive-bayes would outperform winnow-2 for most, if not all of the data sets. That being said, the margin should not be too great between the two algorithms, since both are linear classifiers.

Due to the effect of the number of data in the data set combined with the complexity of the data to learn, it is expected that winnow-2 and naive-bayes will perform similarly, how-

ever they will perform best on house-votes-84 and iris due to the mostly linearly seperable data and high number of test cases, decently on breast-cancer-wisconsin and soybean-small, and less well glass due to the low number of test cases for glass and the non-linear nature of the data set.

2. Experimental Approach

Both winnow-2 and naive-bayes are well-defined algorithms, thus the algorithm for each will not be discussed here. To read in each of the data sets however, some assumptions must be made. Each of these assumptions are discussed in this section in order to solidify the experimental approach taken for each of the winnow-2 and naive-bayes.

Winnow-2 is an algorithm that expects binary data for both the input and expected result. None of the data sets utilize binary data, thus some tricks are employed to convert the input data, which is either multi-value discrete or real valued. Naive-bayes can accept multi-value discrete, but not real valued data, however for a fair comparison with winnow-2 data for the naive-bayes implementation will be binary as well. To convert the multi-value discrete data, one hot encoding is employed to convert that data to binary. For the real valued attributes a discretization method is used to bin each of the attributes into a small set of groups. Equal width binning is used for the discretization for the real valued attributes.

For each of the algorithms a decision needs to be made if there is any missing data in the data set. Some of the data sets do not have any missing data. Glass, soybean-small and iris all have 0 missing attributes, thus no determinations need to be made for these data sets. Breast-cancer-wisconsin has 16 missing attribute values. This is a relatively small amount of missing data in a data set with 699 instances, thus instances with missing attributes are ignored for the breast-cancer-wisconsin data set. House-votes-84 has a high number of "missing" attribute values, making dropping instances with missing data a poor choice for this data set. Important to note for the house-votes-84 data set is the fact that each of the "missing" values does not indicate missing, but indicates a house member abstaining from a vote. Thus for the house-votes-84 data set the "missing" values will be treated as a separate value and will be one-hot-coded. No data imputation methods are used for any of the data sets.

For training the data model, it is prudent to have the training set representative of the testing set. Some of the data sets (glass, soybean-small, iris) are ordered by the result of the set. Thus, if training was done on the first 2/3 of the data set, the training set would not be representative of the test set. To fix this, the order of the data set is randomized before processing in order to create a more representative training set. It should be noted that because the randomization process is random, not every execution of the algorithms for glass, soybean-small and iris will produce the same data models or the same results.

Because each of the data sets require different decisions to be made for processing the data sets (which values to one-hot-code, which values to discretize, whether to re-order the data, missing attributes etc.) an options file is created for each data set with a .options.yaml extension. This file specifies each of the decisions that has to be made for each of the data sets and automates the preprocessing of the sets.

Winnow-2 uses a tuning process to find the best values for alpha and theta for the winnow-2 algorithm. 10 percent of the data is taken out for tuning, leaving the remaining

90 percent of the data set to be split by the aforementioned 2:1 ratio between training and testing. Models are built for given theta and alpha values and tested against the 10 percent tuning data. The alpha and theta values with the best performance against the tuning data is used as the model to compare the test data against. Alpha and theta values are generated for tuning by a grid search.

Winnow-2 is an algorithm that is designed for 2-classification problems. For the data sets with more than 2 result classes, some work needs to be done to make the data sets look like 2-classification problems. For this implementation, k models are generated for each of the k classes. Each of the k models are tuned and trained with the data and weight, alpha and theta values are all recorded for the best tuned model. Then for testing, the sum is computed as usual as the product of the weights and attribute values, however instead of comparing to a threshold, each of the models are compared against each other with the most confident of the k models chosen as the predicted class. Most confident is determined by taking the sum and normalizing by dividing the sum by the corresponding theta value for the model. The highest normalized sum is chosen and that predicted result is compared against the actual result for 0/1 loss as with 2-classification problems.

Another hyperparameter that can be considered for tuning is the size of the bins for discretizing real-valued attributes. While this can be included in an automated tuning process, for this project the size of the bins is tuned manually and specified in the options file for each of the data sets. Since naive-bayes has no other parameters to tune other than the bin size, no other tuning is done for the naive-bayes implementation.

3. Results

Both winnow-2 and naive-bayes were run on five different data sets, for a total of 10 experiments. The results of each of the experiments is described in this section, as well as comparisons between the two algorithms on the data sets. The experiments are ordered below in alphabetical order of the data set. Due to space limitations, items such as attribute weights, tuned alpha and theta values, class probabilities and conditional probabilities can not be included in this section. For more verbose presentation of the results including the items discussed above, each of the experiments has a corresponding .output.txt file which includes all of the values described above in a human readable format. The name of the output file for each of the experiments can be determined by the following tables:

	Winnow-2
breast-cancer-wisconsin	breast-cancer-wisconsin-winnow.output.txt
glass	glass-winnow.output.txt
iris	iris-winnow.output.txt
soybean-small	soybean-small-winnow.output.txt
house-votes-84	house-votes-84-winnow.output.txt

Table 1: Winnow-2 output file names

	Naive-Bayes
breast-cancer-wisconsin	breast-cancer-wisconsin-naive-bayes.output.txt
glass	glass-naive-bayes.output.txt
iris	glass-naive-bayes.output.txt
soybean-small	soybean-small-naive-bayes.output.txt
house-votes-84	house-votes-84-naive-bayes.output.txt

Table 2: Naive-Bayes output file names

There is a lot of information included in the output data files, and referencing those files will give greater insight into low-level items not included in this report such as items such as attribute weights, tuned alpha and theta values, class probabilities and conditional probabilities. A summary of the items included in each of the output files is shown below for each of the two algorithms:

Winnow:

- Demonstration of mapping of non-binary categorical variables to one-hot coding
- Demonstration of discretization method for the real-valued features
- Preprocessed dataframe after executing all the options in the .options.yaml file
- Demonstration of promotion and demotion for given rows in the data set
- The result of the training and tuning portions, including the best values for alpha and theta as well as the trained weights for the corresponding alpha and theta values
- The result of testing each of the test sets against the winnow-2 data model including the predicted and actual results
- A human-readable output of the winnow-2 data model
- The accuracy of the model on the test data

Naive-Bayes:

- Demonstration of mapping of non-binary categorical variables to one-hot coding
- Demonstration of discretization method for the real-valued features
- Preprocessed dataframe after executing all the options in the .options.yaml file
- The number of instances of each of the results class as well as the probabilities for each class
- The attribute probabilities for each class for each attribute
- The result of testing each of the test sets against the naive-bayes data model including the predicted and actual results
- A human-readable output of the naive-bayes data model

- The accuracy of the model on the test data

While there is not space to tabulate the weights for each of the trained models, there is room to summarize the results of the model accuracies for each of winnow-2 and each of the data sets (breast-cancer-wisconsin, glass, iris, soybean-small and house-votes-84). The accuracy for each of the experiments are included below in Table 3.

	Winnow	Naive-Bayes
breast-cancer-wisconsin	98.5%	98.5%
glass	44.6%	64.6%
iris	95.6%	100%
soybean-small	86.7%	100%
house-votes-84	90.8%	91.6%

Table 3: Model accuracy for winnow-2 and naive-bayes against multiple data sets

While detailed discussion on weights can not be included in this report due to space constraints, some discussion on weights and class probabilities is warrented. For winnow-2, the weights can be used to indicate which of the attributes have a large effect on the result. A high weight would indicate a strong positive correlation between the attribute and the result class, while a low weight would indicate either a weak correlation or a strong negative correlation between the attribute and the result class. By looking at the weights we can begin to determine which attributes are important to the result, and further analysis can be done looking at those attributes. Similarly for naive-bayes, each result class has the conditional probability for each of the attribute values. Analyzing the conditional probabilities can help to determine correlation between the attribute and the result class, where high conditional probability can indicate high correlation between an attribute and a result class.

For the breast-cancer-wisconsin data set, higher weight values for higher values of clump thickness, cell size uniformity, cell size shape, marginal adhesion, bare nuclei, combined with lower weight values for lower values would indicate a strong positive correlation between these attributes and the result of a malignant cancer. Weight values circling 1 for attributes such as single epithesial cell size, bland chromatin, and mitoses would indicate weaker correlation between these attributes and the result class. And relatively high weight values for the middle range of normal nucleoti (normal nucleoti between 4 and 7) would have a strong correlation with a malignant cancer. Naive-bayes can produce similar insights into the correlation between attributes and result class. High probability for benign result class for low values and high probability for malignant at high values would indicates a strong correlation between clump thickness, cell size uniformity, cell size shape, marginal adhesion and single epiteshial cell size would mostly agree with the winnow-2 insights. More scattered values for bare nuclei, bland chromatin, normal nucleoti and mitoses would indicate weaker correlations.

Similar determinations on attributes and result classes and their correlation can be made for the other data sets.

4. Algorithm Behavior

Winnow-2 is an algorithm that learns by making mistakes and adjusting the data model to account for the mistakes made. Thus, one behavior of the winnow-2 algorithm is to make the majority of its mistakes at the beginning of training while making less mistakes as the algorithm learns. This makes sense for the algorithm, and is likely to be similar for most supervised learning algorithms. Naive-bayes does not make the "mistakes" during the learning period that winnow-2 makes, since during the learning period it is just counting class and conditional attributes to be utilized in the testing period.

For winnow-2, selecting a good theta value was important in getting a model to train quickly on the relatively sparse data sets given. While for a sufficiently large data set, any theta value should produce similar results, it was more important with the smaller data sets to start minimizing mistakes as quickly as possible in the learning phase. Selecting theta to be a similar order of magnitude as the number of attributes for a data set usually produced decent results.

5. Conclusion

As discussed in the problem statement section, winnow-2 and naive-bayes are linear classifiers and should perform better given large data sets. Because of this, it was predicted that the algorithms would perform well on house-votes-84 and iris, decent on soybean-small and breast-cancer-wisconsin, and poorly on glass. After running the models for each of the experiments, we can check how accurate the predictions were.

Winnow-2 and naive-bayes performed very well against iris and breast-cancer-wisconsin. High performance against iris was expected due to the linear nature of the data and the number of test cases. High performance against breast-cancer-wisconsin was less expected, however this may indicate that the breast cancer data is more linear that previously expected.

Winnow-2 and naive-bayes performed decently against soybean-small and house-votes-84. This was expected for soybean-small due to the low number of test cases. However, it should be pointed out that naive-bayes did very well against soybean-small, possibly indicating a highly linear behavior for that data set. Winnow-2 may have performed better with more test sets. Performance against house-votes-84 was worse than expected, which would indicate that the house votes did not cleanly seperate democrats from republicans. I was expecting the votes to differentiate the two parties more, however that may be biased based on how polarized the current political system is. Perhaps there was more of a gray area for democrats and republicans to go against party consensus in 1984.

Glass performed predictably poorly for both algorithms. Given a high number of result classes, a low number of test data, and poor linear separation between the types of glass, this was expected. I would not recommend winnow-2 or naive-bayes for this classification problem.