PROJECT 1 - WINNOW2 AND NAIVE BAYES

DAVID ATLAS

Abstract. This paper will introduce the Winnow2 algorithm and the Naive Bayes algorithm. Both will be treated in context of classification problems, with strategies for categorical and continuous valued inputs. We find that in several experiments on real-world datasets, both algorithms perform comparably, especially when Naive Bayes is limited to booelan inputs and outputs.

1. Problem Statement & Hypothesis

We seek to introduce two basic supervised learning methods, and apply them to classification problems.

The first, Winnow2, is a very simplistic learning model for boolean inputs and outputs. The second, Naive Bayes, is a simple graphical model that can handle multivalued discrete and continuous inputs.

We will compare both models using Boolean-values inputs and outputs to make the comparison more fair, while also applying Naive Bayes in its more flexible context with multivalued discrete and continuous inputs.

Both algorithms do not explicitly treat interactions between variables (Winnow2 will adjust the weights for all the active weights, and Naive Bayes assumes the feature values are conditionally independent on the class), so large differences in performance between the two are not expected. However, when Naive Bayes is given the capacity to treat continuous inputs via a Gaussian conditional distribution, it might lead to performance inprovements.

Additionally, given the simplicity of the algorithms, it might be reasonable to expect Winnow2 to perform well on large feature spaces, as it has an implicit regularization mechanism (weights are only updated if boolean features are active). Naive Bayes may struggle in those large spaces with overfitting. This issue might also extend to class imbalances, as there won't be many training instances over which to estimate the conditional distributions.

In summary, the hypothesis is that of rough performance parity, with Winnow2 having an advantage in large feature spaces, and Naive Bayes having an advantage with continuous features.

2. Description of Algorithms

Winnow2. The Winnow2 Algorithm is a linear-threshold boolean classifier, in which a set of boolean inputs is mapped to a single boolran output. The algorithm works as follows:

Given a boolean input vector, $X = (x_1, \cdots, x_k)$, create a vector of weights $W = (w_1, \cdots, w_k)$. We also initialize two parameters, θ and $\alpha > 1$. For each example, we calculate $Z(X, W) = \sum_{i=1}^{k} x_i w_i$. If $Z \geq \theta$, we predict a 1, and if $Z < \theta$, we predict a 0.

Our learning mechanism is composed of two operations - promotion and demotion. Under promotion, we update $w_i = \alpha w_i \forall i \mid x_i = 1$. Under demotion, we update $w_i = \frac{w_i}{\alpha} \forall i \mid x_i = 1$. If our prediction is a 1, while the correct response is a 0, we apply demotion. If our prediction is a 0 while the correct response is a 1, we apply promotion. This has the effect of raising the value of Z when our function predicts a value that is too small, and lowers the value of Z when our function predicts a value that is too large.

Naive Bayes. The Naive Bayes algorithm is a graphical model that leverages Bayes' Rule and some simplifying assumptions. To recap, under Bayes' Rule:

$$P(C \mid X) = \frac{P(X \mid C)P(C)}{P(X)}.$$

Suppose C is the random variable representing the class of an instance, while X is a set of random variables representing feature values of a given instance. In creating a classifer, we want to know that likelihood that a given instance belongs to class C, given its feature values X, and so if we can calculate the right hand side of Bayes' Rule, we have a classifier that reflects our goal.

To make the equation above more managable, a simplifying assumption can be made - suppose all of the features are independent conditional on the class. Then we can say that for a set of k feature values,

$$P(C \mid X_i, \cdots, X_k) = \frac{P(X_i, \cdots, X_k \mid C)P(C)}{P(X_i, \cdots, X_k)} = \frac{\prod_{i=1}^{k} P(X_i \mid C)P(C)}{P(X_i, \cdots, X_k)}$$

We can define our classification function as choosing the value of C that is most likely, given the feature values, and by extension:

$$\arg\max_C P(C \mid X_i, \cdots, X_k) = \arg\max_C \prod_{i=1}^{k} P(X_i \mid C)P(C).$$

Note that the denominator above can be dropped, as its value is not impacted by the class of a given instance, and so it won't affect the $argmax_c$ operator.

With that background, $P(X_i \mid C)$ and $P(C)$ must be estimated. If X_i is a discrete random variable, $P(X_i \mid C)$ can be described as multinomial, where p is the proportion of each value of X_i for training instances of class C. If X_i is continuous, $P(X_i \mid C)$ can be described as Gaussian, with a mean of the sample mean of X_i for training instances of class C and a standard deviation of the sample standard deviation of X_i for training instances of class C.

$P(C)$ is multinomial, with p equal to the proportion of total training examples of class C. $P(C)$ would be binomial if it is boolean under this model.

Note that all of these distributional family assumptions can be changed if appropriate, but are generally sensible defaults where no other distributional information is known.

Smoothing can be applied in situations where there are no training examples of class C that take on a value $X_i = Q$ (where Q is a constant), to impose a non-zero likelihood for $P(X_i \mid C)$, which can lead to better generalization. For a multinomial distribution, $p_i = \frac{\sum_{i \in C} x_i + \alpha}{\sum_{i=1}^{n} x_i + \alpha m}$, where $\alpha = 1$ and m is the number of values in X_i. Obviously, this can be done in many other ways, but this is a sensible default. For a continuous distribution, the distribution over all classes can be used as a prior.

After calculating the conditional distributions of the training examples, the prediction is simply

$$\arg\max_C \hat{P}(X_i \mid C)\hat{P}(C),$$

where $\hat{P}$ indicates the estimated distribution from the training process.

3. Experimental Approach

The experiments conducted included 5 datasets:

(1) The iris dataset classifying plant species from leaf measurements.
(2) The breast cancer dataset classifying tumors based on breast measurements.
(3) The glass dataset classifying the origin of broken glass based on measurements of the shards.
(4) The soybean dataset classifying rot based on crop information.
(5) The congressional voting dataset classifying party based on legislation votes.

For each dataset, discrete inputs were one-hot encoded as booleans, and continuous values were discretized based on boxplots of their distributions, segmented by class, to attempt to provide one-hot encodings that best seperated the classes. For multivalued outputs, one vs. rest classifiers were fit. For Winnow2, the same threshold θ was used for all the classes, and the highest value of Z was selected across each classifier. For Naive Bayes, the highest likelihood class was selected.

For each dataset, Winnow2 and Naive Bayes algorithms were used on boolean inputs and outputs for comparability. Additionally, Naive Bayes was fit using multivalued discrete and continuous attributes. A 5 fold randomly-shuffled cross-validation approach was used to evaluate the models.

4. Experimental Results

4.1. **Iris Dataset.**

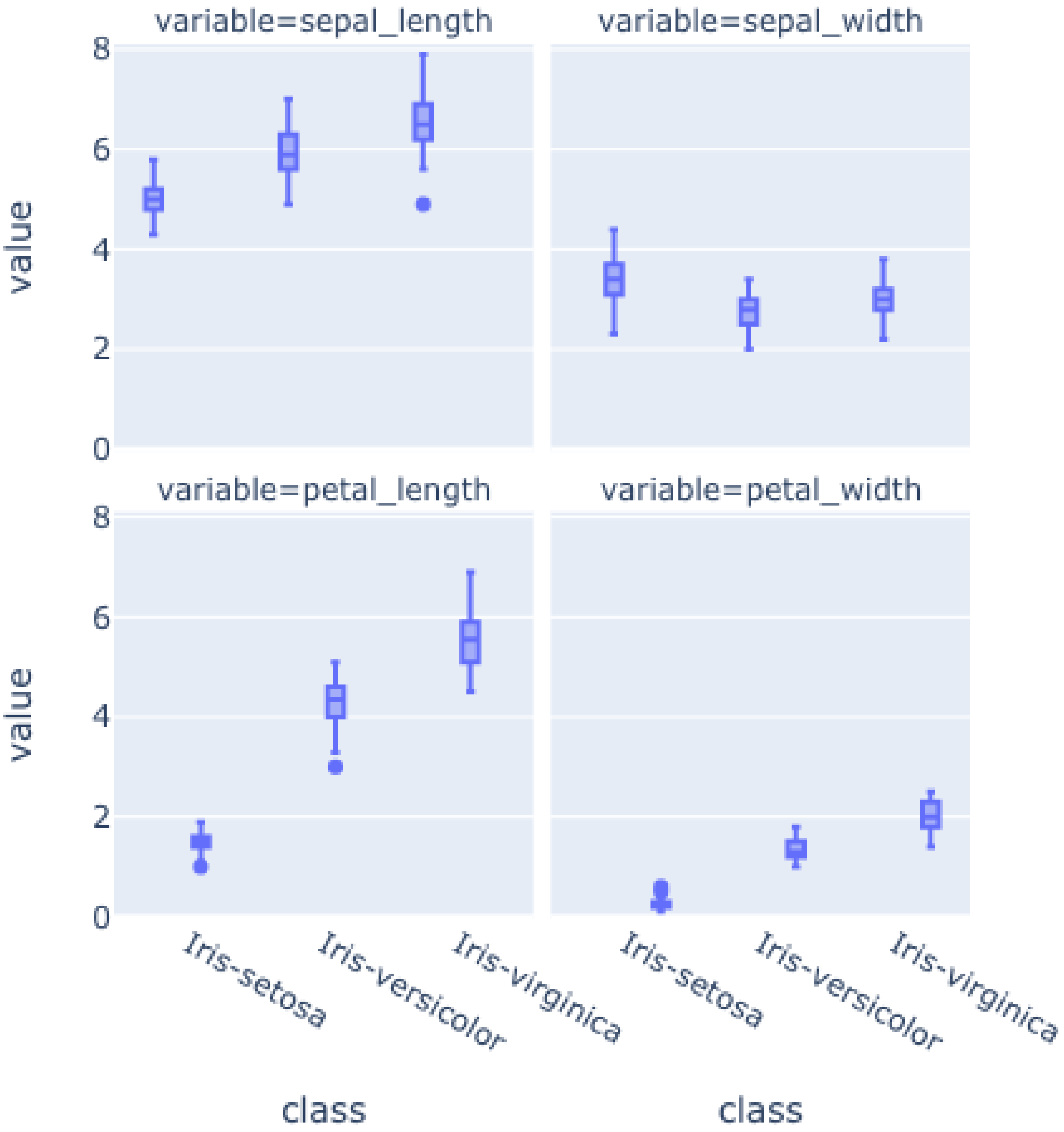

Figure 1. Iris dataset boxplot of feature values by class.

	Range 1	Range 2	Range 3
Sepal Length	[0, 5.5]	(5.5, 10]	
Sepal Width	[0, 3]	(3, 10]	
Petal Length	[0, 1]	(1, 1.6]	(1.6, 10]
Petal Width	[0, 2]	(2, 5]	(5.5, 10]

	Setosa	Versicolor	Virginica
Setosa	9	0	0
Versicolor	0	0	1
Virginica	0	8	12

Table 1. Iris - Winnow2 Confusion Matrix

	Setosa	Versicolor	Virginica
Setosa	9	0	0
Versicolor	0	5	6
Virginica	0	3	7

Table 2. Iris - Naive Bayes with Boolean Inputs Confusion Matrix

	Setosa	Versicolor	Virginica
Setosa	9	0	0
Versicolor	0	8	1
Virginica	0	0	12

Table 3. Iris - Naive Bayes with Continuous Inputs Confusion Matrix

	Benign	Malignant
Benign	86	0
Malignant	0	50

Table 4. Cancer - Winnow2 results.

Data Cleaning & Transformation. The first dataset presented is the Iris dataset. First, the continuous inputs are discretized. Figure 1 was used to pick the cutoff points by finding values that best seperate the classes in a univariate sense. The following cutoff points were used: One-hot encodings were used on the discretized features.

Model Fitting Results. The model results for one of five random folds are shown in Table 1. Note that for the confusion matrices here and below, the column headers are the actual labels, while the row headers are the predicted values. Values along the diagonal indicate correct predictions. The model results for Naive Bayes with boolean inputs are shown in Table 2 The model results for Naive Bayes with boolean inputs are shown in Table 3 These results indicate that the classes are easily seperably in a 4D space of all the features, as the continuous Naive Bayes algorithm outperforms the discretized version. This lines up with the hypothesis presented above - continuous inputs with linearly seperable classes are a good fit with Naive Bayes.

4.2. **Cancer Data.**

Data Cleaning & Transformation. For the cancer dataset, all null values are dropped, as this is only a small proportion of total rows (16/699). The inputs are multivalued discrete, so one-hot encoding representations are made of each column.

Model Fitting Results. The model results for Winnow2 with one-hot encoding variables are shown in Table 4. The model results for Naive Bayes with one-hot encoding variables are shown in Table 5. The model results for Naive Bayes with multinomial input variables are shown in Table 6. The results here indicate near performance parity. Winnow2 does quite well, as this problem has many attributes, only some of which are likely relevant. The inputs are not continuous, and so Naive Bayes does not have a significant advantage.

4.3. **Glass Data.**

Data Cleaning & Transformation. The glass data features are continuous, and so the same procedure as above is followed to discretize them. Table 4.3 shows the mean by class for each of the features. The values chosen reflect points that seperate the means of the classes reasonably. The values chosen are shown in Table 4.3. One-hot encodings are made for each of the discrete cutoff points.

	Benign	Malignant
Benign	84	1
Malignant	2	49

Table 5. Cancer - Naive Bayes with boolean inputs results.

	Benign	Malignant
Benign	83	0
Malignant	3	50

Table 6. Cancer - Naive Bayes with multivalues discrete inputs results.

Class	refractive index	sodium	magnesium	aluminum	silicon	potassium	calcium	barium	iron
0	1.518718	13.242286	3.552429	1.163857	72.619143	0.447429	8.797286	0.012714	0.057000
1	1.518619	13.111711	3.002105	1.408158	72.598026	0.521053	9.073684	0.050263	0.079737
2	1.517964	13.437059	3.543529	1.201176	72.404706	0.406471	8.782941	0.008824	0.057059
3	1.518928	12.827692	0.773846	2.033846	72.366154	1.470000	10.123846	0.187692	0.060769
4	1.517456	14.646667	1.305556	1.366667	73.206667	0.000000	9.356667	0.000000	0.000000
5	1.517116	14.442069	0.538276	2.122759	72.965862	0.325172	8.491379	1.040000	0.013448

	Range 1	Range 2	Range 3	Range 4
refractive index	1.518			
sodium	12.500	14.0		
magnesium	1.000	2.0	3.25	
aluminum	1.200	1.5	2.00	
silicon	72.600	72.8	73.20	73.4
potassium	0.200	0.4	0.50	0.6
calcium	8.600	9.0	10.00	
barium	0.350			
iron	0.200	0.6		

Model Fitting Results. The confusion matrix is shown in Table 4.3 for the Winnow2 algorithm, using the discretization points above. Naive Bayes was also run with the boolean inputs, and the results are shown in Table 4.3.

	0	1	2	3	4	5
0	13	7	4	1	0	3
1	0	3	0	2	0	0
2	0	5	1	0	0	3
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

Table 7. Glass dataset Winnow2 results.

	0	1	2	3	4	5
0	8	2	4	1	0	1
1	5	10	1	2	0	5
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	2	0	0	0	0