Jacob Dineen

NETID: Jdineen

IST565

Homework 6

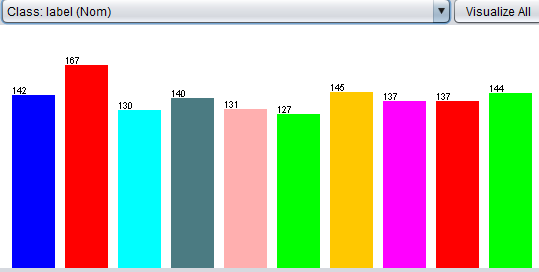
Naïve Bayes works better for digit recognition

Sampled Dataset for DigRec dataset

Section 1: Introduction

In this experiment we will be looking at a systemically sampled dataset focusing on digit recognition as it relates to different machine learning algorithms. In this case, we will be contrasting success metrics between Naïve Bayes and Decision Trees. The purpose of this exercise is to determine which algorithm presents the most accurate model in a controlled setting. The parameter tuning will be slightly different from method to method, including random seed generation, but our testing will be similar (Holdout to Holdout, Cross-Validation to Cross-Validation). With little domain knowledge on digit recognition, I start by looking at the raw data via CSV. I can see that the first column, label, is our class for this dataset, and should be set as such when creating models. The value is nominal/categorically in nature, and represents a value from between 0-9 (10 possible classes) Each row indicates an instance of a digit, so we are looking at 1400 instances. The attributes, or features, are labeled by ‘pixel’ 1..2..3.. etc. After some research, I can see that for this classification problem an image is a 28x28 pixel square, which equals out to 784 pixels. So it is not a coincidence that we have 784 features within our dataset. Loading the dataset into Weka shows me that all the features are numeric. Initial preprocessing will include the transformation of the ‘Class’ variable to nominal, and discretization of the remaining variables (I may only discretize through parameter tuning, or build models with continuous attributes first and compare with binned attributes).

We have a relatively even distribution per our class variable, which will be beneficial for our models:



\*These experiments will be conducted via Weka.

Section 2: Decision Tree

I’ve decided to run a series of controlled tests using parameter tuning to measure differences in accuracy.

Test 1 will use the J48 decision tree algorithm with default settings attached, and 3 different random seeds.

Test 2 will use the J48 decision tree algorithm with Reduced Error Pruning turned on, and 3 different random seeds.

Both tests will use five-fold cross validation, which means that there will be 5 iterations of the model ran, with a portion partitioned off into a subset that will be used for testing. The average of the results will be display after model generation, and seen below. Because we are using K means Cross Validation method, we will not be using the supplied test data set at this point in time.

The results:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Random Seed** | | |
|  | 1 | 5 | 100 |
| J48 Default Settings, 5Fold CV | 71.57% | 70.57% | 72.29% |
| J48, Reduced Error Pruning, 5Fold CV | 66.57% | 67.00% | 67.57% |
|  |  |  |  |

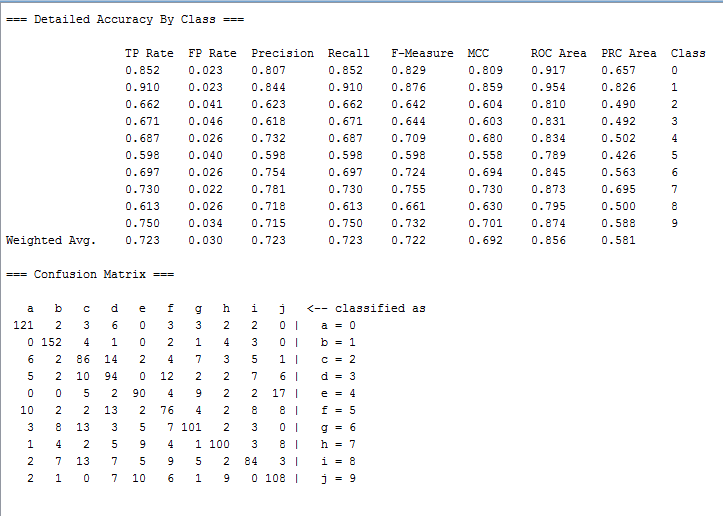
It is evident that the Reduced Error Pruning negatively impacted the accuracy of our model in this instance. It is also evident that a random seed of 100 increased our accuracy over other instances of 1 and 5, respectively.

What is of note, however, is that when Reduced Error Pruning was triggered on, the size of our tree was significantly smaller and less complex than when Reduced Error Pruning was off.

On: 60 Leaves, Size of Tree = 119

Off: 129 Leaves, Size of Tree = 257

Looking at our most successful model, in terms of accuracy, we see the following:



Looking at Fmeasure, which is a metric that combines precision and recall (2 X Precision X Recall / Precision + Recall) we can see that our weighted average was .722 or 72.2%. The class categories that fell below that average were: 2,3,4,5,8. Of those class labels noted, all but the digit 5 fell within 1 standard deviation of the weighted average.

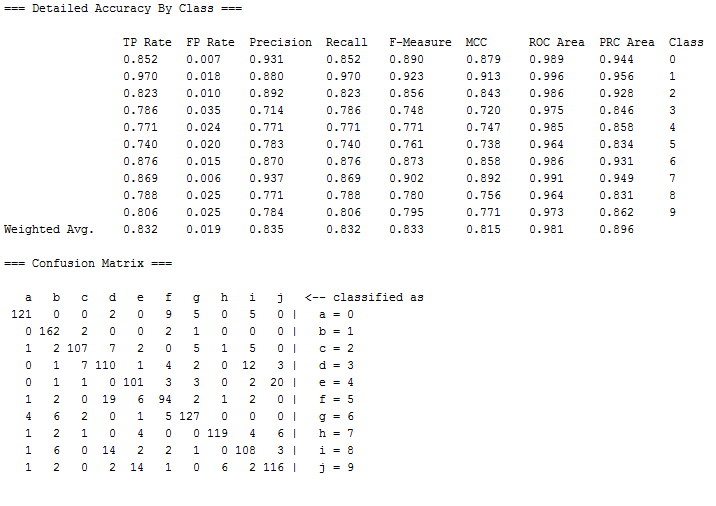
Furthermore, information gain helps us to know which features were most influential in the decision tree, by order of appearance in nodes. These are the top features:

Section 3: Naïve Bayes

Moving onto Naïve Bayes, we will perform similar tests, but with less parameters to tune, we will instead use discretization, or binning, against continuous dependent variables.

Results:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Random Seed** | | |
|  | 1 | 5 | 100 |
| NB, Default, 5Fold CV | 73.14% | 72.57% | 73.71% |
| NB, Supervised Discretization, 5Fold CV | 83.00% | 83.00% | 83.29% |



Our results show that running the Naïve Bayes algorithm with supervised discretization triggered on, and a random seed of 100, delivers accuracy at a clip of 83.29%. This is our most successful model that was generated through this experiment.

It is of note that 3,4,5 and 8 all saw dips in Fmeasure, in contrast to other digit prediction. Going back to the Decision Tree models, we remember that 2,3,4,5,8 were a bit of a struggle. Looking at both models, 3,4,5 and 8 all struggled accordingly. This leads me to believe that there is something about these digits that are more difficult to recognize, or are more easily misconceived than others.

Section 4: Algorithm performance comparison

Utilizing a controlled environment, it was evident that Naïve Bayes was the better algorithm for predicting digit recognition. Even when supervised discretization was turned off, NB still performed better than the best decision tree model.

My thought going into this experiment, after looking at the data set, was that there was not a normal distribution amongst the continuous variables, so using discretization would help improve accuracy.

It should also be noted that the NB algorithm generated, generally, at faster speeds than the decision tree models. Fewer probabilities were required to be computed in the NB models, particularly when discretization was turned on.

As mentioned, there were specific digits that each algorithm struggled to classify. Some of these digits saw crossover between the two types of models built: 3,4,5 and 8. This would require domain knowledge on numeric deciphering, or something along the lines of a linguist for numeric values to understand similarities between specific digits.