Rupali Vohra (rv5rr)

CS 4710: Classifier (Homework 4)

Tuesday, April 28, 2015

**Summary**

The implementation I chose for my classifier involved use of the Naïve Bayes method. This allowed me to easily account for both discrete and continuous variables; a simple proportion was used for discrete features, while the assumption of a Gaussian distribution for continuous variables allowed me to predict outcomes based on means and variances found from the training set.

**Approach**

Though a bit messy, the best way I could think of to keep track of all the various features, whether it was continuous or discrete, and what the outcome was with that given feature was to have multiple maps that kept track of that information. Since I knew I would not be deleting anything from the map, I decided that HashMaps would be the most efficient way to solve the problem.

I ended up creating 9 HashMaps. One of these (featureOptions) keeps track of each feature, and maps it to an array of all the options available for that feature. This was used to populate other HashMaps that are used to keep track of how many times each of these options appears. Two such HashMaps (frequencyMapGreater and frequencyMapLess) were used to keep track of the number of times each option occurred, depending on whether the outcome was “>50K” or “<=50K,” respectively. Similarly, I created two HashMaps (valueMapGreater and valueMapLess) to keep track of the values that appeared for each continuous variable, depending on whether the outcome was “>50K” or “<=50K,” respectively. The maps for discrete variables were used to calculate proportions for probabilities, and the maps for continuous variables were used to calculate the means and variances for each feature, which was then used to calculate probabilities. The probabilities calculated from the discrete variables were stored in two HashMaps (featureProbGreater and featureProbLess), which mapped probabilities for which the classification was “>50K” in the former and outcomes where the classification was “<=50K” in the latter. Similarly, probabilities for continuous variables were tracked in two HashMaps: numericValsGreater, which stored data for when the outcome was “>50K,” and numericValsLess, which stored data for when the outcome was “<=50K.”

All of the HashMaps were populated with data from the training set. The section of the training set I actually used while testing varied (more information given below). This variation did not, obviously, affect my general approach to the problem of making predictions. I did so by following the formula: . In other words, I calculated the probability that each outcome of each feature in the given test case would occur if the classification were “>50K,” and compared that value to the probability that that each feature’s outcome would occur if the classification were “<=50K,” and selected the larger probability as my classification. In the event that these two probabilities tied, I decided to return an outcome based on the decision of a random number generator.

There is a stipulation in the Naïve Bayes approach for classification that mentions the conditional independence for all features in the data set. This is, as stated in the slides, a fairly large assumption, and one that does not hold for the given training set. Specifically, the education\_num field is clearly highly correlated with the education\_level field, and probably derived from that field. Since I created my Classifier to work without accounting for this at first, I decided it would be interesting to see the difference such an account would make in estimations. The results of this experiment are detailed below.

**Testing**