CS412: Introduction to Data Mining, Fall 2017, Homework 5

Name: Nestor Alejandro Bermudez Sarmiento (nab6)

Worked individually

Introduction

In this assignment we explore two topics: classifications methods and clustering methods. For classification we will illustrate the use of a Naive Bayes classifier and also K-nearest neighbor using different values of k. Finally we will look at the k-means, DBSCAN and AGNES clustering algorithms.

Question 1

Consider the following equations:

$$Pr(X, C) = Pr(X \mid C) \times Pr(C) \tag{1}$$

$$Pr(C \mid X) = \frac{Pr(X \mid C) \times Pr(C)}{Pr(X)} \tag{2}$$

1a(i): $Pr(\text{Popularity} = \text{'P'}) = \frac{7}{10}$. Done by simply counting the number of examples that match the criteria. Known as the prior probability.

1a(ii): $Pr(\text{Popularity} = \text{'NP'}) = \frac{3}{10}$. Found by taking the value from the previous question from 1 since this is a binary classification problem.

1a(iii): By using equation 1 and class-conditional independence:

$$Pr(\text{Price} = \text{'\$'}, \text{Delivery} = \text{'Yes'} , \text{Cuisine} = \text{'Korean'} \mid \text{Popularity} = \text{'P'})$$

$$= \prod_{i=1}^{3} Pr(x_i \mid \text{Popularity} = \text{'P'})$$

where x_i is each of the features in the tuple considered. So:

$$x_1 = \frac{4}{7}, x_2 = \frac{4}{7}, x_3 = \frac{2}{7}$$

and then

$$Pr(\text{Price} = \text{'\$'}, \text{Delivery} = \text{'Yes'} , \text{Cuisine} = \text{'Korean'} \mid \text{Popularity} = \text{'P'}) = \frac{4}{7} \times \frac{4}{7} \times \frac{2}{7} = \frac{32}{343}$$

1a(iv): Pr(Price = '$', Delivery = 'Yes' , Cuisine = 'Korean' | Popularity = 'NP')

$$= \prod_{i=1}^{3} Pr(x_i \mid \text{Popularity} = \text{'P'})$$

where x_i is each of the features in the tuple considered. So:

$$x_1 = \frac{1}{3}, x_2 = \frac{2}{3}, x_3 = \frac{1}{3}$$

and then

$$Pr(\text{Price} = \text{'\$'}, \text{Delivery} = \text{'Yes'} , \text{Cuisine} = \text{'Korean'} \mid \text{Popularity} = \text{'P'}) = \frac{4}{7} \times \frac{4}{7} \times \frac{2}{7} = \frac{2}{27}$$

1b: When doing Naive Bayes classification one of the common ways to perform the prediction is to using **MAP**[1] (Maximum a posteriori). MAP indicates that we should calculate the probability of each class given the evidence. For this particular case, we have already done that in part 1a(iii) and 1a(iv).

So to answer the question it is just enough to compare the previous results. Since $\frac{32}{343} > \frac{2}{27}$ the answer to the question is **yes**, it will be class as Popular.

1c: the techniques we discussed in class can, theoretically, apply to Naive Bayes. That is: boosting, bagging and ensemble. There is no "one size fit all" solution and finding the right method usually involves trying multiple of them until you get a reasonable accuracy. One of the algorithms we can use is AdaBoost on top of Naive Bayes. This works as follows:

Let d be the number of classes and k the number of iterations or predictors in our ensemble method. These will be the hyper-parameters of our classifier.

1. make all tuples equally likely, e.g. assign the same weight to all of them. The usual value for the weigh is the inverse of the number of classes ($\frac{1}{d}$).
2. create a new data set D_i by sampling with replacement from the original data set. Whether an example is included or not in D_i is decided by its weigh.
3. train a Naive Bayes classifier using D_i.
4. evaluate the Naive Bayes classifier and find its error rate. We can do it by using the test data or by using some of the examples that were not included in D_i.
5. if the error is greater than a threshold (at least 50%), reset the weighs and try again with a new sample (go back to step 2).
6. if the error is good enough then for every tuple in D_i update its corresponding weigh by a factor of the $\frac{e}{acc}$ where e is the error and acc is the accuracy $(1 - e)$.
7. repeat from step 2 if done less than k times.

[1] https://en.wikipedia.org/wiki/Maximum_a_posteriori_estimation

Once the training is done we need to predict labels for new evidence **X**. To do so we initialize the weights with zero and on each iteration we update them by taking the log of the accuracy over the error rate. On each iteration we see what the prediction for the current classifier is and update the weight of that class only! At the end we predict the class with the highest weight.

1d: There are a few metrics that can help measure the performance of a classifier when one of the classes is rare.

i) Sensitivity: it measures the rate of true positives recognition and is defined by the formula: $\frac{TP}{P}$. Where TP is the total number of true positives (correctly classified as positive) and P is the total number of instances that were positive. Note that the formula does not include the negatives so it ignores the fact that the positives are rare because there is no point of comparison between positives and negatives.

ii) Specificity: measures how successful is the classifier in predicting false when over the examples that were indeed false. As an example, in a medical diagnosis, tests usually have high specificity, meaning that they rarely predict a positive in healthy patients, this doesn't say much about whether it was positive or not but it is a first step to perform more tests. Example inspired by this Wikipedia page

Question 2

Question 3