Assignment 2 Technical Document – COMP3106

Fareha Sultan – 100968491

2021-11-18

1. Briefly describe how your implementation works. Include information on any important algorithms, design decisions, data structures, etc. used in your implementation. [10 marks]

My implementation closely follows the methods and steps we used in class to solve problems involving the Naive Bayes Classifier and Fuzzy Classifier.

For the Naive Bayes Classifier, I started by storing the features of each class in an array called "classes" where each element is a dictionary that stores all the information needed to compute the probabilities for each class. The dictionary element inside classes[] has the following keys: class , priorProb, propBlack_mean, propBlack_sd, topProp_mean, topProp_sd, leftProp_mean, and leftProp_sd. Using a dictionary allows easy access to the values when calculating $P(feature = x|class)$. I also used descriptive variable names to distinguish between mean and standard deviation (sd). To calculate the features (propBlack, topProp, leftProp), I used separate functions, because they require to loop through different x and y values when counting black squares. I also saved the features in a dictionary where the key is the feature name and the value is the proportion. I could have just stored the proportions in an array and just kept track that at index 0 , I have the propBlack value, but I wanted my code to be easily understandable for another person. So, when I access it using the feature name, it is easy to see what I am trying to calculate. The $P(feature = x|class)$ was calculated for the three features using normal_probability_density(x,μ,σ) and then I applied the independence assumption by multiplying all the probabilities. Next, substituting the independence assumption , I derived the posterior probability of a class given a feature x [1].
The posterior probability of each class are also saved in a dictionary where the key is the class name and the values are the class probabilities. Saving it in a dictionary allows to use the methods sum for finding the denominator and max to extract the most likely class. Additionally, it's easy to update the values and print them out by converting them into a list.

For the Fuzzy Classifier, I first defined the Godel t-norm and s-norm functions.
Following the steps from class, I first compute the truth value of each proposition in the antecedent using the trapezoidal membership function. The values for a, b, c, d for each fuzzy set and characteristics are stored in an array which contains a dictionary of each feature. The key is the characteristic and the values are a,b,c and d stored in an array. Next, I compute the rule strength where I make use of the t-norm and s-norm functions. The output membership for each rule is saved in a dictionary called "class_memberships" which stores the class as key and the rule strength as value. The class_memberships is the m_combined(x) as seen in class, and we take the max value as the highest_membership_class. The output is the highest_membership_class and the values of calss_memberships.

2. What type of agent have you implemented (simple reflex agent, model-based reflex agent, goal-based agent, or utility-based agent)? [3 marks]

It is a simple reflex agent because it selects actions based only on current precept (current environmental situation). It does not keep a sequence of precepts it already sensed as in the model - based reflex agent. It basically maps the current precept into the appropriate action regardless of its previous precepts [2]. The mapping process in our case is the Naïve Bayes. Additionally, it does not consider impact of current actions on future actions as in the goal-based agent. We know that Naive Bayes Classifier assume that the presence or absence of a specific feature of a class is independent to the presence and absence of another feature for the given class. Furthermore, it also does not have a utility function that measures the desirability or happiness or utility of an environment state. So, in conclusion, classifiers such as the Naïve Bayes are agents that select actions based on only the current precept at the current time and simple Reflex agents use an “if” “then” implementation which is the case for our program.

3. Suggest a particular feature vector for this problem for the naïve Bayes classifier where using equal prior probabilities for each class changes the most likely class (compared to using the above-specified prior probabilities). [4 marks]

If I run the program with the given prior probabilities : P(A) = 0.28, P(B)=0.05, P(C) = 0.10 , P(D) = 0.15, P(E) = 0.42 as provided in the specs , with input.csv from Example 1, it returns that ‘E’ as the most likely class. Now, if I change the prior probabilities of all classes to 0.20 , I get ‘B’ as the most likely class.

This is feature vector for Example 1’s input file:
[0.515,0.48058252427184467,0.5655339805825242]
Here it is in a dictionary:
{'propBlackValue': 0.515, 'topPropValue': 0.48058252427184467, 'leftPropValue': 0.5655339805825242}

Output with Initial Prior Probabilities: "E", [0.07785911092560284, 0.2293883155199489, 0.0005872395833641492, 0.31105608334936014, 0.38110925062172396]

Output with Modified Equal Prior Probabilities: 'B', [0.03540999970538947, 0.5842191683817047, 0.000747807533823549, 0.26407175669565963, 0.11555126768342266]

4. Suggest a particular feature vector for this problem where the most likely class identified by the naïve Bayes classifier and the highest membership class identified by the fuzzy classifier are different. [4 marks]

(Just to clarify, I am back to using the probabilities as in the specs for this assignment for this question. So , the prior probabilities are the following: P(A) = 0.28, P(B)=0.05, P(C) = 0.10 , P(D) = 0.15, P(E) = 0.42 .)
This is feature vector for Example 1's input file: [0.515,0.48058252427184467,0.5655339805825242]
Here it is in a dictionary:
{'propBlackValue': 0.515, 'topPropValue': 0.48058252427184467, 'leftPropValue': 0.5655339805825242}

It gives us the following outputs.

Naive Bayes Classifier: 'E', [0.07785911092560283, 0.22938831551994887, 0.0005872395833641496, 0.31105608334936014, 0.3811092506217239]

Fuzzy Classifier: 'B', [0, 1, 0, 0, 0]

As you can see, using the feature vector [0.515,0.48058252427184467,0.5655339805825242] , the Naive Bayes Classifier gives us 'E' as most likely class and 'B' as the highest membership class identified by the Fuzzy Classifier.

5. Suppose we try to use our fuzzy classifier to classify a different letter which we have not seen before and do not have any rules for (i.e. not A nor B nor C nor D nor E). How might we identify such a case? [3 marks]

For example, let's say we input a csv file that has the letter O. O closely resembles the shape of C. So our fuzzy classifier will output one of the classes in the fuzzy set that closely resembles O. The LeftProp feature will closely resembles that of 'C'. Recall that "Fuzzy Truth" is not completely true , but it has some degree of truth. The fuzzy true value of a statement is in a range of [0,1]. The membership values will be quite low in comparison to class we have seen before.

6. Suggest a particular instance of this problem for the fuzzy classifier where the class with the highest membership function would be different if we used the Goguen t-norm and Goguen s-norm (instead of using the Godel t-norm and Godel s-norm). [4 marks]

When X = 0 , Y = 1
Godel T-norm= min(x,y)= 0
Godel S-norm= max(x,y)= 1
Goguen T-norm= x*y = 0
Goguen S-norm=x+y-x*y = 1

When X=1 , Y=0
Godel T-norm= min(x,y)= 0
Godel S-norm= max(x,y)= 1
Goguen T-norm= x*y = 0
Goguen S-norm=x+y-x*y = 1

The above two situation give us the same results.

When X and Y are less than 1 and not 0, we have 3 scenarios, X>Y, X<Y, and X=Y.

(1) Let x = 0.6 y=0.4

Godel T-norm= min(x,y)= 0.4
Godel S-norm= max(x,y)= 0.6
Goguen T-norm= x*y = 0.24
Goguen S-norm=x+y-x*y =0.76

(2) Let x = 0.3 and y=0.7

Godel T-norm= min(x,y)= 0.3
Godel S-norm= max(x,y)= 0.7
Goguen T-norm= x*y = 0.21
Goguen S-norm=x+y-x*y =0.79

(3) Let x = 0.5 and y = 0.5

Godel T-norm= min(x,y)= 0.5
Godel S-norm= max(x,y)= 0.5
Goguen T-norm= x*y = 0.25
Goguen S-norm=x+y-x*y =0.75

A particular instance would be when the X and Y value are less than 1 and not 0.

7. Suggest another feature we could use in these classifiers and explain how it could be computed on an input black-and-white image. [6 marks]

We could also account for the other proportions of the image such as the right proportion, bottom proportion, the proportion at the centre of the image.

Right proportion will be the same as Left proportion but it will start to count the number of black square from width_of_image/2 to the range of width_of_image, where width is on the x-axis. Then, divide the number of black squares in the right proportion by total number of black squares.

Bottom proportion will be the same as Top proportion but it will start to count the number of black square from length_of_image/2 to the range of length_of_image, where length is on the y-axis. Then, divide the number of black squares in the bottom proportion by total number of black squares.

To calculate the proportion of the centre of the image:
Let width be on the x-axis and length be on y-axis.

1) Let center_x = Width / 2
2) start_x = center_x/2
3) End_x = center_x + start_x
4) Loop through width from start_x to end_x and count the number of black squares and store in a variable call total_x.
5) Let center_y = length/ 2
6) start _y = center_y/2
7) End_y = center_y + start_y
8) Loop through length from start_y to end_y and count the number of black squares and store in a variable called total_y.
9) Add total_x and total_y and divide the sum by total number of black squares in the image.

8. In this assignment, the probability densities for the naïve Bayes classifier have been provided. Suppose they were not provided. Suggest a method to determine these probability densities. Describe what data you would need to do so. [6 marks]

So, if we were only focusing classes A, B, C, D, and E, we would first need to train the Naive Bayes Classifier. We would require a few images / csv files of each class . Let's say we have 50 files that represent the letter 'A', so 50 different handwritten ways of writing 'A'.
Using these images for each class, we can calculate the 3 features (PropBlack,TopProp,LeftProp). Once we have calculated the features for all 50 images of 'A', we can combine the 50 data points for PropBlack and graph them to find the mean and sd for PropBlack for the 'A' class. We repeat the same process for TopProp and LeftProp. This will be supervised learning

References:

[1] http://shatterline.com/blog/2013/09/12/not-so-naive-classification-with-the-naive-bayes-classifier/

[2] https://towardsdatascience.com/the-ultimate-guide-to-sms-spam-or-ham-detector-aec467aecd85