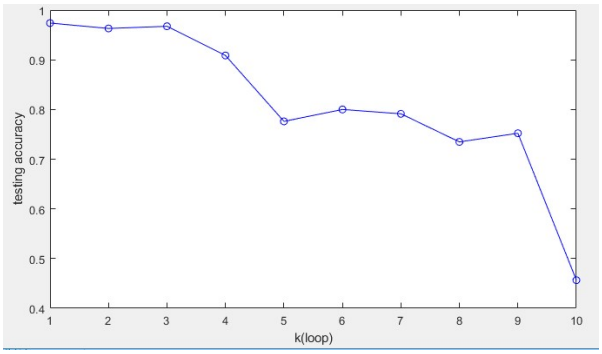
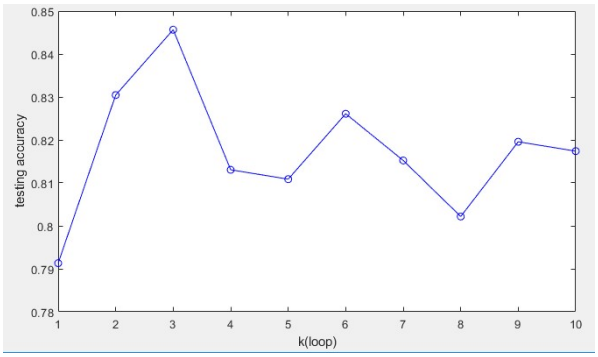
1) For discrete naïve Bayes, those parameters include ① likelihood (possibility for an example with a specific value for each feature given that it belongs to a specific class); ② probabilities that an example belongs to each class and ③ range of possible values; for continuous one, it includes ①mean and ②standard deviation of likelihoods (possibility for an example with a specific value for each feature given that it belongs to a specific class) as well as ③ general possibilities that an example belongs to a class.
2) (val_type represents number of possible values in an attribute set; lbl_type means number of classes in a database.)
   For discrete naïve Bayes:
   ①likelihoods are stored in a matrix of conditional possibility-feature scheme with size (val_type*lbl_type) by 57. Under database av2_c2, number of rows will be 2*2=4; under av3_c2, it will be 3*2=6; under av7_c3 will be 7*3=21. For each feature, conditional possibility p (x=m | c = n) is saved in row (n-1)*val_type+m+1; if the label that a possibility in a specific row is needed, it can be calculated by fix((row-1)/ val_type).
   ②possibilities for labels are stored in a vector with length lbl_type, ③number for examples of each class is stored by val_type.
   For continuous naïve Bayes:
   ①mean and ②standard deviation of likelihoods are both stored in a 2 by 57 matrix. Since the function only needs to teal with one database, number of classes is explicitly given. ③possibilities for labels are stored in a vector with length lbl_type.
   For spam classification, parameters are the same as those for continuous naïve Bayes, yet 10 groups of each of those need to be solved as a 10-fold cross validation is expected. So for each parameter in previous step, a dimension with value of 10 is added as the highest dimension.
3) Accuracy for classification under av2_c2 is 0.889613, under av3_c2 is 0.892221, under av7_c3 is 0.865652. Under av7_c2, the result is 0.769650. Mean and standard deviation of accuracies under 10-fold cross validation for spambase will fluctuate slightly, but generally they will be around 0.818 and 0.02 respectively.
4) Cross validation is for check the stability of our machine learning model. We need some kind of assurance that the model has got most of the patterns from the data correct, and it is not picking up too much on the noise, or in other words, low on bias and variance.
   For 10-fold cross validation in this lab, we first randomly rearrange the sequence in raw attribute set and label set as examples in the raw set are arranged by class they belong to and we cannot directly use cross validation on that set, because it will obviously lead to serious biased result. After randomly rearrange those examples, we sequentially pick 460 examples from new sets (attribute and label) in each loop and use them for validation and the rest part for training. After 10 loops, merely 1 example has not been used for validation, which is still acceptable.
5) If randomization is not practiced and raw dataset is directly used for splitting, the result of accuracy will not be at a relatively stable level. Usually there are 2 possibilities exist: algorithm for continuous naïve-Bayes went wrong, or examples in test set are biased and cannot represent the real possibility that a spam mail appears. Since the classifier works well with continuous naïve-Bayes, we consider the latter circumstance. The cause of fluctuation is lack of randomization, for example, in the last loop of unrandomized cross-validation, we will pick all test examples with label 0, which obviously cannot represent the real case. Preventing occurrence of these kind of difference between classification set and test set is exactly what cross-validation should do.



1.accuracy without randomization



2.accuracy with randomization

Part1:

Since there are 3 datasets need to be dealt with a single set of discrete-valued naïve Bayes functions, size of parameters like val_type and lbl_type should be changeable, leading to the variation of size of likelihood table, vector of possibility for each class.

To build up a discrete-valued naïve Bayes classifier, first I figured out number of types of attribute value that may appear in attribute set (of course also in test attribute set), naming it val_type, and number of types of label value in label set with name lbl_type. Possibilities that an example belongs to each label/class is also calculated by counting number of examples with different label first and divide the result by length of label set. After that, the raw result table is set up with the idea in question 2). The whole attribute set is went through first to get the number of examples that belongs to each value-label combination, coordinate of each result in this newly set up table is also explained in question 2). Then for every result, it shall be divided by the total number of examples of the label this result belongs to. Now that we have a table in which likelihood of every value-label combination is recorded and we notice that some of them are 0, which are true by themselves yet not expected when applying MAP rule as 0 multiplies any number is 0 and that is a serious interruption to the result. So we need to refine the table by m-estimate method. M is set as 1 and p is 1/val_type, by applying m-estimate rule to those features that has possibility of 0, we get a new result likelihood table and get rid of 0-possibility problem. Parameters needed to be sent back to main function are likelihood table, possibility vector and number of value types.

To apply the classifier to test attribute set, MAP rule is needed. We need a new matrix to save MAP results under every label for each example, so its size should be (length of valid Label) * lbl_type, where lbl_type is represented by length of possibility vector from previous function. With this new matrix, we can go through the likelihood table for every example in the test attribute set and get the possibility that an example belongs to a specific class. For each example, we can then compare those results, find the largest possibility that represents this example is most likely to belong to the label and classify the example to the results finally, thus getting a vector that save those classification results through discrimination. The elements in this vector is compared to vector valid Label at last and differences will be recorded to get a final accuracy of our generated classifier. Confusion matrix can also be given by statistics on combination of these two vectors. Accuracy is the result that need to be sent back to main function.

Part 2:

Under circumstance given by the lab, we can assume that both continuous naïve Bayes and spambase task have examples that belong to merely 2 classes.

To build up a continuous-valued naïve Bayes classifier, first number of examples belong to each class is count and possibility that an example belongs to a class is calculated. Attribute set is then spilt into 2 parts, having only examples with label 0 and label 1 respectively, finally mean and standard deviation for every feature is calculated on these 2 steps and stored by 2 2-by-57 matrices.

For discrimination of test examples, a check is run on every element in standard deviation matrix to see if any is 0. Since standard deviation cannot be 0 when calculating possibility, every 0 will be reset as 1. Again, a matrix with size (length of valid Label) * 2 is needed for possibilities that an example belongs to each class. MAP rule is then used on every example across mean and standard deviation tables to get results mentioned above, and likelihoods that were stored in a matrix in discrete-valued case should be calculated by mean and standard deviation this time. A vector with the same length is needed for classification result given by MAP, and the vector is to be compared with valid Label to check differences and get a final accuracy of our generalized classifier. Confusion matrix can also be given by statistics on combination of these two vectors. Accuracy is the result that need to be sent back to main function.

Since attributes and labels are stored in one table at first and 10-fold cross validation needs to be performed, initial dataset should to be split into parts. In my code, labels and attributes are separated first and randomization is then practiced. After producing 10 sets for classification and testing (procedure and data structure is explained in answer to question 4), every matrix for generating a classifier is reformed to get examples in them gather by their labels. Attribute set, label set, test attribute set as well as valid label is then sent to main function.

For each loop of 10-fold cross validation, a group of datasets out of 10 is sent to training function for continuous naïve-Bayes and their results to previous discriminator. Accuracy from 10 loops are all recorded and their mean and standard deviation is given at last.