Slippery Rock University

Naive Bayes Classifier Backend and Frontend Technical Manual

Trevor Hamilton (tjh1003) & Jonathan Stonebreaker(jds1018)

Data Mining - CPSC 405

Dr. Sam Thangiah

Table of Contents

3..........................................................................Backend Main Class and Standard Program Flow

8................................................................................................Expected Input Data File Formating

9...................................................................................................................Output Data File Format

10.................................................................................Backend Functions Explanations and Usage

10.................................................................................................................................................GUI

11.............................................................................................................................Additional Notes

Section 1: Backend Main Class and Standard Program Flow

Main Class Breakdown

* Breakdown of the output and function order of the main class with descriptions of what nonprinting function class do in the next section
  + Strings for the input and output excel files to be used
  + Do the import of the input excel file
  + Generate the training data
  + Notify use training data has been generated and print both the training and testing data
  + Generate the model (Likelihood of all the possible values of each attribute)
  + Inform use model has been generated and print the possible classifications and the model
  + Classify everything in the testing data
  + Inform user the guessed classifications have been generated and print of the guessed and acutal classifications
  + Export results to output excel file
* Output symbols mean the following
  + ::
    - Label was printed and this group of data listed belongs to it
  + :
    - Next data point for this grouping
  + ,
    - New group of data
  + [
    - Start of new dimension of data
  + ]
    - end of dimension of data

Standard Order of Function Execution

* The order of the function calls to import a file, build a model, classify new data based on that model, and export results. Also included is some tips about each one
  + readExcelFile(intputFileName);
    - Should be formatted as specified in section 2
  + generateTrainingDataGENERATOR(trainingDataSize);
    - Multiple different ones a brief explanation of each is as follows
      * Random
        + Randomly selects a remaining data point to remove from test and add to training while trainingDataLL.size < trainingDataSize
      * First
        + Removes the first trainingDataSize from testing data to add to training data
      * Stride
        + Takes every dataLL.size/trainingDataSizeth data point from testing data to add to training data. Example end result you have 15 data points and you want a training size of 5 15/5=3 you will get every 3rd data point in your training data
      * FromFile
        + Load an excel file directly into training data. Does not remove duplicates expects this file to be formated the same as original imported file except does not require metadata worksheet as it uses the one from the original file
  + generateClassifier();
    - Generates the model that will be used to make guesses on future data’s classification
      * Only a naive bayes classifier is implemented a gaussian bayes classifier was hoped to be implemented but that time was spent implementing laplacian smoothing
        + naive bayes algorithm put in simple terms is for each possible classification you for each attribute add up the number of times that value and classification occurred and divide that by the number of times that classification occurred, you then for the classified take the number of times that classification occurred divided by the size of your data set. This generates a likelihood of each value for each classification which will be used in generateClassifications() latter (If smoothing is turned on it will add the laplacian smoother value described below) (Good explanation video done by 5 Minutes with Ingo can be found [here](https://www.youtube.com/watch?v=IlVINQDk4o8))
      * Function is entirely ready to have gaussian implemented you would just need to write a gaussian classifier and have a way to select between the two functions
      * Uses a laplacian smoother to help with unseen values in an attribute if laplacian smoother is greater than 0
        + laplacian smoothing in simple terms is adding the constant k/(k\*n) to each distinct attribute values likelihood where k is the smoothing value (normally 1) and n is the number of distinct values for that attribute (also applied to classifier likelihood and n is number of distinct classifications)
        + This prevent multiplying a 0 into your likelihood and have a strongly classified value now equalling 0 because a value showed up that was unknown
  + generateClassifications();
    - Uses likelihoods generated before to guess what the classification is for each data point in the test data
    - calls classify(testDataLL(i)) for each data point in the test data. If you wanted to you could call this function manually with hand crafted string arrays to get a classification
      * classify for each classification possible multiplies the likelihood of each value of the string array for the classification together and then multiplies it by the current classification storing it in array of the size of possible classifications. It then goes back through and whatever one is the largest is the guessed classification. Gets what the string value is for that classification number and returns that.
        + Note is smoothing is on there should be no 0s occuring
  + outputExcelFile(outputFileName);
    - Outputs the results to the specified location in a nicely formatted excel file
      * Sheet 1 contains all the results with the classifier highlighted in yellow and the guessed classification highlighted in green or red if it was right or not. At the very bottom the percent right is displayed with it being green if it is above 90% and red if it is below 90%
      * Sheet 2 contains the training data in the same format as the original excel file
      * Sheet 3 contains the likelihood value for each of each attributes distinct values and then the likelihood of each classification at the end
      * All sheets are labeled at the top and sheet 3 is labeled on the lefthand side

Section 2: Expected Input Data File Formating

Sheet 1 Breakdown

* Contains a list of all the data going to be used
  + Each column is an attribute
  + Each row is one data point

Sheet 2 Breakdown

* Must be listed in the same attribute order as the first sheet
  + Row 1
    - Contains a label for each attribute that you would like it the be referred to
  + Row 2
    - Contains if the data is discrete or continuous, unused but must contain a value
  + Row 3
    - Contains the word classify if this row is the classifier, can be any value but all other cells must be blank

Section 3: Output Data File Format

Sheet 1 Breakdown

* Contains the testing data in the same format as input file
* Contains the label of each attribute along the top
* Contains the guessed classification on the right end
* Contains the percent right at the bottom right
  + Actual classification in yellow
  + Guessed classification in green if it matched the actual red if it didn't
  + Percent is green if above 90% else it is red

Sheet 2 Breakdown

* Contains the training data in the same format as input file
* Contains the label of each attribute along the top

Sheet 3 Breakdown

* Contains the likelihood of each distinct occurrence in the training data for all known classifications
* Contains label of each attribute along the top
* Contains each distinct possible classification along the left
* Contains the likelihood of each possible classification on the bottom

Section 4: Backend Functions Explanations and Usage

Section 5: GUI

Breakdown

* GUI was not fully fleshed out but as it currently stands is as follows
  + Tab 1
    - A tab to choose a data file to import
    - Browse opens a file browser
  + Tab 2
    - Unimplemented tab that would be used to display each attribute in the data and allow user to rename choose data type and select if it is the classifier
  + Tab 3
    - Tab to export data, currently does the classifications on export to prevent exporting file before finished
  + Intended workflow
    - Select a file to import, click import and the selected file is imported, once finished tab changes to metadata tab
      * If copy to data is selected also store a copy in data
    - See if meta data imported correctly and be able to change the labels, data types, and which attribute is the classifier, click classify and start classification updating user how far along it is, once finished tab changes to export tab
    - Chose a location to export the results to, click export and the file is exported, once done open the file
      * If copy to results is selected also store a copy in results

Section 6: Additional Notes

Common Vocab

* Metadata
  + Labels, data type, and isClassificaiton for each attribute
* Training Data
  + Data to be used to generate the classifier
* Testing Data
  + Data with unknown classifications that the classifier is used to guess the classification
* Known Classifications
  + Classifications that belong to the training data, always known
* Guessed Classifications
  + Classifications that the classifier generated without seeing the actual classifications
* Actual Classifications
  + Classifications that belong to the test data, unknown until compring if classifications are right
* Laplacian Smoothing
  + A constant added to the likelihood to prevent likelihood from being 0, mathematical explanation above
* Gaussian Bayes Classifier
  + Bayes classifier used for continuous values, unimplemented, but the program is setup to allow it if implemented
* Naive Bayes Classifier
  + Classifier used for discrete values, used for a values in this program since gaussian is unimplemented, mathematical explanation above
* Classification
  + What class a data point belongs to
* Frequency/Occurrences
  + Count of the number of times that value occurred for each possible classification
* Likelihood
  + Frequency divided by the number of times that frequency’s classification occurred (May be uncaught times that likelihood is called frequency is in the code due to originally on the frequency was stored but I think I got them all)
* Model/Classifier
  + What the testing data is passed through to guess the classification mathematically defined above
* Distinct/Seen/Known/Possible value(s)
  + Go through attribute count the number of different values listed, each of those is a distinct/seen/known/possible value. Used each of these based on what made more sense for the data being used at the time
    - Note: these counts only come from the training data unseen values end up just getting the value of smoothing constant