Marvin Der Hann Lai
754672

The Task

The task given was to write a language classifier. The classifier is given a set of training data where each data point (or line in this case) was classified into one of 20 languages. It then has to build a model which can be used to classify future test instances into the correct language. There is one caveat in that the classifier must be able to classify a given line as "unknown" if the line is in a language that it has not come across in the training data.

There was not one particular algorithm to use, meaning any suitable method of accurately classifying a line of data into the correct language can be used. In the time given for the assignment, two methods were devised in an attempt to classify text into languages. Method one, and the more successful method was a "word based" method, which consistently achieved an accuracy of around 73%. Method two was a "character based" method, which consistently achieved an accuracy of only around 55%. These methods will be explained and analysed for their shortcomings and successes later while evaluating the classifiers.

Focal Points

The primary focus and goal of the task was to create a classifier that was able to obtain a satisfactory level of accuracy for a wide variety of languages, meaning being able to consistently classify all languages with at the very least majority accuracy. The secondary focus was to use at least two classification methods to classify the language of the given instances.

The extent to which the goal was achieved will be discussed further on.

Details

Two methods were trialled over the duration of the assignment, method one is the "word based" method, and the second is the "character based" method.

For method one, each line was broken down into words (words in the English sense of groups of characters separated by a space – this is a flawed assumption), cast to lower case, and stripped of punctuation in an attempt to standardise words. For example, "hi" and "Hi!" Are not treated as different words, and both will be represented as just "hi". Only one instance of every word is stored. Each language has its own "bag" of such words, based off the words encountered in the training data; essentially it creates a dictionary to look up words in the test data.

In the classification stage, it takes a line and for each language in the training data, counts the number of words common in both the given line and the language's dictionary. The classified

language for that line is then the language with the most unique matches. Note that this only counts the unique matches for every word in a line.

For method two, each line was broken down first into words following the same process as above, and then a count of every letter that occurs after this pre-processing for each language is stored. Each language then has a "vector" of the count of each letter in the training data.
In the classification stage, it takes a line and for each language, counts how many letters the line and the language has in common and stores the values in a vector, and calculates the cosine similarity between the two vectors.
The classified language for that line is then the language with the largest cosine similarity.

Evaluating the classifiers

To the extent of achieving the goal, neither classifier does a particularly good job on its own, as evidenced by the "bag of word" classifier's abysmal ability to classify languages such as Japanese and Thai, with an accuracy of 0% and 3% respectively, as the assumption that a "word" is a bunch of characters separated by a space doesn't hold up for languages such as those. However, for other languages where that assumption does hold, it works well, achieving accuracies in the range of 70%-90% accuracy consistently.
The last downside is a relatively poor ability to identify unknown languages, as the default requirement to be classified into a language is for a single word to match. Through some testing, that requirement was changed to be a match of 3 words. Doing so resulted in a much better identification of unknown languages; resulting in an increase of 8% to 32% accuracy. As the number suggests, it is entirely possible for an "unknown" language to contain words that are present in other languages. However, by increasing the minimum requirement, any hopes of accurately identifying non word based languages is lost, as the slim chance that there were repeated words is ruined given that one line has to have at least 3 words match, hence the non-existent accuracy of Japanese.
The performance of this method varies depending on both the training and test data given to it, it heavily favours training data with wide a wide variety of language-specific words, coupled with test data with the same properties.

The second method picks up where the first one fails, as it works on a character by character basis, it is much better at detecting character based languages than word based languages.
As evidenced by the improvement in the identification of Japanese and Thai, with 87% and 93% accuracy respectively. This is simply down to the fact that many languages have different

alphabets/character sets, so naturally input of the same alphabet will score a relatively high cosine similarity to its appropriate language.
For languages that share a similar alphabet, this classifier does not successfully classify data as well as the word based classifier, as the uniqueness of a language's character composition is only apparent 50% to 70% of the time.
However, there wasn't found to be a lower limit cut off for the cosine similarity which improved the accuracy or allowed for the identification of unknown languages.
The accuracy for this method is much more dependent on the data being long enough to find the pronounced difference between different language's character compositions.

Final thoughts

Each language classifier had its strength and weakness. The word based classifier had an easy time correctly identifying languages by matching words to the dictionary built up in the training data, struggles mildly in classifying unknown languages, and completely fails at classifying character based languages.
On the other hand, the character based classifier had an easy time identifying character based languages, with 80% to 90% accuracy on such instances.

Ideally, if more time was available, a combination of the two classifiers could be made, picking the strengths of character based and word based language identification. Perhaps one such way would be to identify if a given input has a low space to character ratio, then switch to the character based method for classification.

Finally, a quick summary. The word based classifier is better at identifying languages with structures similar to English, with spaces between words, due to the fact that individual words will be extractable from a sentence and matchable to the dictionary generated in the training phase, as well as identifying unknown languages. The character based classifier is better at identifying languages which have little to no spaces between words, as it analyses a string on a character by character basis. Both classifiers have flaws, but mostly in places where the other succeeds, as such, a combination of the two would be ideal.