Lab 2 CSCI Foundations of Artificial Intelligence

Features Chosen:

1. Attribute 1 : If the word contains ‘ij’, then it is way more likely to be dutch than English
2. Attribute 2: If the sentence uses the common English words ‘to’ and ‘a’, ‘is’ then it’s more likely to be English than dutch
3. Attribute 3 : average word length > 5 then dutch, else English

The first three observations were based solely on my judgement of training data, the next two attributes were calculated by taking a huge dictionary of both English and dutch, and counting the number of words that start with each sentence.

4. Attribute 4 : check for words starting with q, likelier to be English
5. Attribute 5 : check for words starting with g , likelier to be dutch

Attribute 6-10 was chosen from the most popular dutch common words from studies worldwide

6. Attribute 6 : check for dutch common words ['ik', 'je', 'het', 'de', 'dat', 'een','zijn']
7. Attribute 7 : check for e, more likely to be in dutch
8. Attribute 8 : check for s, more likely to be in english
9. Attribute 9 : check for the word "the" , more likely to be English
10. Attribute 10 : check for double vowels, more likely for dutch

Lab 2 CSCI Foundations of Artificial Intelligence

Decision tree learning

Decision tree learning employs the use of Entropy and Information Gain to select the best attribute at each height which is the best classifier to separate the two separate goal nodes. The algorithm was developed in closed conjuction by referring Russell and Norvig Pg. 703.

The best parameters for me turned out to be a training data which consisted of English or dutch languages, but also contain a reference to a name, place or an item which is associated with the other language. That enabled me to get a good mix of nodes for my decision tree. There was not much gain by going for nodes beyond a depth of 5, so I will be taking that as my optimum depth of the decision tree, after which I just take the plurality value in my decision tree.

My own testing results were on three .data files consisting of sentences with 10 words in one, 20 words in 1 and 50 words in the last one. And I test my output after training these same files. For 10 and 20 sentences, we are able to get fairly accurate results, as I got all 10 sentences that were given in the problem definition by Prof. Niyazi sir correct. Then I had two custom test results, one with 20 sentences, out of which one was predicticted incorrectly, "*The main bibliographic sources areVan Schoonvorst, De Gavre en een bastaardtak van de familie Van Arberg. In 1794 arriveerden de Fransen, waarmee een einde*".

This sentence while being in English, uses way more Dutch words and terms, and was predicted to be Dutch for the same reason. Thus yielding a 95% accuracy rate.

For 50 words, the algorithm turns out to be a bit more complex, more and more attributes are contentious for the best nodes at each stage, and I made use of a depth of 8 to be able to better classify in this case. Even this program was not able to correctly classify the sentence mentioned above however.

Lab 2 CSCI Foundations of Artificial Intelligence

Adaboost learning

Adaboost algorithm makes use of weak classifiers, and turning them into a strong classifier by attaching weights and giving each “stump” or weak classifier a say. The algorithm was developed in closed conjuction by referring Russell and Norvig Pg. 751.

A decision tree of depth 1 is considered a stump and is in itself a weak classifier. Our Adaboost algorithm makes use of these stumps, generates hypothesis weight for each hypotheses generated at each stump. These hypotheses weights are ultimately useful in the prediction of the algorithm.

The best parameters for me turned out to be a training data which consisted of English or dutch languages, but also contain a reference to a name, place or an item which is associated with the other language. That enabled me to get a good mix of nodes for my decision tree. I used a stump of 5 as the hypotheses weights for any more stumps were going as low as 10^{-14} and thus efficiently making it’s say not count whatsoever in the final decision.

On three .data files consisting of sentences with 10 words in one, 20 words in 1 and 50 words in the last one. And I test my output after training these same files. For 10 and 20 sentences, we are able to get fairly accurate results, as I got all 10 sentences that were given in the problem definition correctly. Again, I got the same sentence that was mentioned above incorrect. This sentence while being in English, uses way more Dutch words and terms, and was predicted to be Dutch for the same reason. Thus, yielding a 95% accuracy rate.

Once again for 50 words, the algorithm makes way less predictions with most attributes and so, I made use of a stump of 6 to be able to better classify in this case. This program wasn’t able to correctly classify the sentence above either