Explanation of Intro to IS Lab 2: Wikipedia Language Classification

Feature (Attribute) And Evaluation:

Based on the natural feature of English and Dutch, several words with relatively high occurrence were chosen as the features to recognize the language type. Most of the words are preposition, conjunction, etc. In general, 10 features were implemented in feature.py file, and all of them will return True if the sentence meets the feature, False otherwise. They are:

1) If the sentence has the word “een” (means “a/an” in English)

2) If the sentence has the word “de” (means “the” in English)

3) If the sentence has the word “bij” (means “with” in English)

4) If the sentence has the word “van” (means “from” in English)

5) If the sentence has any word from “maar, en, als, dan” (a list of conjunctions in Dutch)

6) If the sentence has any word from “ik, jij, u, gij, hij, wij, het” (a list of personal pronouns in Dutch)

7) If the sentence has any word from “this, that, there, which, where, who, whose, when”

8) If the sentence has any word from “in, on, at, to, for, by, of, with, and, or”

9) If the sentence has any word from “I, you, he, she, it, we, they, him, her, us, them, the, a, an”

10) If the sentence has any word from “am, is, are, was, were, being, been, be”

Noted that they are implemented in order and the first feature value is 0 (the last is 9).

With the decision tree, a training set of 1000 sentences and a testing set of 100 sentences were used and the accuracy was 100%. With the ada boosting, the same procedure gives the accuracy of 95%.

The default max tree depth is set as 10 since there are total 10 different features. When the training set size was small, the depth is not a significant effect since the decision tree does not tend to reach very deep. However, when a bigger training set was used, some branches in the decision tree may approach the limit of 10. In fact, a very deep branch does not affect the entire algorithm very much because most of samples can be neatly recognized in a much shallower branch so only few cases will reach the deep branch. In addition, most of these samples that go into a deep branch is because the current features cannot determine them, so it still does not affect the entire learning accuracy very much. In fact, the accuracy starts to dropped only when the tree depth was decreased to 5 (w/ a training set of 1000 sentences).

The default hypothesis of ada boosting is directly taking from the previous 10 features, they can give decent weight and good testing accuracy. Although a list of combined hypotheses (all determination from inducing) from each leaf node was evaluated, it turned out that it did not

affect the accuracy but weight becomes insignificant. It is probably because that the previous 10 features and the decision tree has already recognized each example very well.

Execute Instruction

To run the project, the main function is in language.py file. It has 3 different modes.

1) train (following three arguments:) <examples> <hypothesisOut> <learning-type>. <examples> is the sample input file, <hypothesisOut> is the learning object output file. <learning-type> is the desired learning type, “dt” refers to decision tree, “ada” refers to ada boosting. P.S. the max tree depth can be modified at the front of language.py file.

2) predict (following two arguments:) <hypothesis> <file>. <hypothesis> is the learning object input file. <file> is the testing example input file. It will print out the prediction in command line.

3) help. Display the help message.

The example files in the folder with size of 10, 100, 1000, 2000, 5000, and 10000 sentences are given for training. The example file in the folder with size of 10 and 100 (test1.dat and test100.dat, respectively) sentences are given for testing.