Introduction to Artificial Intelligence
Lab 2 Write-Up

Derrick Kempster
Jansen Orfan

For this project, I wrote a multifunctional program (lab2.py) to be able to determine if a line of text comes from an English or Dutch Wikipedia page. This program has two different usages based on the first argument passed into the program:

```
python3 lab2.py train <examples> <hypothesis> <learning-type>

python3 lab2.py predict <hypothesis> <file>
```

The first usage creates a hypothesis based on training data; the second predicts the language of several lines of text. The `examples` argument is the name of a file holding lines of text in English or Dutch, each prepended by a language indicator. The `hypothesis` argument is the name of the file that holds the hypothesis, which is created or overwritten in the first usage and read from in the second. The `learning-type` argument must be either `dt` or `ada`, to either create a decision tree or adaboost hypothesis about the example data, respectively. The `file` argument is the name of a file holding lines of text in English or Dutch to evaluate.

When looking at files containing lines of English or Dutch text, each line is analyzed for features. There are 26 features, for the frequencies each possible letter. Table 1 shows the frequencies of letters in English and Dutch. The letters in the Dutch alphabet with diacritical markings or umlauts are treated as if they lack them. Also, IJ is considered one letter in Dutch, but often typed as the two separate characters I and J. To reflect all of these differences in interpretation for this program, recalculated frequencies are shown in the Reduced Dutch Frequency column of Table 1. Then, the average between the English frequency and Dutch frequency of a letter is used as a threshold for determining language difference.

The function to build the decision tree runs recursively. Examples fall into nodes of the tree based on the truth values of their features. At the current node being checked, if all examples are of the same language, it is marked for that language. Otherwise, if the entropy for every feature is determined, and the node is marked for the feature of lowest entropy. The examples in this node are passed onto child nodes based on their truth value for that particular feature. If a node is at the maximum depth of the tree, it is assigned a language based on the languages of the examples contained within it. When the tree is stored in the hypothesis file, a similar method is used to store binary trees in arrays. Each of the nodes can be represented by a single character, by using a letter for nodes that split on a feature, '+' for a node that is distinctly English, and '-' for a node that is distinctly Dutch. The character '.' is used for any nodes present in the binary tree that are not in the decision tree. Since one character can be used for every node in this binary tree, and the equivalent array, it can simply be written out to the file in order. The file position of the character representing a node can be used to locate the node's children.

When using the adaboost implementation, stumps can be generated for all features and possible output child node languages. In this program, all 104 possible stumps are used. If less than all stumps for one feature are used, the feature may be misrepresented in the final calculation. If less than all features are used, then some important features may be ignored. Without either of these cases, the calculation may be incorrect for a complex formulation like this, so they all can be used relevantly. The hypothesis file is populated with the all constituent hypothesis values. To distinguish this type of hypothesis file from that of decision trees, the delimiter between values is also present at the start of the file. This delimiter is distinct from characters used to represent tree nodes.

Table 1: Letter frequency data. Reduced Dutch Frequency shows letter frequency when using English letters.

Letter	English Frequency*	Dutch Frequency*	Reduced Dutch Frequency	English – Reduced Dutch Average
A	0.0834	0.0776	0.0769	0.08015
Ä	-	0.0003	-	-
B	0.0154	0.0138	0.0140	0.01470
C	0.0273	0.0131	0.0129	0.02010
D	0.0414	0.0548	0.0541	0.04775
E	0.1260	0.1931	0.1910	0.15850
Ë	-	0.0003	-	-
É	-	0.0001	-	-
È	-	0.0001	-	-
F	0.0203	0.0074	0.0073	0.01380
G	0.0192	0.0317	0.0313	0.02525
H	0.0611	0.0316	0.0312	0.04615
I	0.0671	0.0503	0.0630	0.06505
Ï	-	0.0001	-	-
IJ	-	0.0134	-	-
J	0.0023	0.0051	0.0183	0.01030
K	0.0087	0.0283	0.0279	0.01830
L	0.0424	0.0385	0.0380	0.04020
M	0.0253	0.0260	0.0257	0.02550
N	0.0680	0.1004	0.0991	0.08355
O	0.0770	0.0588	0.0581	0.06755
Ö	-	0.0001	-	-
P	0.0166	0.0151	0.0149	0.01575
Q	0.0009	0.0001	0.0001	0.00050
R	0.0568	0.0570	0.0563	0.05655
S	0.0611	0.0391	0.0386	0.04985
T	0.0937	0.0650	0.0641	0.07890
U	0.0285	0.0213	0.0212	0.02485
Ü	-	0.0002	-	-
V	0.0106	0.0227	0.0224	0.01650
W	0.0234	0.0174	0.0172	0.02030
X	0.0020	0.0005	0.0005	0.00125
Y	0.0204	0.0006	0.0006	0.01050
Z	0.0006	0.0162	0.0160	0.00830

* = sourced from www.sttmedia.com