**MP2 Report**

Sarah Darwiche

CS165a

**Architecture**

My code consists of three main classes. A data class, attribute class, and decision tree class. The data class takes care of processing the example information by reading from file and filling in the data structure that will essentially construct the decision tree. The attribute class handles the data of each attribute, including its name, values it can take and computes information gain for each attribute. Decision tree constructs the decision tree using the corresponding data and also contains a classification function. Main takes care of timing and setting up files for the other functions to parse.

**Preprocessing**

In order to make parsing easier I had several stages of file reading. My first step transform the train and test files to contain the important relevant information that will be used to fill in their corresponding data structures. This includes example count, attribute count, the number of values each attribute takes and what they represent, the number of values the label takes and what the represent, and finally the actual examples.

**Model Building**

The decision tree pruning I have implemented is pulled directly from the text, page 705. If a test node has only leaves as descendants, and the test appears to be irrelevant we eliminate it replacing it with a leaf node. Information gain helps us calculate relevance, and a significance test tells us whether to reject the null hypothesis (attribute is relevant) or fail to reject the null hypothesis (attribute is irrelevant).

**Results**

My final results based on the given test files are 96 percent training accuracy, 78 percent test accuracy with a tree build in less than a second and the test file classified on .002 of a second. The pruning performance was minimal to the algorithm as a whole as it made almost no difference. The features I choose were based on words likely to appear in movie reviews. They were subjectively chosen and stored into a fixed list.

**Challenges**

One major challenge that I faced occurred in the process of computing information gain. There was a mathematical error occurring when computing entropy – remainder. In order to resolve this, I realized that entropy should be the same for every attribute, so it was not necessary to have that as part of the calculation. The solution was to leave that out of the calculation and instead of maximizing information gain, I minimized it.

**Weaknesses**

One of the weaknesses my program has is the list of attributes is fixed and not determined by the training data. Depending on the scale and thoroughness of the data this could result in worse performance.