# Data

The original data comes from here:

<http://ai.stanford.edu/~amaas/data/sentiment/> - [Large Movie Review Dataset v1.0](http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz)

It contains the score of words, training data and test data. Score of words is provided within two files: imdb.vocab for all the words and imdbEr.txt for all the corresponding scores under the folder root.

For the training data folder, it has 12500 negative rated comments and 12500 positive rated comments. For the test data folder, it also has 12500 negative rated comments and 12500 positive rated comments. That is to say, we have 50000 labeled data in all.

Since we did our own part separately, the data interface we shared is csv file format instead of list. I understand that we can combine all 50000 comments into one large dataset, and do cross validation on it, but due to the argument we use (csv file name), it’s hard to apply cross validation on our handmade Decision Tree and Naive Bayes classifier. As a balanced result, we apply cross validation on the existing sklearn classifier.

# Data preprocessing

As mentioned above, we have a list of word scores, and we have also 50000 labeled comment files. What we did first was to preprocess the word scores into categorical label: 0, 1, …, 9 representing sentiment level from strong negative to strong positive, and then save to a dictionary for further use.

After this, we tokenize each comments into a list of words. We filtered the stop words and the punctuations simply using regular expression tokenizer. I admitted that the way of tokenizing is too simple to get a good classification score. Therefore, we can have more complex ways to tokenize the comments, as mentioned on class, adding negation filtering in the future.

By mapping these word categories dictionary, we generated a list of sentiment word counts, which has 10 columns representing how many words are under those 10 categorical labels. We use these data to test the Logistic Regression and SVM classifier.

In order to fit the data into our Decision Tree and Naive Bayes classifier, we change the previous data into binary one. It surely lost lots of information but it is the only way we figured out currently.

# Comparison

Now we only use one fold (given training data for training and given test data for testing) to fit all the classifiers. If we have enough time, we will fix the data interface, and try k-fold cross validation on the whole data.