# Sentence and Phrase Sentiment Analysis

## Data Source of Kaggle Movie Sentiment Analysis

Overview:

Textual sentiment analysis is of growing importance for many commercial and social uses. For example, review aggregation sites (e.g., Metacritic) wish to automate the aggregate scoring of reviews from critical or user sources; research and news sites (e.g., NYT Upshot) might wish use analysis to examine twitter commentary to judge political feeling of a populace; or manufacturers might wish to examine user comments on store sites or reviews on consumer sites to see what people do and don’t like about their products. Therefore, the ability to examine texts and accurately derive their emotional or valuative meaning has great consequence across a wide variety of disciplines. Due to the value of sentiment analysis, therefore, this project will attempt classification of both full and partial sentences into a ranking from 0 to 4: strongly negative to strongly positive.

Data Source:

The data comes from a Kaggle competition at this link:

[Sentiment Analysis on Movie Reviews](https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews)

The source file has over 8500 complete sentences from the movie review aggregation website “Rotten Tomatoes”. This file has further been chunked into over 150,000 lines which consist of anything from single words to the complete sentence, as well as each line having a pre-annotated score classification from 0 (being very negative) to 4 (being very positive). Each line also contains a sentence identifier so one can see the sentences where the chunks come from, as well as an incremental key for each line. In detail the line looks like this with made up examples:

PhraseID, SentenceId, Phrase, Sentiment

1 1 This sentence is not positive or negative. 2

2 1 This sentence is not positive 1

As we see PhraseID 2 is a subset of PhraseID 1, so they both share the SentenceId: 1. The first phrase is neutral granting a sentiment score of 2; the second sentence is negative thereby receiving a score of 1. If the second phrase perhaps had been something like: “This sentence is horrible.” Then the score would be 0.

Goals:

The goals of this project are to experiment with a number of techniques in order to, not surprisingly, create a model that as accurately as possible classifies each line with the sentiment that has been pre-identified in the file. Techniques and libraries that will be used consist of the following possibilities:

* Various NLTK Libraries used to parse, chunk and classify text: including examples like:
  + nltk.sentiment package, which is a module that helps identify sentiment of text
  + nltk.classify package which contains various algorithms and methods to “classify” text, such as max entropy.
  + nltk.stem package which “stems” words so that we get the root of a word, rather than a variation of a word (e.g., happy is the stem for happily, happiness, happier, etc.).
* SKLearn Classifiers, for example, random forest or naïve bayes. The classifiers here are similar to the ones offered in the NLTK libraries, but depending upon how the research develops, one or both of classifier libraries (NLTK or SKLearn) may be used.

Do note we said “possibilities” in regards to what libraries or techniques. This is because we both do not wish to limit the research to these if an alternate interesting or more accurate technique for this data returns better results, nor do we wish to suggest that if one or more of these techniques does not provide good results, that we may eliminate them from the work presented.

Results:

Results will be presented in a Jupyter notebook with commentary through explaining the steps. Data will be divided into training and test sets, and the various techniques accuracy (including recall and precision) will be shown.