# SH1: Sentiment Analysis

August 30, 2018; due September 6, 2018 (11:59pm)

In this homework, you will write code that generates features for the positive/negative sentiment classification of movie reviews. Example of reviews with labels:

| Label | Review |
|---|---|
| pos | GREAT movie and the family will love it ! ! If kids are bored one day just pop the tape in and you 'll be so glad you did!!! Rube i luv raven-s ! |
| neg | I was duped into seeing this movie after reading a positive review from another website and man was I p.o'd!!! it took me at least 15 minutes to pick it up off the shelf b/c I didn't want anyone to see me . then another 10 minutes to build the courage to take it to the counter and actually use real money to rent it . I thought that all my stress would pay off by the time I got home to and watching the movie b / c the review I read said the movie was a pleasant surprise ; what a joke ! if you can make it through the first hour of the movie then your in luck! b / c it 's not until then the movie turn 's into a horror. don't bother with this one folks , your better off watching " dankness falls " |

You are given the following files:

- `sentiment_classifier.py` A file with starter code
- `train.txt` : Training set to train your model
- `dev.txt` : Development set to tune parameters and evaluate your model's performance
- `test.txt` : Test set to report your model's performance

The starter file does all the data parsing, model training, and evaluation for you. During training, a basic logistic regression model (from scikit-learn package) is trained with your features to predict whether the sentiment of a movie review is positive or negative. Your ultimate goal for this assignment is to featurize the text and optimize the accuracy of the model on the test data in test.txt.

## 1 Bag of Words

a.) The sentiment classifier in `sentiment_classifier.py` currently uses some "dumb" features to make predictions. We are going to get a little fancier and implement a basic bag of words featurization of the text. Follow the instructions below:

1.) Implement the `bag_of_words` function in `sentiment_classifier.py`, so that it adds the bag of words representation of `text` to the `feats` dictionary. (SPL Chp.4)

2.) Generate features, train and evaluate the model by running:

```
python3 sentiment_classifier.py --train train.txt --dev dev.txt --test test.txt
```

Running this command will generate the following files:

- `dumb_featurize_predictions.csv` : predictions for test.txt with dumb features
- `dumb_featurize_weights.txt` : weights for model trained on dumb features
- `fancy_featurize_predictions.csv` : predictions for test.txt with fancy features
- `fancy_featurize_weights.txt` : weights for model trained on fancy features

b.) Look at the ouput in `fancy_featurize_weights.txt`. What does the sign of the coefficient mean?

c.) Pick a word with a large positive coefficient and a large negative coefficient. Why do you think these words have these coefficients? Cite examples from the training data.

## 2 Feature Improvement

Now we are going to get fancier with our features and compete with our peers.

a.) Come up with and implement a minimum of 3 different featurizations for sentiment analysis. Describe your features in your write-up and include the accuracy on the `dev.txt` data (printed to the terminal).

Points will be awarded for creativity (defined as implementing three sensible, different features that few others thought of). You are allowed to use other data (e.g., sentiment dictionaries, pre-trained word vectors), but you are not allowed to import other libraries (e.g. NLTK). For feature ideas look at SPL Chp.4 and lectures from the week of 8/28-8/30.

b.) Using the best model you trained, generate predictions for `test.txt`. Upload your predictions to the Kaggle leader-board. For your Kaggle submission, you may optionally add more features. Aside from the featurization function, the only other free parameters you should manipulate are the feature minimum count and the regularization strength. In your report, include your Kaggle name as it displays on the leader-board, your highest Kaggle score, and a description of what you did.

Points will be awarded for reaching the baseline accuracy on Kaggle and for absolute accuracy (being in the top 30% of the Kaggle leader board). Extra credit will be awarded to the top 5 scorers on Kaggle.

Start early! Kaggle only permits two submissions per leader-board per day.

Kaggle URL: `https://www.kaggle.com/t/e52cc66f233546bb8c9d80bc33a3533b`

## 3 Deliverables

Submit to GradeScope Homework 1 writeup: `sh1_writeup.pdf`
Submit to GradeScope Homework 1 code: `sentiment_classifier.py`, if you used other data submit a zipped folder with your code and any other data used.
Submit to Kaggle: `fancy_featurize_predictions.csv`