# 1. Sentiment Analysis via Lexicon-Based Approach

## Preamble

Download the “Product Sentiment” dataset: *sentiment_train.csv* and *sentiment_test.csv.*

Note: this dataset is originally from
http://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences

## Tasks

1. [Code] Perform sentiment analysis on the dataset using the lexicon-based approach. If using R, using the qdap library. If using Python, use the nltk.sentiment.vader library.
    a. Load, clean, and preprocess the data as you find necessary.
    b. Use the training data to tune the hyperparameters of the function.
        i. In qdap, there are four hyperparameters: *question.weight*, *amplifier.weight*, *n.before*, and *n.after*.
        ii. In vader, there are no hyperparameter to tune ☺
    c. Use the testing data to measure the accuracy and F1-score of your model.
2. [Text] Given the accuracy and F1-score of your model, are you satisfied with the results? Explain.

### *Solution:*

After training the model using *nltk.sentiment.vader* library test set **accuracy score is 0.79304** and **F1 score is 0.8021**. I believe these results are good given the variation in the reviews such as slang, exclamation marks, tone, etc., the model was able to accurately predict the positive and negative reviews. The high F1 score tells us that the classifier was precise (correctly classified many instances) as well as robust (it did not miss a significant number of instances). This level of accuracy and F1 score is possible because Vader can identify *social media type* of sentiment without additional training or pre-processing.

3. [Text] Show five example instances in which your model was incorrect. Describe why the model was wrong.

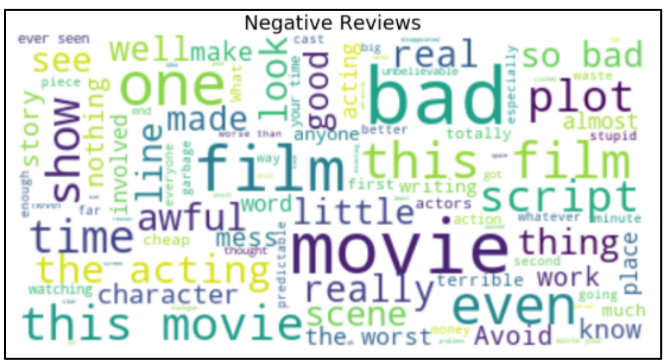
### *Solution:*

Below are the five examples where the model was incorrect.

| **Sentence** | **Actual** | **Predicted** |
|---|---|---|
| I won't say any more - I don't like spoilers, so I don't want to be one, but I believe this film is worth your time. | 1 | 0 |
| Not frightening in the least, and barely comprehensible. | 0 | 1 |
| At no point in the proceedings does it look remotely like America. | 0 | 1 |
| The plot, such as it is, is so derivative and predictable that the ending is like a mercy killing. | 0 | 1 |
| Lifetime does not air it enough, so if anyone knows what store sells it let me know because this is a must-have. | 1 | 0 |

From the examples above it is evident that the reviews that are incorrectly predicted had both positive and negative in them. That is, in the first example terms like “spoilers”, “don’t like”, and “don’t want” are high scored negative words while a word like “worth” is lower scored positive word. Similarly, in example three the word “like” probably has a high score in the lexicon which made the model predict it as positive sentiment.

The below word clouds show these similarities between positive and negative reviews. For example, the word “good” is in both the reviews although generally it is considered high scored positive word in a lexicon. These mix terms in a sentence makes it difficult for the model to capture because the lexicon is pre-built in Vader. We could avoid this inaccuracy by building our own lexicon.

# 2. SENTIMENT ANALYSIS VIA ML-BASED APPROACH

## PREAMBLE

Download the "Product Sentiment" dataset: *sentiment_train.csv* and *sentiment_test.csv.*

## TASKS

1. [Code] Perform sentiment analysis on the dataset using the ML-based approach.
    a. Load, clean, and preprocess the data as you find necessary.
    b. Using the training data, extract features from the text (i.e., vectorization using BOW and/or Bag of N-Grams and/or topics and/or lexical features and/or doc2vec).
    c. Use your favorite ML algorithm to train a classification model. Don't forget everything that we've learned in our ML course: hyperparameter tuning, cross validation, handling imbalanced data, etc. Make reasonable decisions and try to create the best-performing classifier that you can.
    d. Use the testing data to measure the accuracy and F1-score of your model.
2. [Text] Given the accuracy and F1-score of your model, are you satisfied with the results? Explain.

### *SOLUTION:*

After performing sentiment analysis via ML-based approach the **accuracy score is 0.7363** and **F1 score is 0.7187**. The model was very precise in identifying the positive sentiment while robust in identifying negative ones. I am not very satisfied with the results of the model, especially after performing detailed pre-processing and hyperparameter tuning. The low accuracy and F1 score are probably due to the variation of context and bag of words in the test set compared to the training set. That is, the training set is based on food and movie reviews while the test set is only on movie reviews. In addition, as shown in the word clouds below, there is a lot of overlap between words in positive and negative reviews. This makes it a difficult for the classifier to predict the sentiment on top of the variation of context.


*Word cloud - Training set*


*Word cloud – Test set*

3. [Text] Show five example instances in which your model was incorrect. Describe why the model was wrong.

*SOLUTION:*

Below are the five examples where the model was incorrect.

| Sentence | Actual | Predicted |
|---|---|---|
| The soundtrack wasn't terrible, either. | 1 | 0 |
| The only place good for this film is in the garbage. | 0 | 1 |
| I struggle to find anything bad to say about it. | 1 | 0 |
| I'm so sorry but I really can't recommend it to anyone. | 0 | 1 |
| It's a sad movie, but very good. | 1 | 0 |

As mentioned earlier the train and test set had different context, therefore using the classifier that was trained on different set of bag-of-words and to test it on a complete different context words is not ideal way of using a ML-based approach. For example, from number five above it is evident that the model learned "recommend" as a positive word especially when is comes to food reviews and classified it as positive sentiment when tested using movie only reviews. Therefore, this mishap could have been avoided if the train and test have similar context.

4. [Text] Compare and contrast the performance of the lexicon-based approach from Q2 with the ML-based approach here.

*SOLUTION:*

There were 113 incorrect predictions using lexicon-based approach and 144 using ML-based approach. Generally, ML-based approach performs better than lexicon-based approach, but it was not the case in this analysis. This is because in regular practice we train and tune a classifier on the dataset it is required to predict. But in this situation the train and test sets had a lot of differences which was not captured by the ML model. Lexicon-based approach did a better job because it was able to pick the *slag*, *social media type text, sarcasm,* etc. with no additional training and so was able to outperform ML model. If the train and test sets had similar context, ML-based approach would have once again performed better.

# 3. Sentiment Analysis via Deep ML-Based Approach

## Preamble

Download the "Product Sentiment" dataset: *sentiment_train.csv* and *sentiment_test.csv.*

## Tasks

1. [Code] Perform sentiment analysis on the dataset using the ML-based approach, using a deep learning algorithm.
    a. Load, clean, and preprocess the data as you find necessary.
    b. Transform the data into an embedding appropriate for deep learning approaches.
    c. Use a deep learning algorithm to train a classification model. Don't forget everything that we've learned in our ML course: hyperparameter tuning, cross validation, handling imbalanced data, etc. Make reasonable decisions and try to create the best-performing classifier that you can.
    d. Use the testing data to measure the accuracy and F1-score of your model.
2. [Text] Given the accuracy and F1-score of your model, are you satisfied with the results? Explain.

### *Solution:*

For test set, DL-based approach the **accuracy is 0.4560 and F1 score is 0.0000**. Whereas, for training set accuracy is 0.9991. The model failed to classify any positive sentiments correctly. I am not at all satisfied by the results because although the model is developed and trained very well due to external reasons it performed very poorly on the test set.

3. [Text] Show five example instances in which your model was incorrect. Describe why the model was wrong.

### *Solution:*

Below are the five examples where the model was incorrect.

| **Sentence** | **Actual** | **Predicted** |
|---|---|---|
| A good commentary of today's love and undoubtedly a film worth seeing. | 1 | 0 |
| For people who are first timers in film making, I think they did an excellent job | 1 | 0 |
| It was very popular when I was in the cinema, a good house and very good reactions and plenty of laughs | 1 | 0 |
| It's a feel-good film and that's how I felt when I came out of the cinema | 1 | 0 |
| It has northern humour and positive about the community it represents | 1 | 0 |

As mentioned above most of the wrongly predicted sentences are positive sentiments. Although the deep learning model is trained and tuned properly, it was unable to predict accurately, because the training set vocabulary used while training the model is different than the test set inputs, and a lot of terms that are in positive are also in negative (as seen in word clouds presented in question 2). For example, the term "good" is in both positive and negative reviews of both training and test set, this mix of terms caused the model to predict poorly.

4. [Text] Compare and contrast the performance of the lexicon-based approach from Q2 with the ML-based approach here.

*SOLUTION:*

Lexicon-based approach wrongly predicted 113 instances while DL-based approach did for 297 out of 546, which is $\sim 50\%$ of the instances were predicted incorrect. Generally, a DL-based approach outperforms a lexicon-based one because DL model is trained and tuned on the training set therefore, it provides best results. However, in this case the training set and test set had different vocabulary and so the model was unable to recognize and predict correctly. With Vader, it requires no additional training, so the test results were much better.