German University in Cairo
Media Engineering and Technology
Lecturer: Mervat AbuElkheir
TAs: Hadeel Mostafa

CSEN1076 Natural Language Processing and Information Retrieval

Winter term 2018

Project – Exploring Airline Twitter Sentiment Data

Your project will be to build a classifier for sentiment analysis from tweets, by learning from previously classified tweets. The task is to build a model that will determine the tone (neutral, positive, negative) of the text. To do this, you will need to train the model on the existing data. The resulting model will have to determine the class (neutral, positive, negative) of new texts (test data that were not used to build the model) with maximum accuracy.

Data Collection and Preprocessing

The task in this part is to read and preprocess a dataset compromised of 14640 tweets about six major U.S. airlines {American, Delta, Southwest, United, US Airways, Virgin America} (You can download the dataset from kaggle). Twitter data was scraped from February of 2015 and contributors were asked to first classify positive, negative, and neutral tweets, followed by categorizing negative reasons (such as "late flight" or "rude service"). The dataset is in csv format and contains the following attributes:

- *tweet_id*
- *airline_sentiment*
- *airline_sentiment_confidence*
- *negativereason*
- *negativereason_confidence:*
- *airline*
- *airline_sentiment_gold*
- *name*
- *negativereason_gold*
- *retweet_count*
- *text*
- *tweet_coord*
- *tweet_created*
- *tweet_location*

- *user_timezone*

The steps should be as follows:

a) Read all the tweets into your development environment.

b) Save the labels {positive, negative, neutral} of each tweet to a list.

c) Split the dataset into 80% training and 20% testing.

d) Do some preprocessing on the text: You tokenize and perform the processes of normalization, stemming, stop-word removal, case folding, and URL removal

e) Extract the features and classify

f) Evaluate

Scikit-Learn Classifiers

In this part, you will be using classifiers from `sklearn` to predict whether a tweet is positive, negative, or neutral.

The three classifiers to be used (import them from `sklearn`) are:

a) Multinomial Naive Bayes
b) K Nearest Neighbors Classifier
c) Random Forest Classifier

For the Random Forest Classifier, you can set `random_state=0` as not to have varying results with each run.

Feature Extraction

A feature is any variable which can help our classifier in differentiating between the different classes. You can produce features from tweets via two methods:

Method 1: You may use the text of a tweet to classify the tweet. You tokenize and perform the processes of normalization, stemming, stop-word removal, case folding, and URL removal, then build a term-document incidence matrix, a bag of words (`count matrix`), or *tf-idf* matrix.

Method 2: Alternatively, you can extract some structured and useful features that contribute to modeling of sentiment. The features that you may compute to classify positive vs. negative sentiment may be:

- Overall emoticon score (where 1 is added to the score in case of positive emoticon, and 1 is subtracted in case of negative emoticon)
- Overall score from online polarity lexicon MPQA (where presence of strong positive word in the tweet increases the score by 1.0 and the presence of weak negative word would decrease the

score by 0.5) (Alternatively, you may use any of the emotion lexicons in this link, with proper referencing in your final submission: http://saifmohammad.com/WebPages/lexicons.html)

- Unigram word models calculated using Naive Bayes
- Number of total emoticons in the tweet
- Number of positive emoticons in a tweet
- Number of negative emoticons in a tweet
- Number of positive words from MPQA lexicon in tweet
- Number of negative words from MPQA lexicon in tweet
- Number of base-form verbs in a tweet
- Number of past tense verbs in a tweet
- Number of present participle verbs in a tweet
- Number of past participle verbs in a tweet
- Number of 3rd person singular present verbs in a tweet
- Number of non-3rd person singular present verbs in a tweet
- Number of plural nouns in a tweet
- Number of singular proper nouns in a tweet
- Number of cardinal numbers in a tweet
- Number of prepositions or coordinating conjunctions in a tweet
- Number of adverbs in a tweet
- Number of wh-adverbs in a tweet
- Number of verbs of all forms in a tweet

The steps are:

a) Create a list for every feature, where every element is the feature value of a given tweet (or use a dictionary, key is feature name, value is feature list).

b) Build a feature matrix (list of lists), where every row corresponds to a tweet, and every column corresponds to a feature value of this tweet.

c) Feed the feature matrix and the labels to any of the `sklearn` classifiers.

You may use the `sklearn` feature extraction tutorial to do build the feature matrix in either Method 1 or Method 2.

Since there can be a large number of features, you may filter out the information that you need and discard the rest. This goes for the attributes already in the dataset and for the features that you extract (words in Method 1 or features in Method 2). Example ways to filter are:

Filtration of actual tweets:

- Remove Retweets (any tweet which contains the string "RT")
- Remove very short tweets (tweet with length less than 20 characters)
- Remove non-English tweets (by comparing the words of the tweets with a list of 2,000 common English words, tweets with less than 15% of content matching threshold are discarded)
- Remove similar tweets (by comparing every tweet with every other tweet, tweets with more than 90% of content matching with some other tweet is discarded)

Feature selection:

To reduce the complexity of feature selection, you may work with a subset of the features chosen at random. An example of a feature subset for Method two can be:

1. Unigram word models (for prior probabilities of words belonging to positive or negative classes)
2. Number of positive emoticons in tweet
3. Number of negative emoticons in tweet
4. Emoticon score for the tweet
5. MPQA score for the tweet

You may choose your subset randomly or based on some intuition. In all cases illustrate your choices.

Evaluation

The evaluation metric for this competition is harmonic F1-Score. The F1 score, commonly used in information retrieval, measures accuracy using the statistics precision p and recall r. Precision is the ratio of true positives (tp) to all predicted positives (tp + fp). Recall is the ratio of true positives to all actual positives (tp + fn). The F1 score is given by:

$$F_1 = 2 \times \frac{R \times P}{R + P}, where\ P = \frac{TP}{TP + FP}\ and\ R = \frac{TP}{TP + FN}$$

The F1 metric weights recall and precision equally, and a good retrieval algorithm will maximize both precision and recall simultaneously. Thus, moderately good performance on both will be favored over extremely good performance on one and poor performance on the other.

Bonus

You may earn bonus grades in one of Four ways:

1. Instead of random fixed selection of features as provided above, you select the most pertinent features by computing the **information gain** of all the features under exploration and then selecting the features with highest information gain. This means you have to learn about information gain. If interested, please indicate as such and we will provide illustrative material to help.
2. Build your model using both methods provided above and compare the performance.
3. Instead of the airline tweets, use the sentiment140 dataset, with more general tweets and a larger dataset size.
4. Filtration of actual tweets then compare between before and after filtration.

Submission

Deadline is on Saturday, 1st of December at 23:59. You should submit a Jupyter notebook named as:

Team “X”, where “X” is your team ID.

Best of Luck!

How to extend your project for future work (if you liked playing with the project)

Data in the form of raw tweets is acquired by using the python library “tweestream” which provides a package for simple twitter streaming API [26]. This API allows two modes of accessing tweets: SampleStream and FilterStream. SampleStream simply delivers a small, random sample of all the tweets streaming at a real time. FilterStream delivers tweets which match a certain criteria. It can filter the delivered tweets according to three criteria:

- Specific keyword(s) to track/search for in the tweets
- Specific Twitter user(s) according to their user-id’s
- Tweets originating from specific location(s) (only for geo-tagged tweets).

A programmer can specify any single one of these filtering criteria or a multiple combination of these.

References

Twitter Sentiment Analysis: https://arxiv.org/ftp/arxiv/papers/1509/1509.04219.pdf