Twitter Sentiment Analysis

By – HARSHIT BAJPAI

bajpaiharshit14@gmail.com

Project Overview –

Microblogging today has become a very popular communication tool among Internet users. Millions of users share opinions on different aspects of life every day in popular websites such as Twitter, Tumblr and Facebook. Spurred by this growth, companies and media organisation are increasingly seeking ways to mine these social media for information about what people think about their companies and products. **Sentiment analysis** is the process of computationally identifying and categorising opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral(Oxford dictionary). It is extremely useful for various businesses to monitor the attitude of consumers and then market their products accordingly. It helps businesses to accurately plan their marketing strategy. Not just businesses but political campaign runners also make use of this strategy to analyse the preferences of voters and their attitude towards the policies they are campaigning.

With over 500+ million Tweets (short text messages) per day, Twitter is becoming a major source of information. Out of all the popular social media platforms like Facebook, Google+, Myspace and Twitter, I choose Twitter because of the following reasons:

- Twitter contains an enormous number of text posts and it grows every day. The collected corpus can be arbitrarily large.
- Twitters audience varies from regular users to celebrities, company representatives, politicians, and even country presidents. Therefore, it is possible to collect text posts of users from different social and interests groups.
- Tweets are small in length, thus less ambiguous
- Tweets are unbiased in nature

Problem Statement –

For the purposes of this project we will be implementing a program that categorizes a user's tweets a positive or negative or neutral.

The values of class labels are – Positive – 4.0, Neutral – 2.0, Neutral – 0.0

I attempted to solve the problem by three ways –

1. K-Nearest Neighbour Algorithm(KNN)
2. SVM
3. Hybrid of KNN and SVM - SVM for polarity and KNN for Subjectivity/Objectivity Test

Refer Proposed Technique section for detail.

Overall our aim is simple. We want to allow the user to classify any tweet he wants and get its score as positive, negative or neutral.

Steps to run and resultant accuracy for all the three ways is mentioned in the result section.

Input Data –

The training dataset is collected SemEval challenge. This dataset was constructed for the Twitter sentiment analysis task in the Semantic Evaluation of Systems challenge (SemEval-2013).

(http://alt.qcri.org/semeval2014/task9/index.php?id=data-and-tools)

Stanford Twitter Sentiment Test Set (STS-Test)
(http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip)

Sanders dataset (http://www.sananalytics.com/lab/twitter-sentiment)

The testing dataset is from STS-Gold
(http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip)

The training dataset is split into 3 files containing a processed version of tweets in the three classes: positive (data/used/positive1.csv), negative (data/used/negative1.csv) and neutral (data/used/neutral1.csv)

Reference - http://ceur-ws.org/Vol-1096/paper1.pdf

Apart from these other libraries required are –

1. Sklearn - (http://scikit-learn.org/dev/index.html)
2. Numpy -(http://www.numpy.org/)
3. Nltk - (http://www.nltk.org/)

Refer Readme file for instructions on executing the code

Data Exploration –

Twitter is a social networking and microblogging service that allows users to post real time messages, called tweets. Tweets are short messages, restricted to 280(as of Nov 2017) characters in length. Due to the nature of this microblogging service, people use acronyms, make spelling mistakes, use emoticons and other characters that express special meanings. Following is a brief terminology associated with tweets.

- Emoticons: These are facial expressions: pictorially represented using punctuation and letters; they express the user's mood.
- Target: Users of Twitter use the @ symbol to refer to other users on the microblog. Referring to other users in this manner automatically alerts them.
- Hashtags: Users usually use hashtags to mark topics. This is primarily done to increase the visibility of their tweets.

Tools –

1. Emoticon Dictionary: We use the emoticons list as given in and manually annotate them. Following is a sample table of the dictionary and the score of emoticons. In this project, Features.py categorise the emoticons into four classes:

a) Extremely- Positive

b) Positive

c) Extremely- Negative

d) Negative

Emoticon	Score
D : D8D = DXv.vDx	Extremely Negative
: −/ : / = / = < /3	Negative
>:)B)B−) :) : −) >	Neutral
: −) :) : o) :] : 3	Positive
: −D : D8DxDXD	Extremely Positive

Emoticon Dictionary

2. Internet Slangs**:** Since Twitter has restriction on number of characters to be used, people use slangs very frequently. For example – ASAP means as soon as possible. This allows people to express their intent in less words. So it is very important to know the meaning of the slang and replace it with its meaning while calculating the sentiment of a phrase.

Following are a few examples of slangs or acronyms commonly used –

Slangs	Meaning
LOL	Laughing out loud
BTW	By the way
ROFL	Rolling on Floor Laughing
OMG	Oh my god

Slangs table

3. AFINN –

AFINN is a list of English words rated for valence with an integer between minus five (negative) and plus five (positive). The words have been manually labeled by Finn Årup Nielsen in 2009-2011. The file is tab-separated.

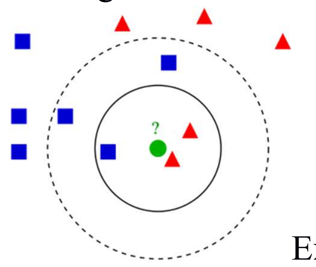
Here is an example table for AFINN dictionary –

Words	Score
Admit	-1
Admiring	3
Bastard	-5
Breathtaking	5

Proposed Technique –

First approach to solve the problem was by using K nearest neighbour algorithm. In this approach we applied KNN algorithm by using function neighbors provided in Scikit (http://scikitlearn.org/stable/modules/neighbors.html)

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.



Example of k-NN classification. The test sample (green circle) should be classified either to the first class of blue squares or to the second class of red triangles. If k = 3 (solid line circle) it is assigned to the second class because there are 2 triangles and only 1 square inside the inner circle. If k = 5 (dashed line circle) it is assigned to the first class (3 squares vs. 2 triangles inside the outer circle).

Second approach is to solve it by SVM classifier. The reason for this was better accuracy and SVM is perfectly able to deal with linear data by separating data into different classes. This is how SVM works – Suppose a dataset D = {Xi, Yi} exist where Xi is set of tuples and Yi is associated class label of tuples. Class labels are -1 and +1 for no and yes category respectively. The goal of SVM is to separate negative and positive training example by finding n-1 hyperplane. SVM is considered a modern technique achieving fast acceptance due to the good results achieved in many fields of data mining problems, based on its solid foundation in statistical learning theory.

It has been well researched fact that the well-performing SVM classification approach may suffer from high time consumption, high CPU and physical memory usages, due to its convoluted training and classifying processes, especially when the dimensionality of data is high. On the other hand, KNN classification approach is outstanding with its simplicity and low cost training process. However, it has been reported to be less accurate than the SVM classification. In the **final approach** I have created a hybrid of SVM and KNN by having SVM for polarity and KNN for Subjectivity and Objectivity test.

Pre-processing –

In this section we will discuss the pre-processing of data so that it can be used for classification.

1. Repetition of characters –
   Twitter provides a platform for users to express their opinion in an informal way. Tweets are written in random form, without any focus given to correct structure and spelling. Spell correction is an important part in sentiment analysis of user-generated content. People use words like ’whaaaaaaaaaaaat’ and ’hunnnnngry’ in order to emphasise the emotion. In order to capture such expressions, we look for 2 or more repetitions of character and replace with the character itself. For example, wooooow is replaced by wow.

2. Stop-word Removal –
   Stop words play a negative role in the task of sentiment classification. Stop words occur in both positive and negative training set, thus adding more ambiguity in the model formation. And also, stop words don’t carry any sentiment information and thus are of no use to us. We create a list of stop words like he, she, at, on, a, the, etc. and ignore them while scoring the sentiment.

3. Replace Slangs –
   As explained above, people use slangs very often to overcome the word restriction. They are not proper English words, so they are not found in AFINN dictionary. So we replace these slangs by taking their meanings from internetSlang.txt and then they are used for calculation.

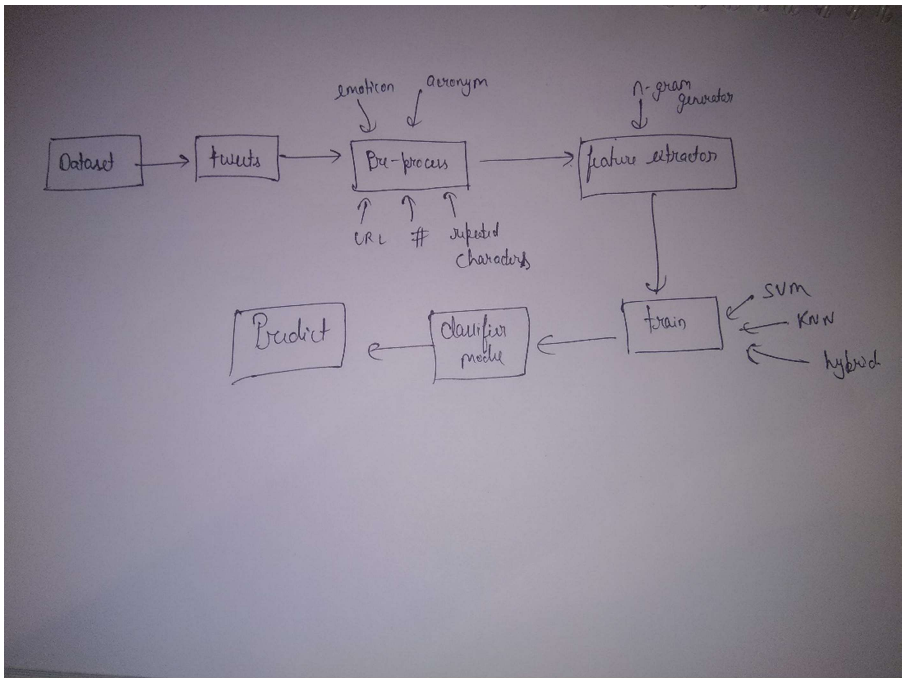
4. Replace Target –
   The target mentions in a tweet done using ’@’ are usually the twitter handle of people or organisation. This information is also not needed to determine the sentiment of the tweet. Hence they are converted to AT_USER.

5. Replace URL –
   People use URLs in their tweet to reference some information. These URLs do not play any part in determining the sentiment and hence they are also replaced by converting www.* or https?://* to URL.

6. Replacing Emoticons –
   Emoticons play an important role in determining the sentiment of the tweet. Hence we replace the emoticons by their sentiment polarity by looking up in the Emoticon Dictionary.

Implementation –

In this section we will discuss the implementation and approach to solve the problem. Here is a flow chart for solving the problem statement –



We perform the pre-processing steps listed above to clean the tweets. We then use feature extractor to generate unigram vector and calculate its positive (posuni), negative (neguni) and neutral score (neuuni).We calculate polarity of the tweet using SentiWordNet3.0 dictionary also consider the fact for giving more weightage to # tags containing word

Refinement –

As listed above, we have calculated results by using three different methods – KNN, SVM and a hybrid of both KNN and SVM model. Results for all three can be found in the result section.

RESULTS -

Model	Accuracy
Hybrid	0.588353413655
SVM	0.56
KNN	0.54

We can clearly see that hybrid model performs best and has the highest accuracy rating and KNN has the least accuracy. Instruction on how to achieve this rating is in Readme file.

Error Analysis -

We manually investigate the phrases or messages which were wrongly labelled by the system. It is quite clear that the model gives error when ambiguous tweets are pushed. For ex. 'yeah,right!' is negative , 'hell,yeah' is a positive tweet. These sarcastic intent is what leads to error in analysis. Following is a table for few of the sample responses

Please note that all the output are of hybrid model.

Statement	Output	Analysis
Yeah,right	Positive	Wrong
Are you kidding me	Neutral	Wrong
I hate you	Negative	Correct
I love Udacity	Positive	Correct

Conclusion

I have presented results for sentiment analysis on Twitter. It reports an overall accuracy for 3-way classification tasks: positive versus negative versus neutral. I investigated two methods first: KNN and SVM first and demonstrate that combination of both these (hybrid) perform the best. In future work, I will explore even richer linguistic analysis, for example, parsing, semantic analysis and topic modelling.

References –

- Acronyms - https://www.noslang.com/dictionary/
- AFINN - http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010
- Emoticon - https://en.wikipedia.org/wiki/List_of_emoticons
- Dataset – Kindly refer to input data section
- Sentiment Analysis on Twitter Data using KNN and SVM by Mohammad Rezwanul Huq https://thesai.org/Downloads/Volume8No6/Paper_3-Sentiment_Analysis_on_Twitter_Data_using_KNN_and_SVM.pdf
- Sentiment Analysis: How to Derive Prior Polarities from SentiWordNet by Marco Guerini - https://arxiv.org/pdf/1309.5843.pdf
- Evaluation Datasets for Twitter Sentiment Analysis by Hassan Saif1, Miriam Fernandez1 , Yulan He2 and Harith Alani1 - http://ceur-ws.org/Vol-1096/paper1.pdf
- 5 Minutes With Ingo: Understanding Support Vector Machines - https://www.youtube.com/watch?v=YsiWisFFruY
- https://docs.cs50.net/problems/sentiments/sentiments.html#code-positive-words-txt-code-code-negative-words-txt-code
- https://nlp.stanford.edu/sentiment/
- https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/
- https://marcobonzanini.com/2015/01/19/sentiment-analysis-with-python-and-scikit-learn/