Title: Towards detecting star quality review sentiment from Beer review

Author: Sunny Changediya (A20353568)

Problem Overview:

USA has heavy daily consumption of alcohol and particularly of Beer. It's been observed that Beer Lovers are very peculiar about taste of beer and they generally tend to spend time reviewing beer. Most of the social network sites provide collective review of breweries and beers. But it is observed that most beer reviews are long and comprised of noise in terms of scores such as look, smell, taste, and feel which adds predominantly to actual beer review. Most of the time even though Beer not good, user tend to give good rating based on place, ambiance, or service and that counts towards overall Beer review. In general it becomes hard to predict actual beer rating from the sentiment of review. This situation is very confusing to other beer lovers trying to figure out actual beer sentiment. So In this paper, we tried to predict sentiment of a beer from the noisy beer review.

Data:

1. Beer reviews:
   The raw review data used for this analysis is crawled from famous social network BeerAdvocate (https://www.beeradvocate.com). The review html pages were crawled using python requests library and filtered into training and testing instances. The reviews collected contains only most-recent reviews for Beer.

2. Afinn word sentiment:
   The afinn data contains words from English library with integer valence associated with it. Afinn is used to rate sentence as positive or negative based on terms in it. The data is downloaded using python request library from http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010

Method:

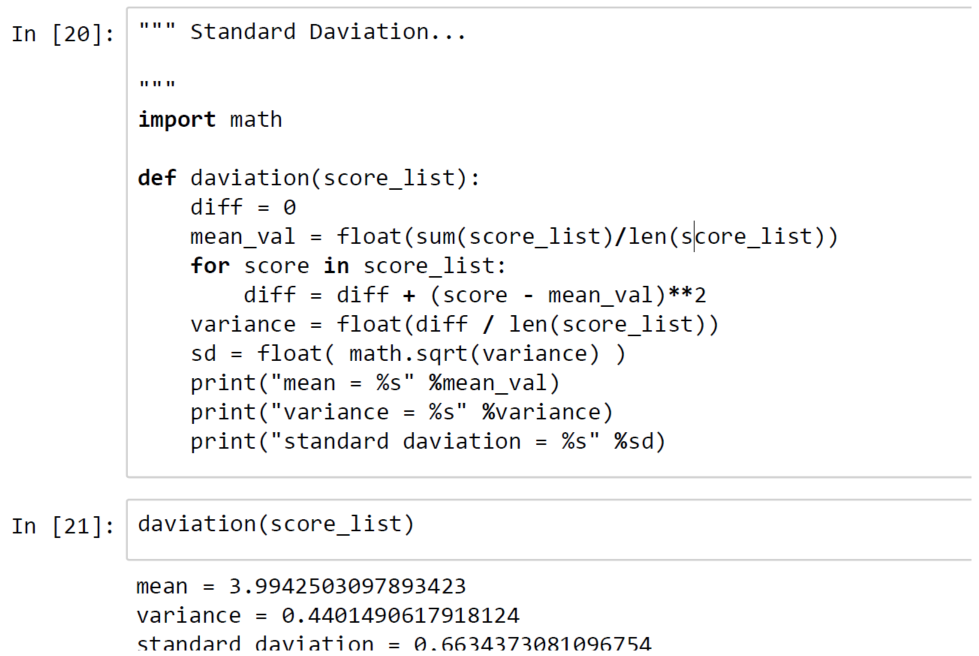
Logistic Regression is the primary algorithm used for sentiment analysis of reviews. Following are some methods and algorithms used for pre-processing and feature selection from data.

1. For feature selection, we use methods such as tf-idf score, token pair features and Afinn sentiment.
2. More detailed feature selection involves generating Parse-Tree structure using Stanford dependency parser library and word_tokenize() method.
3. We also tried to take reference of NLTK stopwords database using library nltk.corpus.stopwords
4. For classification purpose, we use in-built libraries such as scipy.sparce.csr_matrix and sklearn LogisticRegression() classification algorithm.
5. For classification and evaluating best setting results, we use k-fold cross validation accuracy measure to find best setting for reviews.

To evaluate results, I used 5-fold cross validation accuracy as a measure of detecting accuracy per settings of feature selection. For all combinations of feature selection, we calculated cross validation and found best setting of feature selection which will be used to classify training set and then predict on testing set using LogisticRegression().

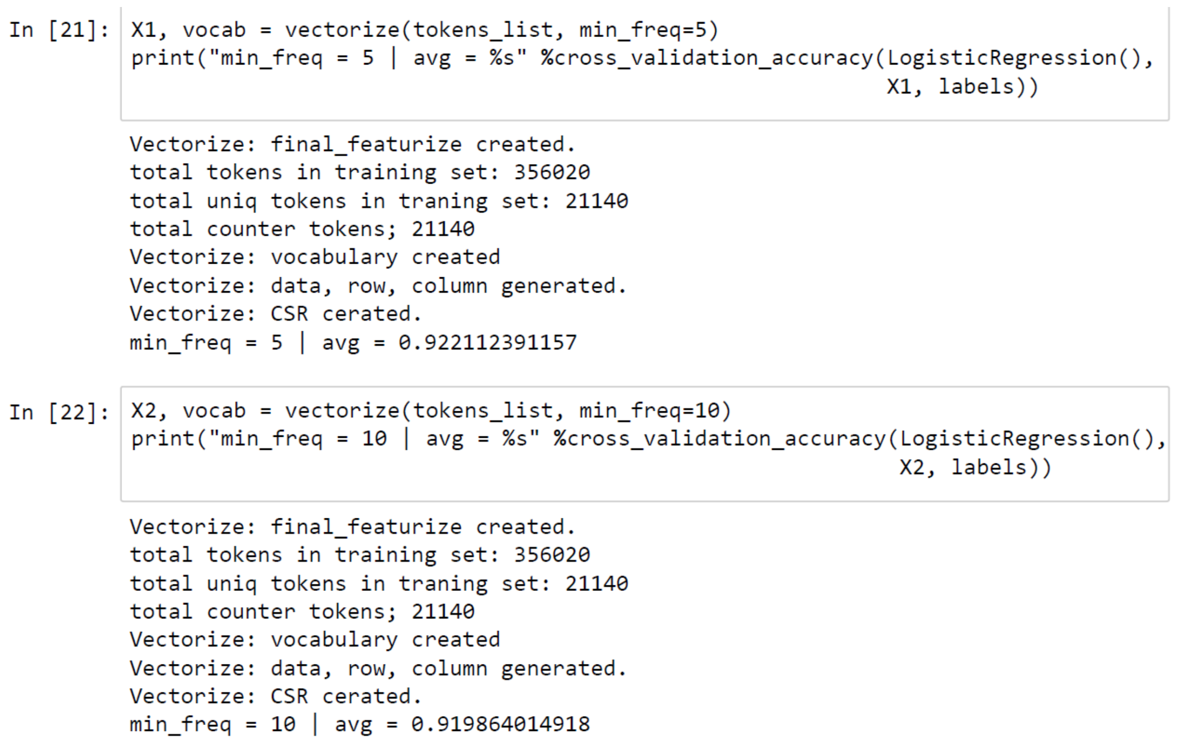
Intermediate Results:

1. Mean and Standard deviation of data collected:



```
In [20]: """ Standard Daviation...

         """
         import math

         def daviation(score_list):
             diff = 0
             mean_val = float(sum(score_list)/len(score_list))
             for score in score_list:
                 diff = diff + (score - mean_val)**2
             variance = float(diff / len(score_list))
             sd = float( math.sqrt(variance) )
             print("mean = %s" %mean_val)
             print("variance = %s" %variance)
             print("standard daviation = %s" %sd)

In [21]: daviation(score_list)

         mean = 3.9942503097893423
         variance = 0.4401490617918124
         standard daviation = 0.6634373081096754
```

2. Cross-Validation accuracy on 5-fold:



```
In [21]: X1, vocab = vectorize(tokens_list, min_freq=5)
         print("min_freq = 5 | avg = %s" %cross_validation_accuracy(LogisticRegression(),
                                                                     X1, labels))

         Vectorize: final_featurize created.
         total tokens in training set: 356020
         total uniq tokens in traning set: 21140
         total counter tokens; 21140
         Vectorize: vocabulary created
         Vectorize: data, row, column generated.
         Vectorize: CSR cerated.
         min_freq = 5 | avg = 0.922112391157

In [22]: X2, vocab = vectorize(tokens_list, min_freq=10)
         print("min_freq = 10 | avg = %s" %cross_validation_accuracy(LogisticRegression(),
                                                                      X2, labels))

         Vectorize: final_featurize created.
         total tokens in training set: 356020
         total uniq tokens in traning set: 21140
         total counter tokens; 21140
         Vectorize: vocabulary created
         Vectorize: data, row, column generated.
         Vectorize: CSR cerated.
         min_freq = 10 | avg = 0.919864014918
```

3. TOP-Miss-classified documents:

```
truth=1 predicted=0 proba=0.999998
Look: Opaque light golden/tan color, frothy off-white head Smell: bready malts
 and  yeast, little bit of banana and maybe some other fruits, Taste: Sweet mal
ts and yeast, apple, little bit of spice, citrus (maybe the hops?). Overall pre
tty sweet, with some sour apple/citrus.  Slightly bitter and sour finish. Feel:
Crisp carbonation, medium-bodied Overall a high quality tripel.

truth=0 predicted=1 proba=0.999991
L: Poured a nice orange-amber color with a thick layer of head on top. Good amo
unt of carbonation on this one out of the can leading to a steady flow of bubbl
es to top of the glass. Leaving nice lacing on the glass. Looks like a classic
 West Coast style IPA.  S: Man, does the Citra shine through on the nose of thi
s one! Oranges and tangerines followed by a bit of piney resin, but not too muc
h. Nice combination of fruit juiciness and sticky hop aromas.  T: Tastes exactl
y as I would expect: piney, sticky, slightly earthy, and a nice juicy backbone.
The 8.2% alcohol is hidden well.  F: As stated above, sticky and juicy. It's go
t a great body on it - exactly as I had hoped it would be. No alcohol burn on t
he way down. Good carbonation as well, as was evident from the pour.  O: Solid,
solid beer here. It gets every aspect of a classic West Coast style DIPA down c
orrectly. This one is dangerous because it doesn't drink like an 8.2% ABV beer.
Very tasty.
```

Related work:

1. http://nlp.stanford.edu/courses/cs224n/2010/reports/amirg.pdf
2. http://nlp.stanford.edu/courses/cs224n/2010/reports/dpreston-rmelnick.pdf
3. http://nlp.stanford.edu/courses/cs224n/2010/reports/ekuefler-estelle.pdf
4. http://nlp.stanford.edu/courses/cs224n/2010/reports/pgrafe-moontae.pdf
5. http://nlp.stanford.edu/courses/cs224n/2010/reports/rothfels-jtibs.pdf

The above papers (2, 3, 4, 5) mostly focus on improving feature selection using variants of bag of words such as token-features, lexical features, token-pair-features etc. Depending on raw data available, they tried to use different classifier model for different data format. Paper (1) presents Maximum-entropy and LogisticRegression() classification using n-gams feature selection. In our paper, we usually emphasize on improving feature selection by adding NLP features such as token-pair, parse tree, and Afinn valence feature. For evaluating best setting, we used 5 fold cross validation instead of naïve method such as mean-error.

Timeline:

1. Implementation of Parse-tree feature selection:
   This is a one more important feature selection which may impact classification predominantly. We will be implementing this feature selection method using Stanford dependency parser.
2. Rating Model:
   Based on cross validation accuracy, we will be implementing rating module to start rate each review based on LogisticRegression() classifier result. The reviews will be rated on a scale of 5 to identify beer sentiment.
3. Maximum-Entropy:
   We may try to use maximum entropy classifier model and compare to efficiency of LogisticRegression().

References:

N/A