NLP Lab 8

Dixitha Kasturi
dkasturi@syr.edu

Topic: Continuation of classification and feature exploration

I explored both Option 1 and Option 2.

Option 1:

Accuracy with stop words: 0.76

Accuracy with stop words removal: 0.788

The accuracy did increase, which in a way showed that the stop words were indeed contributing negatively to the model's overall performance. I removed the stop words in the NOT_featuresets function

Top 30 features with stop words:

```
Most Informative Features
           V_engrossing = True              pos : neg    =     19.5 : 1.0
               V_stupid = True              neg : pos    =     17.8 : 1.0
             V_mediocre = True              neg : pos    =     17.1 : 1.0
              V_generic = True              neg : pos    =     15.1 : 1.0
            V_inventive = True              pos : neg    =     14.9 : 1.0
              V_routine = True              neg : pos    =     14.4 : 1.0
               V_boring = True              neg : pos    =     14.2 : 1.0
                 V_flat = True              neg : pos    =     13.5 : 1.0
               V_unique = True              pos : neg    =     12.9 : 1.0
        V_refreshingly = True              pos : neg    =     12.9 : 1.0
           V_refreshing = True              pos : neg    =     12.9 : 1.0
            V_wonderful = True              pos : neg    =     12.5 : 1.0
                   V_90 = True              neg : pos    =     12.4 : 1.0
                 V_warm = True              pos : neg    =     11.3 : 1.0
                V_stale = True              neg : pos    =     11.1 : 1.0
          V_mesmerizing = True              pos : neg    =     10.9 : 1.0
                  V_car = True              neg : pos    =     10.4 : 1.0
             V_mindless = True              neg : pos    =     10.4 : 1.0
                 V_dull = True              neg : pos    =     10.3 : 1.0
              V_quietly = True              pos : neg    =     10.3 : 1.0
             V_captures = True              pos : neg    =     10.1 : 1.0
             V_powerful = True              pos : neg    =     10.0 : 1.0
             V_annoying = True              neg : pos    =      9.7 : 1.0
             V_provides = True              pos : neg    =      9.7 : 1.0
             V_chilling = True              pos : neg    =      9.6 : 1.0
                V_waste = True              neg : pos    =      9.5 : 1.0
             V_supposed = True              neg : pos    =      9.1 : 1.0
             V_tiresome = True              neg : pos    =      9.1 : 1.0
           V_meandering = True              neg : pos    =      9.1 : 1.0
           V_unexpected = True              pos : neg    =      8.9 : 1.0
```

Top 30 features after stop words removal:

```
Most Informative Features
           V_engrossing = True              pos : neg    =     19.7 : 1.0
              V_routine = True              neg : pos    =     14.9 : 1.0
            V_inventive = True              pos : neg    =     14.4 : 1.0
              V_generic = True              neg : pos    =     13.6 : 1.0
                   V_90 = True              neg : pos    =     13.6 : 1.0
             V_mediocre = True              neg : pos    =     13.6 : 1.0
                 V_flat = True              neg : pos    =     13.4 : 1.0
            V_absorbing = True              pos : neg    =     13.0 : 1.0
           V_refreshing = True              pos : neg    =     13.0 : 1.0
             V_intimate = True              pos : neg    =     13.0 : 1.0
            V_NOTenough = True              neg : pos    =     12.3 : 1.0
                 V_warm = True              pos : neg    =     12.2 : 1.0
            V_wonderful = True              pos : neg    =     12.2 : 1.0
               V_boring = True              neg : pos    =     12.1 : 1.0
                 V_dull = True              neg : pos    =     12.0 : 1.0
         V_refreshingly = True              pos : neg    =     11.7 : 1.0
               V_stupid = True              neg : pos    =     11.0 : 1.0
                   V_tv = True              neg : pos    =     10.7 : 1.0
               V_beauty = True              pos : neg    =     10.6 : 1.0
             V_provides = True              pos : neg    =     10.6 : 1.0
                 V_thin = True              neg : pos    =     10.6 : 1.0
             V_touching = True              pos : neg    =     10.5 : 1.0
        V_extraordinary = True              pos : neg    =     10.4 : 1.0
                V_stale = True              neg : pos    =     10.3 : 1.0
             V_powerful = True              pos : neg    =      9.9 : 1.0
             V_captures = True              pos : neg    =      9.8 : 1.0
             V_tiresome = True              neg : pos    =      9.6 : 1.0
               V_unless = True              neg : pos    =      9.6 : 1.0
               V_flawed = True              pos : neg    =      9.4 : 1.0
             V_document = True              pos : neg    =      9.0 : 1.0
```

Option 2:

When the given function was used with 4 numeric features the following accuracy was obtained

```
# retrain the classifier using these features
train_set, test_set = SL_featuresets[1000:], SL_featuresets[:1000]
classifier = nltk.NaiveBayesClassifier.train(train_set)
nltk.classify.accuracy(classifier, test_set)
```

```
0.746
```

When the given function was used with 1 numeric feature the following accuracy was obtained

```
def SL_features2(document, word_features, SL):
    document_words = set(document)
    features = {}
    for word in word_features:
        features['V_{}'.format(word)] = (word in document_words)
    # count variables for negative and positive
    weakPos = 0
    strongPos = 0
    weakNeg = 0
    strongNeg = 0
    for word in document_words:
        if word in SL:
            strength, posTag, isStemmed, polarity = SL[word]
            if strength == 'weaksubj' and polarity == 'positive':
                weakPos += 1
            if strength == 'strongsubj' and polarity == 'positive':
                strongPos += 1
            if strength == 'weaksubj' and polarity == 'negative':
                weakNeg += 1
            if strength == 'strongsubj' and polarity == 'negative':
                strongNeg += 1
            negcount = weakPos + strongPos
            poscount = weakNeg + strongNeg
            features['totcount'] = negcount - poscount
    return features
```

```
train_set2, test_set2 = SL_featuresets2[1000:], SL_featuresets2[:1000]
classifier2 = nltk.NaiveBayesClassifier.train(train_set2)
nltk.classify.accuracy(classifier2, test_set2)
```

```
0.758
```

```
def SL_features3(document, word_features, SL):
    document_words = set(document)
    features = {}
    for word in word_features:
        features['V_{}'.format(word)] = (word in document_words)
    # count variables for negative and positive
    negcount = 0
    poscount = 0
    for word in document_words:
        if word in SL:
            strength, posTag, isStemmed, polarity = SL[word]
            if polarity == 'positive':
                poscount += 1
            if polarity == 'negative':
                negcount += 1
            features['totcount'] = negcount - poscount
    return features
```

```
train_set3, test_set3 = SL_featuresets3[1000:], SL_featuresets3[:1000]
classifier3 = nltk.NaiveBayesClassifier.train(train_set3)
nltk.classify.accuracy(classifier3, test_set3)
```

```
0.758
```

The accuracy increased when only 1 numeric feature was considered. I wrote 2 versions of the same action to verify if my approach was correct.