Data Mining: Assignment 3

Lucia-Eve Berger (lucia.berger@mail.mcgill.ca)
Giorgi Gabunia (ggabunia@unibz.it)
European Masters in Software Engineering
`https://github.com/lberger1993/data_mining_sentiment_tweets`

December 12, 2017

Question (1). how did you construct the attributes.

Solution. The attributes were constructed after cleaning the data. The csv was passed through a pandas (python) data-frame. In the data-frame, the text-content was isolated (tweet-column).

- When isolated,
  - text was transformed to lowercase
  - punctuation was removed
  - stop words were removed
  - emoji characters, links and user names were removed
- Creating the arff structure (weka)
  - terms were counted
  - top 500 attributes were taken
  - added the attribute **classes** and appended the **(0,2,4)**. These labels represent the tweet's sentiment. The 0 is the most negative and the 4 is most positive.
  - constructed into an .arff file

□

Question (2) A comparison between the two models with different settings and evaluation strategies.

Solution. **Bayesian vs. J48 Decision Trees:**
After extensive testing with the unfiltered data(i.e. including all the words as attributes), we conjure the Naive Bayesian classifier performed better than the J48 Decision Tree (Table 1) because:

1. **Independence assumption**: The Naive Bayesian *assumption of independence* allows all attributes to be independent given the value of the class variable. While this rarely is the case in real world applications [1], in our application it yielded accurate classifications and low error rate. We think that one of the reasons could be that in a language, especially short sentences like twitter, the words **are** relatively independent from other ones, if not directly related to them. Direct relation could be considered as "being a parent" in the context of Bayesian network, which could explain the good results. It was also considerably faster than the J48 Tree.(0.06 seconds vs. 1.19 seconds)

[1] http://researchpublications.org/IJCSA/NCAICN-13/189.pdf

Table 1: Table 1: Classification results (at cross validation)

	Native Bayes (%)	J48 Tree (%)
Correctly Classified Instances	71.0843	61.0442
Incorrectly Classified Instances	28.9157	38.9558
Mean absolute error	0.1849	0.2256
Relative absolute error	55.7722	68.0689

Table 2: Table 2: Detailed Accuracy By Class (at cross validation)

	Native Bayes	J48 Tree
TP Rate (abc)	0.711	0.610
FP Rate (abc)	0.142	0.209
Precision	0.715	0.630

2. **Algorithm**: We estimate the Naive Bayesian also performed better than the J48 Tree due to the tree's algorithm. Namely the tree consistently over fits the sample data which usually necessitates pruning, either pre-prune, or post-prune. Pruning causes lack of precision because of several leafs are merged into one. Weka's J48 tree implementation automatically prunes the tree, causing lower precision classification.

3. **Detailed Accuracy By Class**: As shown in Table 2, the weighted TP (True Positive) rate was higher for the Native Bayes. Arguably more important though is the J48 Tree's higher FP Rate. This higher FP rate is a call for concern. While not pressing in the sentiment application, if this were a survey on customer satisfaction, this would greatly misrepresent the sample. In further analysis, cost would be applied to each to see the true damage of the misclassification.

□

Solution. **Best methodology:**
Using training set methodology saw superior results under both classifiers however, in practice, the cross validation is the stronger methodology.

Training set In our estimation, the training set methodology had high performance however, is bad practice. The algorithm uses the training data as the test data which naturally gave us high results. In practice, this would be an unacceptable metric to use as code or models could be formed on the segments. It gives us no indication of how well it would perform on new, unknown data.

Cross validation: In our assessment, cross validation was the best methodology to use. It is the best because it *folds the data*. At 10-folds, the data has been processed 10 times which may have lower accuracy, however, is more thorough. Besides, in increases the probability that many different combinations of sets will be used as training and test data respectively, ensuring that outliers and noise will be filtered out.

Splitting value: Even with experimentation on splitting value (10%, 33%, 66%) percentage split consistently showed the worst results. We estimate its poor performance due to lack of data distribution and data stratification. In other words, it chooses an arbitrary set that may not be representative of the data on a whole.

□

Question (3) the effect of using only positive and negative attributes..

Solution. Despite our initial expectations, the *filtered* dataset performed worse under both the Naive Bayesian Classification and the J48 Decision Tree than the *unfiltered*.

In our analysis, when only the positive and negative attributes are kept, the sample is a bare skeleton. In other words, we lose all the words that give context to those *positive* and *negative* expressions, making the learning process less efficient. Besides, some sentiments can be negative or positive without containing explicitly negative or positive words, which the classifier would miss if we filter those out.

For example, as we saw in class, *his health situation was serious.* If we only analyze the positive and negative word sources, we lose too much information to classify it precisely. In this example, both would probably be classified as "neutral", while in reality first one is positive and second - negative.

For example, in our dataset, we classify the tweet

```
['good', 'relief', 'sick', 'scams', 'stealing']
```

with a label '4' (Positive). With only these words, it's very challenging to hone any real meaning or sentiment from the sample.

Another important issue is the fact that choosing only the negative or positive words leaves some tweet representations empty, i.e. no keywords ([]). That would make classifying them impossible except for guessing: two completely different tweets can both end up empty with this approach and both would be labeled same.

□

Corollary Optional behavior on larger set:.

Solution. When preparing the larger set, we preformed the same cleaning as described in question 1. Unable to run the full file, the file was reduced to 25,000 tweets, 12500 with label 0 and 12500 with label 4. Using this 25,000 tweets, our classified accuracy for both methods was approximately 47%.

We project this lack of accuracy due to the larger data set only contains 0 and 4, the most extreme labels. Thus, all the tweets that were "neutral" (2) in the test set would naturally be misclassified - the model did not consider "neutral" sentiment at all. It is also possible our test set is too small to achieve an acceptable accuracy.

□