# Sentiment prediction in user reviews

**Jakub Ziec, Marcin Trzcinski**

University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
jakziec@gmail.com, mrctrzcinski@gmail.com

**Abstract**

This document proposes a method and shows result of sentiment classification and rating prediction model for user reviews on a certain product from different doimains such as: books, dvds, ect. Each user review is classified as either Positive, or Negative. Additionally, the system is be able to predict the rating assigned to the product based on the review (i.e. on a scale from 1 to 5).

## 1. Introduction

The goal of this project is to build a sentiment classification and rating prediction for textual user reviews on certain products. Reviews were taken from multiple domains, including books, DVDs, electronic devices etc. The classification model was evaluated in terms of accuracy, precision, recall and F1-score. The rating prediction model was evaluated using a regression metric called Pearson correlation.

## 2. Data prepatarion

As we used three diffrent tools i.e. lingpipe in java program, libshorttext and libsvm, we preapared our raw input data[1] in three diffrent formats. For the first one we used xml format and for others format required by used libraries.

## 3. Sentiment Classification

For a classification problem i.e. deciding whether review has positive or negative sentiment, we chose two techniques: Naive Bayes classifier and support vector classification by Crammer and Singer. The first one we used as baseline.

### 3.1. Naive Bayes classifier

Naive Bayes classifier is simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. The Bayes classifier is the function that assigns a class label $\hat{y} = C_k$ for some $k$ as follows:

$$\hat{y} = \underset{k \in \{1,..,K\}}{\operatorname{argmax}} P(C_k) \Pi_{i=1}^{n} P(x_i | C_k) \tag{1}$$

In our implementation we used *Java* library *lingpipe*. We represented document/review as *bag of words* (**BoW**). We removed stop-words and removing all non-letter characters.

#### 3.1.1. Results

The most frequent words given from our model for electronics domain for positive and negative training cases are shown in tables 1 and 2.

[1] http://www.cs.jhu.edu/~mdredze/datasets/sentiment/domain_sentiment_data.tar.gz

Table 1: The most frequent positive words

| Word | nomber of occurrence |
|---|---|
| i | 2721 |
| you | 915 |
| my | 859 |
| t | 519 |
| very | 339 |
| use | 323 |
| great | 311 |
| good | 310 |
| sound | 295 |
| these | 292 |
| your | 290 |
| just | 244 |
| quality | 231 |
| like | 231 |

The model for electronics domain gives following results:

$$\begin{aligned} TestCases &= 497 \\ Correct &= 324 \\ Accuracy &= 0.65191 \\ Precision &= 0.65060 \\ Recall &= 0.65322 \\ F1 &= 0.65191 \end{aligned} \tag{2}$$

### 3.2. Support vector classifier by Crammer and Singer

We decided to use libshortext library for support vector classification by Crammer and Singer. LibShortText is an open source tool for short-text classification and analysis. It can handle the classification of, for example, titles, questions, sentences, and short messages - The fast training and testing is built upon the linear classifier.

We used libshorttext in following manner:

```
$ text-train.py -f -G 1 train_set
$ text-predict.py -f test_set model \
result.txt
```

Parameter -G 1 means that we using libshorttext with enable grid search (slightly better results than without). It

Table 2: The most frequent negative words

| Word | nomber of occurrence |
|---|---|
| i | 3078 |
| my | 827 |
| you | 711 |
| t | 673 |
| me | 311 |
| get | 276 |
| just | 272 |
| work | 255 |
| these | 253 |
| your | 236 |
| product | 226 |
| very | 226 |
| them | 217 |
| use | 215 |

uses no stopword removal, no stemming, bigram preprocessor options and binary feature representation.

## 4. Rating prediction

For raiting prediction we used support vector regression (SVR). The model produced by support vector classification depends only on a subset of the training data, because the cost function for building the model does not care about training points that lie beyond the margin. Analogously, the model produced by SVR depends only on a subset of the training data, because the cost function for building the model ignores any training data close to the model prediction.

In our solution we used libsvm library with following comend:

```
$ svm-train -s 3 -t 2 -c 20 -g 64 \
-p 0.1 train_set
```

which means that we used epsilon-SVR regression with kernel function equal to $\exp(-\gamma * |u - v|^2)$, hyperparameter $C$ equal to 20, $\gamma$ parameter equal to 64 and $\epsilon$ equal to 0.1. For those parameters we obtain following results for electronics domain:

$$Mean\_squared\_error = 1.13938$$
$$Squared\_correlation\_coefficient = -4.00855e-15 \quad (3)$$

## 5. Demonstration program

Our demonstraition program was written in *Python 2.7* under *Ubuntu 14.04.1 LTS*. It requires *Java 1.8*, *libshorttext-1.1* and *libsvm-3.20*.

### 5.1. User Interface

Our program has command line interface (see figure 1).

Example of using:

```
sentiment-prediction/src$ python \
main.py -m 3 -d 0
```

```
Usage: main.py [options] arg

Options:
  -h, --help            show this help message and exit
  -j JAVA, --java=JAVA  set java directory
  --libsmv=LIBSVM       set libsmv path
  --libshorttext=LIBSHORTTEXT
                        set libshorttext path
  -m METHOD, --method=METHOD
                        set method 0-CLASSIFICATION, 1-RE
GRESSION, 2-BAYES
  -d DOMAIN, --domain=DOMAIN
                        set domain 0-BOOKS, 1-DVD, 2-ELEC
TRONICS, 3-KITCHEN
  -c, --clean
```

Figure 1: Command line interface

will execute sentimental classificasion using Naive Bayes classifier and data from "books" domain.

An example output is show in figure 2.

```
/usr/bin/java -jar dist/sentiment-bayes.jar 0


Training.
  # Training Cases=1498
  # Training Chars=1499280

Evaluating.
  # Test Cases=498
  # Correct=315
  # ACC=0.6325301204819277
  # P=0.6398305084745762
  # R=0.606425702811245
  # F1=0.622680412371134
```

Figure 2: Program output

## 6. Conclusion

How we can see Accuracy in vector classification by Crammer and Singer is about 12% greater then in Naive Bayes Classifier. In naive Bayes classifier we used tokenizer which avoiding following english stopwords: a, be, had, it, only, she, was, about, because, has, its, of, some, we, after, been, have, last, on, such, were, all, but, he, more, one, than, when, also, by, her, most, or, that, which, an, can, his, mr, other, the, who, any, co, if, mrs, out, their, will, and, corp, in, ms, over, there, with, are, could, inc, mz, s, they, would, as, for, into, no, so, this, up, at, from, is, not, says, to. The results are a bit better then when we used tokenizer which taking into account every word but not so much how we expected We suspect that results in SVR regression are not correct the mean squared error and correlation coefficient seems to be ok but the results file contain the same output for every test data. We spend about 15 hours trying to repair it but without any positive effects. We were trying to scale test and training data, using library with different parameters but each attempt was unsuccessful. We had not enough time to resolve this problem.

## 7. References


http://www.cs.jhu.edu/~mdredze/datasets/sentiment/domain_sentiment_

data.tar.gz
http://www.cs.jhu.edu/~mdredze/
publications/sentiment_acl07.pdf
http://www.csie.ntu.edu.tw/~cjlin/
libshorttext/
http://www.csie.ntu.edu.tw/~cjlin/
libsvm/