UNIVERSITÉ
CÔTE D'AZUR

Sentiment Analysis and Emotions Recognition

Report of TATIA project

Anastasiia KOZLOVA
Meryem BOUFALAH

MASTER INFORMATIQUE

December 2020

Contents

Introduction

Recognizing emotions and sentiments has always been an exciting challenge for scientists because it plays a significant role in many fields of our life: decision making, learning, communication, and others. And emotion detection in human-written texts gives a lot of opportunities to both business and science, since it helps them to figure out the sense of the data and adjust the future decisions respectively.

However, manual analysis of an unstructured text, coming from million sources like client reviews, articles and other documents is time-consuming. Moreover, it is not always an easy task: non-obviousness and ambiguity often lead to variation in different experts' opinions concerning the same text sample. And here comes the importance of automatic emotion and sentiment recognition, which also explains our project choice.

The objective of this report is to present the implementation of the classifiers that solve the problems of sentiment analysis and emotion recognition. This document is organized as follows: in chapter 1 we start with a description of the tasks and their purposes, then in chapter 2 we present the data sets and their pre-processing, afterwards we introduce the approaches (Rule-Based, Support Vector Machines, Naive Bayes) used to build the systems and we finish with comparing the results obtained and giving the conclusions with possible improvement proposals.

1 Description of tasks

1.1 Binary sentiment analysis

Detecting the sentiment of the text consists of saying either it has a positive or negative polarity. The results will be presented in a file with the following structure:

id polarity

Id is the identifier of a text sample, polarity is either 0 (negative) or 1 (positive).

1.2 Ternary sentiment analysis

In ternary sentiment analysis of a text, we detect either it has a positive, neutral, or negative polarity. The results will be presented in a file with the following structure:

id polarity

Id is the identifier of a text sample, polarity is either -1 (negative), 0 (neutral) or 1 (positive).

1.3 Emotion recognition

The purpose is to determine whether a given short text sequence contains the expression of six basic emotions. The results will be presented in a file with the following structure:

id anger disgust fear joy sadness surprise

Id is the identifier of a text sample, the rest six parameters are binary values one for each emotion, where 0 signifies the absence of corresponding emotion and 1 its presence.

2 Conception and Implementation

2.1 Introduction

Different algorithms can be used for sentiment analysis and emotion recognition of a text:

1. Rule-based: These systems operate automatically using a set of manually developed rules.
2. Automatic: systems rely on machine learning techniques to learn from data.
3. Hybrid systems combine both rule-based and automatic approaches.

In our case, we decided to adopt the rule-based and the automatic approaches in order to set up different classification systems and compare their results afterwards. We will evaluate the results of our classifiers using classification reports with some key metrics as Precision, Recall, F1-score and Accuracy.

2.2 Data sets

2.2.1 Initial data sets

For our classifiers we used several data sets:

1. [1]**Positive and negative words lists:** two files, one with positive words and other with negative ones
2. [2]**Emotion lists:** six files, each containing the words that express one of the following emotions: anger, disgust, fear, joy, sadness and surprise.
3. [2]**Training and testing data sets:** one data set for training and other one for testing, each contains a file with text samples and two files that contain the results.

   The first result file contains sentiment analysis in the following format:

   id result

   Each result value is a score in the [-100, 100] range where -100 is maximum negative and 100 is maximum positive.

   The second one gives the emotion recognition in the following format:

   id anger disgust fear joy sadness surprise

   Each emotion value is a score in the [0, 100] range where 0 is absence and 100 is maximum presence of the emotion.

2.2.2 Modifications of data sets

1. **Result files transformation**

   As our system classifies the samples without giving scores for each feature (emotion or sentiment), we have to adopt the result files from initial training and testing data sets so that they return binary results. Therefore, we applied the following modifications to each result value:

   Binary sentiment analysis

   - If value is more or equals 0, we consider that is belongs to class 1 **(positive)**
   - If the value is lower than 0, we consider that it belongs to class 0 **(negative)**

Ternary sentiment analysis

Firstly, we fix the threshold for sentiments. The value we've come to after experiments is 30.

- If value is in range [threshold, 100], we consider that it belongs to class 1 **(positive)**.
- If value is in range (-threshold, threshold), we consider that it belongs to class 0 **(neutral)**.
- If value is in range [-100, -threshold], we consider that it belongs to class -1 **(negative)**.

Emotion recognition

Firstly, we fixed the threshold for emotions. The value we've come to after experiments is 20.

- If value is more or equals the threshold, we consider that it belongs to class 1 (emotion is **present**)
- If value is less than the threshold, we consider that it belongs to class 0 (emotion is **absent**).

2. **Data sets for rule-based system** For the rule-based systems the size of data sets has a large impact on the final accuracy. Our initial data sets with emotions and sentiments ended up to be insufficient for good prediction. Accordingly, we applied several techniques to increase them:

    - **Manual processing of the training data set**

      Despite the fact that our rule-based system does not imply training, we processed the training data manually and added the words to the corresponding data sets.
    - **Adding the synonyms**

      For all the words in data sets we also added their synonyms in order to improve our predictions.
    - **Automatic polarity check**

      We used python libraries to check automatically the polarity of the words in the sentiment files to make sure that they are in right places and reorganize them if necessary.

2.3 Pre-processing

The pre-processing is essential to make the data workable and to provide an opportunity to process the matching with the data sets. Therefore, we have implemented following steps:

1. **Letter casing**: We converted all letters to lower case.
2. **Noise removal**: We removed all punctuation signs and special symbols.
3. **Tokenizing**: We transformed each text sample to the set of tokens (words separated by spaces).
4. **Lemmatization**: We reduced each given word to its root word.
5. **Stop-words removal**: We removed the stop-words which do not contribute to the classification.
6. **Term Frequency – Inverse Document Frequency (TF-IDF) vectorization**: For classifiers with learning we transformed each token into numerical feature vectors.

We decided to use lemmatization rather than stemming because our data sets consist of real words. In this way, lemmatization will ensure more accurate results and will reduce errors.

2.4 Rule-Based system

2.4.1 Algorithm

Our rule-based algorithm consists of three parts: binary sentiment analysis, ternary sentiment analysis and emotion recognition. In the beginning we already have the data sets of words corresponding to each of the six emotions and of polarized words (negative and positive) defined. In order to implement the rule-based system we did the following steps:

1. **For each text sample in test data:**
   (a) **Sentiment analysis**
      i. **Counting the number of words having the sentiment**
      For sentiment analysis, we search the current token in sets of positive and negative words. If the token is present in the set, we increase the counter for the corresponding sentiment. As a result, for each text sample, we have two numbers with a total of positive and a total of negative words.
      ii. **Assigning label for binary sentiment analysis**
         - If the number of negative words is greater than or equals to the number of positive words, then the label is 0 **(negative)**
         - If the number of negative words is less than the number of positive words, then the label is 1 **(positive)**
      iii. **Assigning label for ternary sentiment analysis**
         - If the number of negative words is greater than the number of positive words, then the label is -1 **(negative)**
         - If the number of negative words is less than the number of positive words, then the label is 1 **(positive)**
         - If the number of negative words equals to the number of positive words, then the label is 0 **(neutral)**
   (b) **Emotion recognition**
      i. **Counting the number of words expressing the emotion**
      For emotion recognition, we search the current token in the sets corresponding to each emotion. If the token is present in the set, then we increase the counter for the corresponding emotion. As a result, for each text sample, we have six numbers, each of which indicates the total number of the words expressing a certain emotion.
      ii. **Assigning the emotion labels to the sample**
      In order to assign the resulting emotion labels, we introduced the **threshold**. It indicates the minimum percentage of a certain emotion in relation to the total number of the words in a text sample which will lead to the conclusion that this emotion is present. For example, if we set the threshold to 0.1 (10%) we can assign the label of the presence of certain emotion only if it occurs in not less than 10% words of the text sample. Accordingly,
         - If the number of words expressing current emotion is more than threshold% of all the words of the text sample, then label is 1 **(emotion is present)**
         - If the number of words expressing current emotion is less than threshold% of all the words of the text sample, then label is 0 **(emotion is absent)**
2. **Writing results to files**

2.4.2 Experiments and results

In order to build the classification reports and evaluate the results we compare our classifier's result files with the results from our test data sets after applying the modifications described in 2.2.2 Modifications of data sets.

In Table 1 there is a classification report for binary sentiment analysis:

Table 1: Rule-based classification report for binary sentiment analysis

Class	Precision	Recall	F1-score	Support
0	0.64	0.87	0.73	526
1	0.75	0.45	0.57	474

In Table 2 there is a classification report for binary sentiment analysis:

Table 2: Rule-based classification report for ternary sentiment analysis

Class	Precision	Recall	F1-score	Support
-1	0.58	0.62	0.60	377
0	0.36	0.35	0.36	321
1	0.55	0.52	0.53	302

We also evaluated the results for each emotion separately in Table 3:

Table 3: Rule-based classification report of emotion recognition

Emotion	Class	Precision	Recall	F1-score	Support	Accuracy
Anger	**0**	0.84	0.98	0.90	834	83%
	1	0.24	0.03	0.05	166	
Disgust	**0**	0.94	0.98	0.96	939	93%
	1	0.00	0.00	0.00	61	
Fear	**0**	0.71	0.98	0.82	713	71%
	1	0.24	0.02	0.03	287	
Joy	**0**	0.61	0.98	0.79	605	61%
	1	0.48	0.03	0.05	395	
Sadness	**0**	0.66	0.98	0.79	662	66%
	1	0.43	0.03	0.05	338	
Surprise	**0**	0.63	0.98	0.76	622	62%
	1	0.52	0.03	0.05	378	

The overall results including the number of errors and accuracy for three classification types are presented in Table 4:

Table 4: Results of rule-based system

Classification type	Number of test samples	Number of errors	Accuracy
Binary sentiment	1000	330	67.0%
Ternary sentiment	1000	496	50.4%
Emotions (six classes)	6000	1671	72.5%

In many existing rule-based systems of emotion recognition, the threshold is not used, the values are compared strictly. However, we decided to use the threshold for obtain more accurate results and remove "noise". For example, if a text sample consists of 40 words, 30 of them express anger and two of them express joy, we do not consider that this sample express joy. Indeed, in most of the cases if we manually analyse the samples considering the context, the presence of the words of a certain emotion is irrelevant if their number is relatively small.

We experimented with the value of the threshold for emotion recognition, the most optimal results are obtained if we set it for 0,2 (20%).

However, when we used the threshold for sentiment analysis, our system showed less accurate results. For this reason we decided to compare strictly the numbers of positive and negative words and consider the sample neutral only if the numbers are equal. The results were also improved after increasing data sets with the techniques described in 2.2.2 Modifications of data sets

We can also see the correlation between the data distribution and the accuracy. The more equal distribution we have, the better precision we get for each class and the whole classifier. In particular, for the "disgust" class we have 94% accuracy. Nevertheless, the precision for 1-class is 0.03 due to a small number of samples comparing to 0-class.

2.5 Automatic learning

2.5.1 Support Vector Classifiers

Support-vector machines proved to be efficient when applied to natural language processing problems. In particular, they give good results for multi-class classification if the data sets are not too big. That's why we decided to use the SVC variations from the python machine learning toolkit **scikit**.

The idea of the Support Vector Machine is to build the hyper plane which separates the data into different classes. This hyper plane must have the maximum distance between points of both classes. This distance is also called margin. Therefore, We tried out different implementations with various kernels (type of hyper plane used to separate the data):

1. **SVC with radial basis function (RBF) kernel**
2. **SVC with polynomial kernel**
3. **SVC with sigmoid kernel**
4. **SVC with linear kernel**
5. **LinearSVC (The different implementation of SVM with linear kernel)**

2.5.2 Naive Bayes

Naive Bayes is a classifier that use probability theory and Bayes' Theorem to predict the label of a text. In our case we used it to determinate both sentiment and emotions of a text sample, which means that it calculate the probability of each label for a given text, and then output the label with the highest one.

1. **Training**

   We use the training data sets to calculate P(word|Label) and P(Label).

   (a) **Sentiment analysis**

   We calculate:

   P(x|Positive) and P(x|Negative) for binary sentiment analysis

   P(x|Positive), P(x|Neutral) and P(x|Negative) for ternary sentiment analysis

   where x is a word in the text sample.

   So, for example, to calculate P(x|Positive) we need to go through all the text samples and calculate the probability where the word x is in the text sample and this text sample is positive. Then we repeat these actions for each label.

   (b) **Emotion recognition**:

   For each emotion we calculate **P(x|Emotion) , P(x|Not emotion)**

   Accordingly, for example to calculate P(x|Anger) we need to go through all the text samples and calculate the probability where the word x is in the text sample and this text sample expresses anger. Likewise, to calculate P(x|Not Anger) we calculate the probability where the word x is in the text sample and the text sample doesn't express Anger.

2. **The principle of Naive Bayes classifier:**

   We suppose having a text sample X, the steps are the following:

   (a) **The binary sentiment analysis**

   Calculate the probabilities that a text sample X is positive and negative. As a result, we assign the label with the largest probability.

   (b) **The ternary sentiment analysis**

   Calculate the probabilities that a text sample X is positive, negative and neutral. As a result, we assign the label with the largest probability.

   (c) **Emotion recognition**

   For each emotion, calculate the probabilities that a text sample X expresses an emotion and doesn't express this emotion. As a result, for each emotion class we assign the label with the largest probability.

   To calculate for example P(Positive| X), P(Negative| X), P(Neutral| X), we use the Bayes' Theorem:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

   Since we're just trying to find out which label has a bigger probability, we can discard the divisor which is the same for all three probabilities and just compare:

P(X|Positive) x P(Positive)
and
P(X|Negative) x P(Negative)
and
P(X|Neutral) x P(Neutral)

We have

$$X = [x_1, x_2, ..., x_n]$$

where X is a text sample and x are the words of X.

Accordingly, having each P(x|Positive) and P(Positive) obtained after training, to calculate P(X|Positive), we can proceed as follow:

$$P(X|Positive) = P([x_1, x_2, ..., x_n]|Positive)$$

$$P(X|Positive) = P(x_1|Positive)XP(x_2|Positive)X.....XP(x_n|Positive)$$

2.5.3 Experiments and results

After introducing the classifiers, we experimented with parameters of TfidfVectorizer. If the min_df (minimal frequency for the token to use it for learning) is less than 3, the precisions for 1-class for each classification deteriorate. So we tuned it to be 3 as it gives the best score. We also tuned the vectorizer to extract 3-grams as it also improves the results.

The final results of classifiers with learning are presented below:

Table 5: Results of SVC with radial basis function (RBF) kernel

Classification type	Number of test samples	Number of errors	Accuracy
Binary sentiment	1000	446	55.4%
Ternary sentiment	1000	603	39.7%
Emotions (six classes)	6000	1705	71.58%

Table 6: Results of SVC with polynomial kernel

Classification type	Number of test samples	Number of errors	Accuracy
Binary sentiment	1000	447	55.3%
Ternary sentiment	1000	622	37.8%
Emotions (six classes)	6000	1741	70.98%

Table 7: Results of SVC with sigmoid kernel

Classification type	Number of test samples	Number of errors	Accuracy
Binary sentiment	1000	450	55.0%
Ternary sentiment	1000	616	38.4%
Emotions (six classes)	6000	1707	71.55%

Table 8: Results of SVC with linear kernel

Classification type	Number of test samples	Number of errors	Accuracy
Binary sentiment	1000	445	55.5%
Ternary sentiment	1000	610	39.0%
Emotions (six classes)	6000	1703	71.62%

Table 9: Results of LinearSVC

Classification type	Number of test samples	Number of errors	Accuracy
Binary sentiment	1000	450	55.0%
Ternary sentiment	1000	600	40.0%
Emotions (six classes)	6000	1749	70.85%

Table 10: Results of Naive Bayes

Classification type	Number of test samples	Number of errors	Accuracy
Binary sentiment	1000	418	58.2%
Ternary sentiment	1000	577	42.3%
Emotions (six classes)	6000	1757	70.71%

As a consequence, Naive Bayes showed the best result for binary sentiment analysis, LinearSVC worked the best for ternary sentiment and SVC with radial basis function kernel gave the highest score for all the emotions.

2.6 Comparison of the results

The accuracy results for all the classifiers implemented in this project are presented in table 11:

Table 11: Accuracy results for different classifiers with learning

Classifiers	Rule-based	SVC RBF	SVC poly	SVC sigmoid	SVC linear	LinearSVC	Naive Bayes
Binary sent.	67.0%	55.4%	55.3%	55%	55%	55%	58.2%
Ternary sent.	50.4%	39.7%	37.8%	38.4%	39%	40%	42.3
Emotions	72.5%	71.6%	70.9%	71.55%	71.61%	70.85%	70.71%
Anger	83%	83%	82%	83%	83%	82%	80%
Disgust	93%	90%	89%	89%	90%	87%	89%
Fear	71%	72%	71%	73%	72%	72%	71%
Joy	61%	57%	56%	57%	58%	56%	58%
Sadness	66%	67%	66%	66%	67%	67%	66%
Surprise	62%	61%	62%	60%	60%	62%	61%

As we can see, our rule-based system showed the best results for all the classification types. This must be due to the fact that our data sets are relatively small to build the learning systems.

2.7 Improvements proposal

One of the factors affecting the results is that data sets we used are not initially in the form we need. So the results vary depending on which thresholds we choose to transform scores to binary results.

In able to achieve better accuracy we could provide more equal distribution by classes. For example, in several emotion classifiers implemented, the precision rates for 0-class are much better than the precision rates for 1-class due to the fact that the number of 0-samples exceeds considerably the number of 1-samples in our data sets.

Overall, our data sets are relatively small, so we could increase them in order to improve all the classifiers in this project. On the other hand, if the number of vectors is large, Support Vector Machines will require a lot of storage and performances, so we might have to use a different model as linear regression.

References

[1] Kaggle. *Positive and negative word lists*
<https://www.kaggle.com/harshaiitj08/positive-and-negative-words/>

[2] Carlo Strapparava and Rada Mihalcea. *Affective Text*
<https://web.eecs.umich.edu/~mihalcea/affectivetext/>

[3] Bouwe Ceunen. *Natural Language Understanding With SVM's*
<https://medium.com/axons/natural-language-understanding-with-svms-87f1b8a63ea0>

[4] monkey learn. *Sentiment Analysis: A Definitive Guide*
<https://monkeylearn.com/sentiment-analysis>

[5] Gaurav Singhal. *Building a Twitter Sentiment Analysis in Python*
<https://www.pluralsight.com/guides/building-a-twitter-sentiment-analysis-in-python>

[6] monkeylearn. *A practical explanation of a Naive Bayes classifier*
<https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/>

[7] Weka 3. *Machine Learning without Programming*
<https://www.cs.waikato.ac.nz/ml/weka/>

[8] Karthikeya Boyini. *How to get synonyms/antonyms from NLTK WordNet in Python*
<https://www.tutorialspoint.com/how-to-get-synonyms-antonyms-from-nltk-wordnet-in-python>

[9] Pragyan Subedi. *Machine Learning — The different ways to evaluate your Classification models and choose the best one!*
<https://medium.com/kharpann/machine-learning-the-different-ways-to-evaluate-your-classifi

[10] Naresh Kumar. *Advantages and Disadvantages of SVM (Support Vector Machine) in Machine Learning*
<http://theprofessionalspoint.blogspot.com/2019/03/advantages-and-disadvantages-of-svm.html>

[11] scikit learn. *Support vector machines*
<https://scikit-learn.org/stable/modules/svm.html>

[12] Krish Naik. *Tutorial 48- Naive Bayes' Classifier Indepth Intuition- Machine Learning*
<https://www.youtube.com/watch?v=jS1CKhALUBQ>