# Machine Learning for the Sentiment Coding of Flemish Tweets

Organisation: Statistiek Vlaanderen

Author(s): Marc Callens, Michael Reusens

Date: 29/04/2020

Version: 2

## 1. Background and why and how this study was initiated

This pilot study has been set up to test the use of big data sources like Twitter to measure the sentiment of a population (e.g. perceived quality of life, beyond GDP). Can the combined use of big data, text mining and machine learning provide a valuable alternative for surveys?

The study provides a use case for Flemish-language tweets in Flanders (Belgium). Other cases for e.g. Poland (Polish) and Mexico (Spanish) are already available in the field of official statistics (https://github.com/jmaslankowski/WP7-Population-Life-Satisfaction) (https://www.inegi.org.mx/app/animotuitero/).

The case for Flanders has some special issues such as the relatively low penetration of Twitter in Belgium compared to other countries. The absence of a Flemish language-category in Twitter is another specificity. Even though the sentiment coding of tweets is the focus of this report, these specificities make the Flemish case interesting to try.

The main goal for the project is to ultimately set up the production of quality of life statistics for Flanders on a continuous base. The current quality of life statistics are produced via surveys, which results in an infrequently available statistics. Using social media data, daily (and even hourly) measures of quality of life statistics can be calculated and presented on the website of Statistics Flanders. An essential step in the production process for these statistics is to code the incoming tweets into sentiment categories (e.g. positive, negative). As manual coding of continuously incoming tweets is not feasible in practice, it is necessary to rely on mechanical methods such as Machine Learning (ML) to perform this classification task in a satisfactory way.

Although this paper concentrates on the ML part of the problem, we will also touch upon some Natural Language Processing (NLP) aspects involved. ML are the set of algorithms that will output a sentiment classification for a given tweet. NLP are the practices and techniques used to convert text to a format that can be used by ML algorithms.

The use case for this experiment is general and binary. General means that all available tweets are used and not a specific selection representing a specific topic (e.g. the keyword “Vlaanderen”). On the other hand, the binary case (positive and negative tweets) is a simplification of use cases with more categories (e.g. the use case of Poland has 6 categories).

A schematic overview of the approach followed in the experiments can be seen below in Figure 1. What follows is a high level overview of each of the steps in the schema. The next sections provide a more in depth discussion for each step. The code for each step can be found on our GitHub referenced in Section 5 of this report.

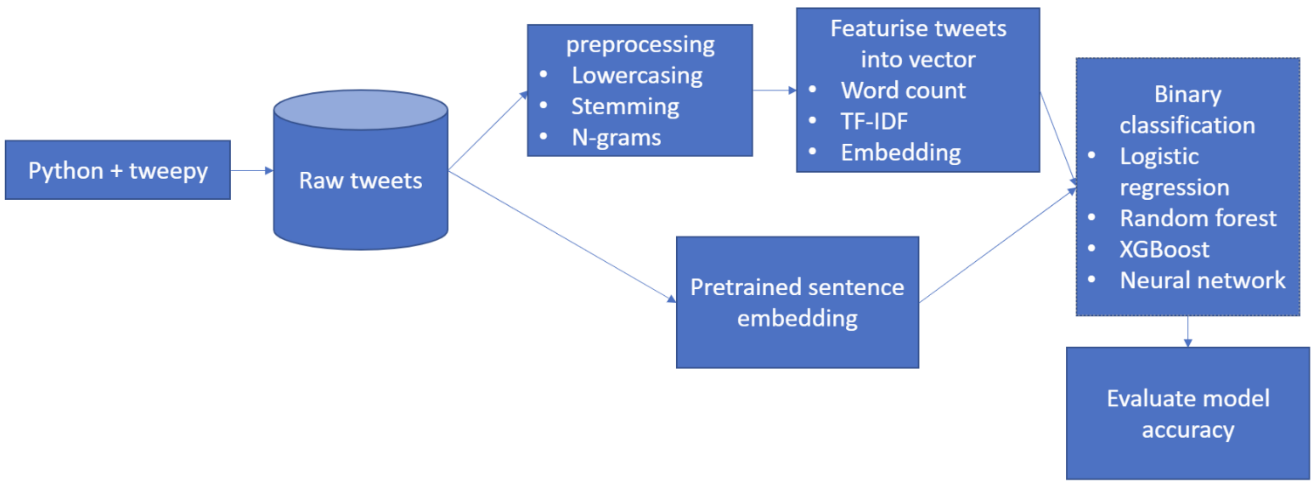


**Figure 1: schematic overview of approach**

Before obtaining Twitter data, one has to register for a Twitter account, get API keys and create a new app on the Twitter website. Actual collection of the data is performed by calling the rest Twitter API from specific-written Python code. The second step is to normalize the text (removing stop words, lemmatization, removing special characters, etc..) and perform feature extraction (derivation of meaningful features from the raw data that can be used by a machine learning model to make predictions).

The ML part of the process starts with coding a training data set. This coded training set serves as examples of positive and negative tweets, from which the (supervised) ML program then tries to learn patterns. The result is a classification model. This model is expected to predict outcomes (sentiment) for new tweets in the future. In this paper we evaluate several machine learning models: penalized logistic regression, random forest, gradient boosting trees and multi-layered perceptrons.

An important aspect of the process is the evaluation of the model: how well are new tweets predicted by the model? In this paper we use different evaluation measures (accuracy, recall precision,...) and apply them to different ML algorithms to compare the performance of these algorithms.

This project was started as a first data science experiment for Statistics Flanders. It was started by our data strategist in collaboration with a data science consultant. Midway through the project a data scientist joined Statistics Flanders and took over the technical project work.

## 2. Data

### 2.1 Input Data

This is the first project for Statistics Flanders in which Twitter data is used.

Using the tweepy Python library, we requested a maximum of 20.000 dutch tweets twice.  Once with a positive search term, requiring the resulting tweets to contain at least one of the following happy emoticons ":), :-), :D, :-D, : )", and once with a negative search term, requiring the resulting tweets to contain at least one of the following negative emoticons ":(, : (, :'(". The library immediately allows to request only original tweets (no re-tweets).

This method of data gathering is based on some assumptions that require further investigation. First, we only filter based on language (dutch), and not based on location. This is because filtering both on language and location (which requires the tweets to contain a geo-tag) results in significantly fewer retrieved tweets containing the smiley-search strings. We do not expect this to strongly influence our experimental results. However, this is something to keep in mind when moving closer to a production statistic, as there are some (small) differences between the dutch language as spoken in Flanders compared to dutch as spoken in the Netherlands. Once we are confident with the performance of the sentiment detection model, we will only select dutch tweets with a Belgian (ideally Flanders, if this is possible) geotag. Secondly, because we have not yet set up the methodology to manually annotate tweets with their sentiment, we only retrieve tweets with happy/sad smileys. We assume that the presence of a happy smiley in a tweet indicates positive sentiment, and the presence of a sad smiley indicates negative sentiment. In the near future, we plan to create a "gold standard" dataset, consisting of tweets that are manually coded by a panel of experts.

The resulting dataset contains 19.000 positive and 7.000 negative tweets. For model evaluation, it is key to keep in mind that our dataset has a class imbalance of 73% (positive) vs 27% (negative).

## 2.2 Data Preparation

The following pre-processing operations were performed on the tweets:

- Lowercase all letters (e.g. "The capital of Belgium is Brussels" --> "the capital of belgium is brussels")
- Stemming (e.g. "playing children" --> "play child"
- n-gramming (up to 3-gramming). This is the combination of separate words that denote one concept (e.g. "the justice system of the european union" --> "the justice_system of the european_union"). This helps the machine learning model understand text in terms of concepts, rather than individual words that on their own do not always carry the full meaning.
- Stopword removal ("the cat is an animal" --> "cat is animal")

## 2.3 Feature Selection

We experimented with the following strategies to convert a tweet to a vector of numbers on which a classifier can be trained:

- Count vectorization: vector with the occurrence (count) of each word in the vocabulary. The vocabulary is the set of all words that occur at least once in the dataset of tweets.
- Tf-IDF vectorization: vector with for each word in the vocabulary the occurrence divided by the total occurrence of that word in all tweets. A high value indicates that this word occurs often in this tweet and does not normally appear in tweets.

- Autoencoder neural network embedding: a dense representation of the count vector in k numbers (similar to dimensionality reduction techniques such as Principal Component Analysis (PCA) or Singular Value Decomposition (SVD)) trained on our dataset.
- Pretrained neural network embedding: a pretrained sentence embedding model: https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/3

### 2.4 Output data

We split the dataset randomly into a training set (75% of the data) and test set (the remaining 25%). The training set is used for hyperparameter optimization and model training. The test set is used for model evaluation.

## 3. Machine Learning Solution

### 3.1 Models tried

Several machine learning models were evaluated to learn the mapping between feature vectors and sentiment:

- Penalized logistic regression
- Random forests
- Gradient boosting trees
- Multi-layered perceptron

The hyperparameter tuning for each model was done using grid search using 5-fold cross validation for each parameter combination.

### 3.2 Model(s) finally selected and the criterion

The best observed accuracy of featurization-classifier combinations was 81%.

We did not observe a big difference between different classifiers: random forest, logistic regression, gradient boosting trees and multi-layered perceptrons resulted in a very similar performance.

Count-, TF-IDF and the pretrained embedding model resulted in close to identical performance (all close to 81%).

The self-trained embedding model seemed to converge to around 68% accuracy, which is even worse than the majority voting baseline (a naïve system that always predicts the most likely outcome. In our case we have around 73% positive tweets, so by always predicting positive you could have a system that is 73% accurate without learning anything). This could be due to the lack of training data. We also did not optimize the neural network embedding structure, which could lead to better performance.

### 3.3 Hardware used

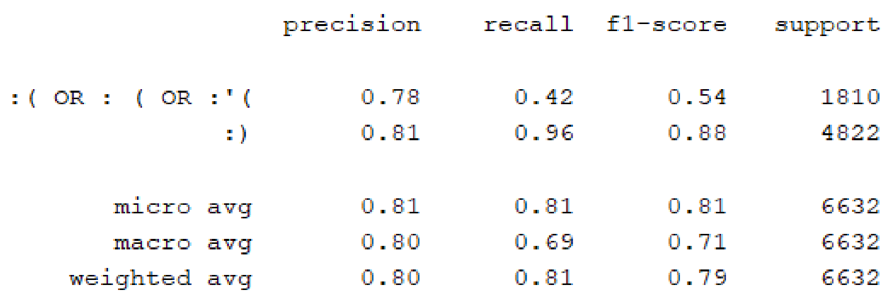
The code was partly run on a personal computer (Intel i7 6700K @4GHz, 16GB RAM), and partly on Google's Collaboratory notebooks. No GPU/TPU acceleration was used. GPU/TPU's could be useful to speed up neural network training time during further experiments.

### 3.4 Runtime to train the model

Most models took a lot less than 1 hour to train, the basic models such as count vectorization in combination with logistic regression even less than 1 minute. When optimizing parameters using grid search, the size of the parameter space will directly impact the total runtime of the program.

## 4. Results

Below is the evaluation report of the count vectorization – logistic regression model with 81% accuracy. As discussed earlier, other models achieved similar performance. This evaluation report gives us some interesting insights besides pure accuracy. The precision and recall metrics show us that the model performs better on positive tweets than on negative tweets. This could be because of the larger training set available for positive tweets.



```
                  precision    recall  f1-score   support

:( OR : ( OR :'(       0.78      0.42      0.54      1810
              :)       0.81      0.96      0.88      4822

       micro avg       0.81      0.81      0.81      6632
       macro avg       0.80      0.69      0.71      6632
    weighted avg       0.80      0.81      0.79      6632
```

**Table 1 : Performance of count-vector + penalised logistic regression on test set**

## 5. Code/programming language

All code is written in Python 3, and can be found on the following GitHub:
https://github.com/mireusen/hlmos-statistiek-vlaanderen-twitter

## 6. Is it a proof of concept or is it already used in production?

This paper describes work that is still in the proof of concept phase. The results described here are valuable to our organization as they show that decent accuracy can be achieved with relatively simple models. This gives us confidence that, given further experiments, this accuracy can be further improved in order to obtain a better statistic in the end product. It is still an open question how good the ML coding has to be in order to be 'production ready'. 100% accuracy will never be achieved. There are however quite some obvious improvements that we can make to our sentiment analysis method (see Section 10, Next Steps).

### 6.1 Is there already a roadmap/service journey available how to implement this?

Before implementing twitter sentiment in Flanders as an official statistic, many challenges still need to be addressed. Concrete challenges that we will tackle are described in the 'Next steps' section below.

### 6.2 Who are the stakeholders?

The following parties are stakeholders in these experiments and/or the resulting statistic:

- Us, as NSI who's ambition it is to publish informative statistics about Flanders.
- The consumer of these statistics: citizens, academia, industry and government.
- Other NSI's/expertise on this topic to collaborate with

### 6.3 Fall Back

There is no fall-back plan in place, besides simply not publishing this statistic. Manual coding would be highly impractical.

### 6.4 Robustness

We believe for this type of model to be useful in the context of official statistics transparency is key. Therefore, we will publish and document the full analysis pipeline, so the process is open for scrutiny.

Furthermore, once we are satisfied with the model's performance, we will ask other NSI's with experience in analysing tweets to review our methods. By collaborating with other NSI's and experts we hope to thoroughly challenge and fine-tune our assumptions and chosen methods.

## 7. Conclusions and lessons learned

- Valuable experience working with the Twitter search API.
- Relatively simple approach performs quite well
- Huge space of techniques to try out and evaluate. Our experiments are far from over and we believe that we can further improve model accuracy with other featurization and classification techniques.

## 8. Potential organisation risk if ML solution not implemented

Opportunity cost of not offering this statistic. Innovation in official statistics is one of our strategic goals, so we must be ambitious in our efforts to fully evaluate the opportunities that these types of projects offer.

## 9. Has there been collaboration with other NSIs, universities, etc?

So far there have been no formal collaborations with other NSI's for this project specifically. Later on, we see it as part of our robustness checks to let our model be challenged by other NSI's that have experience analysing Twitter data.

## 10. Next Steps

We plan to put further effort in developing this project into a statistic that can be published. Concretely, we see the following next steps:

- Other featurization techniques
    - Improve self-trained embedding with more data and optimization of embedding structure
    - Use other pretrained embedders (or pretrained sentiment classifiers), with focus on dutch social media posts. There are several available online both commercial and open source, such as https://github.com/wimulkeman/dutch-sentiment-analysis, https://github.com/wietsedv/bertje/blob/master/README.md
- Gather more data, currently only 1 week back at time of querying twitter API. More data will allow for the optimization of more complex models with a high number of trainable parameters (such as deep learning models).
- Development of a program that lets users manually annotate tweets. This will alleviate us of the dependency (and corresponding assumptions and biases) on tweets containing smileys. Tweets that have been annotated by different users can be used to create a "gold standard" dataset that can be used to track the quality of the system.
- Set up a production IT infrastructure that allows for the continuous retrieval, analysis and publishing of Twitter sentiment statistics.
- Look into specific twitter sentiment use cases
- Twitter sentiment as a replacement/complement to the existing subjective wellbeing statistic
- Twitter sentiment on specific topics, such as the sentiment on "Vlaanderen" around the world.