Analyzing Twitter users' feelings about the takeover of Twitter by Elon Musk using 75000 tweets

Natural Language Processing Project

Amir Hajiabadi: amir.hajiabadi@studio.unibo.it

Molin Zhang: molin.zhang@studio.unibo.it

Jialing Li: jialing.li@studio.unibo.it

Academic year 2021-2022

Abstract

Sentiment analysis is the automated process of identifying and classifying subjective information in text data. This might be an opinion, a judgment, or a feeling about a particular topic or product feature.
The most common type of sentiment analysis is 'polarity detection' and involves classifying statements as Positive, Negative, or Neutral.
Sentiment analysis uses Natural Language Processing (NLP) to make sense of human language and machine learning to deliver accurate results automatically.

Elon Musk is a famous business magnate and investor.
He recently expressed his desire to buy Twitter, met with many reactions. It has been a hot topic since then, and we used a large dataset to analyze people's feelings about it.

Steps:
Performing sentiment analysis on Twitter data involves five steps:

1. Gather relevant Tweets
2. Clean your data using pre-processing techniques
3. Visualizing the data
4. Create a sentiment analysis machine learning model
5. Analyze your Twitter data using your sentiment analysis model

Contents

Problem presentation

In this project, we are supposed to predict the sentiment of a given tweet and predict if it is positive, negative, or neutral.

Setup

The project is developed on Colaboratory Pro from Google Research, which provides the following hardware:

- Intel® Xeon™ CPU @ 2.30GHz;
- NVIDIA® Tesla P100-PCIE™ 16GB;
- DDR5 32 GB RAM.

The main libraries employed to tackle the problem are:

- Pandas, a data analysis and manipulation tool.
- NumPy, a package for scientific computing with Python.
- matplotlib.cm, Built-in colormaps, colormap handling utilities.
- matplotlib.pyplot is a state-based interface to matplotlib. It provides an implicit, MATLAB-like way of plotting. It also opens figures on your screen and acts as the figure GUI manager.
- rcParams, dynamically change the default rc (runtime configuration) settings in a python script or interactively from the python shell. All rc settings are stored in a dictionary-like variable called matplotlib.rcParams, which is global to the matplotlib package.
- %matplotlib inline, we can use the magic function %matplotlib inline to enable the inline plotting, where the plots/graphs will be displayed just below the cell where your plotting commands are written. It provides interactivity with the backend in the frontends like the jupyter notebook.
- Warnings messages are typically issued when it is helpful to alert the user of some condition in a program, where that condition (usually) doesn't warrant raising an exception and terminating the program. For example, one might want to issue a warning when a program uses an obsolete module.
- sys, lets us access system-specific parameters and functions. We used sys.path.append() to import our external modules.
- re, a regular expression specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression (or if a given regular expression matches a specific string, which comes down to the same thing).

• nltk, or Natural Language Toolkit, is a Python package that you can use for NLP. A lot of the data you could analyze is unstructured data and contains human-readable text.

•string contains a set of useful constants, such as ascii_letters and digits, and the module is often still imported for that reason.

•nlp_utils, Python package providing natural language processing utilities.

•Contractions, a Python library for expanding and creating common English contractions in text. This is very useful for dimensionality reduction by normalizing the text before generating word or character vectors. It performs contraction by simple replacement rules of the commonly used English contractions.

•Counter is a dict subclass for counting hashable objects. It is a collection where elements are stored as dictionary keys, and their counts are stored as dictionary values. Counts are allowed to be any integer value, including zero or negative counts. The Counter class is similar to bags or multisets in other languages.

•Stopwords is a list of lexical stop words in English. These words are ignored during most natural language processing tasks, such as part-of-speech tagging, tokenization, and parsing.

•WordNetLemmatizer, is a large, freely, and publicly available lexical database for the English language aiming to establish structured semantic relationships between words. It offers lemmatization capabilities as well and is one of the earliest and most commonly used lemmatizers.

•WordCloud, is a visual representation of text data. Words are usually single words, and the importance of each is shown with font size or color. Python, fortunately, has a wordcloud library allowing us to build them.

•word_tokenize, a method to split a sentence into tokens or words.

•sent_tokenize, a method to split a document or paragraph into sentences.

•TfidfVectorizer, Convert a collection of raw documents to a matrix of TF-IDF features.

•nltk.stem, remove morphological affixes from words, leaving only the word stem.

•LabelEncoder, Sklearn provides a very efficient tool for encoding the levels of categorical features into numeric values. LabelEncoder encodes labels with a value between 0 and n_classes-1 where n is the number of distinct labels. If a label repeats, it assigns the same value as assigned earlier.

•train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: training data and testing data. With this function, you don't need to divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.

•LogisticRegression, based on a given set of independent variables, is used to estimate discrete values (0 or 1, yes/no, true/false).
•accuracy_score, in multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must exactly match the corresponding set of labels in y_true.
•TensorFlow, an end-to-end open-source platform for machine learning.
•timeit is a method in the Python library to measure the execution time taken by the given code snippet. The Python library runs the code statement 1 million times and provides the minimum time taken from the given code snippets.
•swifter, a package that efficiently applies any function to a pandas dataframe or series in the fastest available manner.
•nltk.sentiment.Vader, VADER (Valence Aware Dictionary for Sentiment Reasoning) is a model used for text sentiment analysis that is sensitive to both polarity (positive/negative) and intensity (strength) of emotion. It is available in the NLTK package and can be applied directly to unlabeled text data.
•SentimentIntensityAnalyzer, takes in a string and returns a dictionary of scores in each of three categories: negative. Neutral. Positive.
•DecisionTreeClassifier, the algorithm uses training data to create rules that a tree structure can represent
•RandomForestClassifier, is a meta estimator that fits several decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
•KNeighborsClassifier: this classifier implements learning based on the k nearest neighbors.
•SVC is a C-support vector classification whose implementation is based on libsvm. This class handles the multiclass support according to the one-vs-one scheme.
•metrics, implements several loss, score, and utility functions to measure classification performance.

Data

The dataset used is from Kaggle, which contains 100,000 tweets in different languages and 38 columns of details about them. We just wanted to work on the tweets in English, so we removed the rest of the rows and columns before. Our final dataset contains 75,000 tweets.

Data Preprocessing

The dataset is not so clean. Therefore, some further cleaning is needed.

We do text normalization, transforming the text into a canonical (standard) form.

Tokenization

Tokenization is when a significant quantity of text is divided into smaller parts called tokens.

Natural language processing is used for building applications such as Text classification, intelligent chatbot, sentimental analysis, language translation, etc. It becomes vital to understand the pattern in the text to achieve the above-stated purpose. These tokens are handy for finding such patterns and are considered a base step for stemming and lemmatization.

Sentence tokenization is the process of splitting text into individual sentences. It does this by looking for the types of textual constructs that confuse the tokenizer and replacing them with single words.

Alphanumeric characters

Alphanumeric characters and decimals have been replaced with characters.

Expanding Contractions

A contraction is a shortened form of a word (or group of words) that omits certain letters or sounds.

Expanding Contractions is a process where words like is not, did not are expanded to is not, did not.

Finally, we store the cleaned data in a dataframe and eliminate the duplicate or empty rows.

Removing additional characters

We can further see that there are still some special characters in the dataframe.

They must be eradicated using the strip function.

Lemmatization of the text

Lemmatization usually refers to doing things properly using a vocabulary and morphological analysis of words, aiming to remove inflectional endings only and return to the base.

Alternatively, the dictionary form of a word is the lemma.
Lemmatization will generate the root form of the inflected words.

Swifter is a package that efficiently applies any function to a pandas dataframe or series in the fastest available manner than the normal apply function.

In the end, we will Remove stopwords from the dataframe.

Visualization

Visualizing the most frequent words in the dataframe will be done using the wordcloud technique.
We have listed the 100 most repeated words throughout the data.
The term 'Twitter' has been repeated 166959 times.

Sentiment Analysis.

VADER belongs to a sentiment analysis type based on lexicons of sentiment-related words.
In this approach, each of the words in the lexicon is rated as to whether it is positive or negative and how positive or negative in many cases. We can see an excerpt from VADER's lexicon, where more positive words have higher positive ratings and more negative words have lower negative ratings.
In addition, a compound score shows the importance of a sentence.

In the end, we converted all polarity scores and sentences into a dataframe.

We arranged the dataset in descending order based on (Compound score) to find the most crucial sentence from the given data.
We did the same for the most positive and negative sentences.

Giving threshold values for classification

The threshold value will categorize if a given sentence is positive, negative, or neutral, and the predictions will be saved into a Target or sentiment column in our dataframe.

Final dataframe

We do not need other columns such as neg(negative), neu(neutral), pos(positive), and compound anymore, so we will remove them to reach the Final dataframe with sentiments.
There are 20853 positive, 37155 negative, and 17531 neutral sentences present in the dataset.

Then, before applying machine learning models, we need to Label Encoding the target column. We do that by Label encoder, which converts categorical values into numeric ones.

Vectorizing training data

Applying Tf-Idf vectorizer on the Text column converts text to a feature vector that can be used to input our estimator. Then we split the data.

Logistic Regression

We fit the model to our data and, after some predictions, calculate the accuracy.

Model accuracy on train is: 0.832933428207377

Model accuracy on test is: 0.757280910775748

Decision Tree Classifier

We fit the model to our data and, after some predictions, calculate the accuracy.

Moreover, the Confusion matrix, the number of wrong Predictions made, and Kappa Score.

```
--------------------------------------------------
Model accuracy on train is:  0.9995532094454833
Model accuracy on test is:  0.9994704792163093
--------------------------------------------------
confusion_matrix train is:
[[29735     0     0]
 [    9 13986     0]
 [    6    12 16683]]
confusion_matrix test is:
[[7420    0    0]
 [   1 3535    0]
 [   2    5 4145]]
-------------------------------------------------
Wrong predictions out of total
8 / 15108
-------------------------------------------------
KappaScore is:  0.9991574222978195
```

RandomForestClassifier

We fit the model to our data and, after some predictions, calculate the accuracy.

Moreover, the Confusion matrix and the number of wrong Predictions made.

```
-------------------------------------------------
Model accuracy on train is:  0.999619400638745
Model accuracy on test is:  0.6894360603653693
-------------------------------------------------
confusion_matrix train is:
[[29733     2     0]
 [    5 13987     3]
 [    5     8 16688]]
confusion_matrix test is:
[[7041   73 306]
 [2291 737 508]
 [1268 246 2638]]
-------------------------------------------------
Wrong predictions out of total
4692 / 15108
```

KNeighborsClassifier

We fit the model to our data and, after some predictions, calculate the accuracy.

Moreover, the Confusion matrix and the number of wrong Predictions made.

```
--------------------------------------------------
Model accuracy on train is:  0.6922605947278715
Model accuracy on test is:  0.5873709293089754
--------------------------------------------------
confusion_matrix train is:
[[27687 1244   804]
 [ 7158 5964   873]
 [ 6289 2229 8183]]
confusion_matrix test is:
[[6367 668 385]
 [2224 960 352]
 [1901 704 1547]]
--------------------------------------------------
Wrong predictions out of total
6234 / 15108
```

Support Vector Machine

We fit the model to our data and, after some predictions, calculate the accuracy.

Moreover, the Confusion matrix and the number of wrong Predictions made.

```
--------------------------------------------------
Model accuracy on train is:  0.8686270291737684
Model accuracy on test is:  0.7839555202541699
--------------------------------------------------
confusion_matrix train is:
[[28445   779   511]
 [ 2793 9801 1401]
 [ 1317 1138 14246]]
confusion_matrix test is:
[[6868 361 191]
 [1126 1877 533]
 [ 447 606 3099]]
```

```
--------------------------------------------------
Wrong predictions out of total
3264 / 15108
```

Conclusion:

Decision Tree Classifier with 8 wrong predictions out of 15108 predictions (99% accuracy) and Support Vector Machine with 3264 wrong predictions out of 15108 predictions (0.78% accuracy) had the best performance.

limitations and possible improvements:

By spending more time, we can use more models and compare the accuracy or, for example, XGBoost to improve the performance.
We can work on different languages or create a web application where the user can enter a sentence to receive the sentiment analysis related to it.

References

[1] Kaggle Dataset:

https://www.kaggle.com/datasets/prabowoyogawicaksana/elon-musk-twitter-acquisition-opinion

[2] VADER Sentiment Analysis GitHub:

https://github.com/cjhutto/vaderSentiment

[3] VADER Sentiment Analysis explanation:

https://medium.com/@piocalderon/vader-sentiment-analysis-explained-f1c4f9101cd9

[4] Sarkar, D. Text Analytics with Python. Karnataka, India: Apress.

[5] Joshi, P. Python Machine Learning Cookbook. Birmingham, UK: Packt Publishing.

[6] Bird, S. et al. Natural Language Processing with Python. California, CA: O'Reilly Media.

[7] Shahzad Qaiser and Ramsha Ali. "Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents". In: International Journal of Computer Applications 181 (July 2018). Doi: https://doi.org/10.5120/ijca2018917395

[8] Pandas Documentation: https://pandas.pydata.org/docs/

[9] NumPy Documentation: https://numpy.org/doc/stable/

[10] nltk Documentation: https://www.nltk.org/

[11] Contractions Retrieved from: https://github.com/kootenpv/contractions

[12] nlputils: https://pypi.org/project/nlputils/

[13] worldcloud: https://pypi.org/project/wordcloud/