Wpierw zalozylem prywatne repozytorium na gicie, posciagalem wszystkie potrzebne biblioteki, django, nltk itp

**TWEETS**

I created a twitter account mas15@aber.ac.uk, kol

Created twitter app to obtain keys and access tokens

https://apps.twitter.com/app/14789943

Consumer Key (API Key) sHEmHwtt3koxdLoa6Ok2vEduH

Consumer Secret (API Secret) fJZsN0OQW80Vqnw265rT8Jvc7VwADGNS0kB5vMjIRG4d3eywzJ

Access Token 962374758783471616-KZtDMvJkmJigxZWUr3EI8x5iOgguRQB

Access Token Secret hLtD8RMXyT2kcUDR9oLg7P7MtXSkrlgBWgsZk4u8GJY84

I used a code from <http://tweepy.readthedocs.io/en/v3.5.0/getting_started.html>

tweety z http://uk.businessinsider.com/trump-tweets-of-the-year-2017-12?r=US&IR=T/#when-he-chastised-so-called-russian-hacking-1

https://news.sky.com/story/sad-pathetic-a-history-of-donald-trumps-twitter-insults-11123543

**Sentiment anysis:**

I have chosen manually about 120 tweets that half was positive and half was negative.

we have got 2 data sets : positive, negative, each 60 tweets

Then I split corpus into test and trainig to do k-cross validation with stratification(same amount pos and neg for train and test). I wrote a code which does k-cross validation by myself because one from scikit learn was harder to use and I had some problems with stratification?

First step to do classification is to get a vocabulary. To do so tweets are split by sentences and are stripped from unnecessary whitespaces etc. Then each tweet is lowercased.

When tweets are preprocessed we extract phrases from each of the tweets. To do so we split each tweet by stopwords (on, at, the, yours). I used stop words from blablabla. To sa nasi candidates.

Then for each of the candidates we check if the phrase meets our expecteation (is it 2- 3 words in phrase) and we check how many times the phrase occurred in the corpus. It gave the best results when I set min\_phrase\_freq to 2.

Phrases that did not match these warunkom are split into words. From these words we remove ones that are shorter than 3 letters. Each of the words is lemmized then. Example here. I also tried to use stemming but it gave very bad results and was useless.

When we have got a vocabulary we can then do a classification.

To do so we need a feature\_extraction function that will extract features from vobaluary in the tweets we want to classify. This function does almost the same as building…

K-fold z http://thelillysblog.com/2017/08/18/machine-learning-k-fold-validation/

Firstly I tried NB without preprocessing the data. Only lower case. It had about 82% percents of accuracy but was using wrong words as features. Most of them were stopwords and () etc.

Using MaxEntClassifier gave similiar results

Average accuracy was abnout 75% (0.71, 0.82, 0.83)

But because they were classifitying w oparciu o stopwrods they were not able to give a sentiment for example “Croocked hilary”. The model was overfitted.

When I removed stopwords and (), @ etc I got 83% accuracy..

Then I split by phrases. It gave about 84% accuracy.;

When added RAKE then got 77% accuracy.

(using lower + remove stopwords gave 83-84% on average of 30 but included words like of, the, as)

Wiec preprocessing to split by words, lower case itp…

Poprawilem pare błedów + #@ I 0.805% accu

# do czegos tam uzylem from http://textblob.readthedocs.io/en/dev/classifiers.html

#stop word list from SMART (Salton,1971). Available at ftp://ftp.cs.cornell.edu/pub/smart/english.stop

RAKE: https://www.airpair.com/nlp/keyword-extraction-tutorial

To do feature extraction I had to write a simple algorithm by myself. I tried few different ones like ConllExtractor or FastExt(z textbloba) but they were dawaly zle rezulataty. One that was was better was a Rake (z githuba) but had not good opcji konfiguracji so I had to write one by myself. I skipped adjusted keywords (us OF a) because while using original rake it was finding maybe one of them so it was not worth to write it and it gave more FP that proper ADJ keywords.

I wrote it based on https://www.researchgate.net/publication/227988510\_Automatic\_Keyword\_Extraction\_from\_Individual\_Documents

Lemmatizing worked mostly properly except of few cases: (, isis - isi, philippines - philippine, pass – pas)

dodac trzeba do tego że jaki POS print(lemmatizer.lemmatize("best", pos="a"))

**FLASK**

IMAGE Z http://www.pngmart.com/image/28615

I wrote a simple interface in flask and made charts using chart.js.

CSS was made using Bootstrap.

**ASSOSIATION**

Twitter only allows access users to get most recent 3240 tweets with this method

Hopefully we obtained about 2935 tweets from 01.2017 to 03.2018. I removed few which were in Japanesse or included only videos or images

I removed Unicode characters that are useless such as, ✔💜 ➡✅ and changed few of then into proper characters: &amp -> and.

I deleted all links from tweets.

Now 2929 tweets.

719F, 2929 Tweets, UP/DOWN target:

I trained using weka NB on these tweets features…(719) + sent + change (up/down).

The same day, for tweets after 22 used next day. NB gave 54% accuracy on 10-kros, j48 – 53.7%

Running a select attributes:

Merit: 0.634, 5-fold on full training set with Wrapper and BestFirst

Removed short and about happy birthdays 2685

Usuniecie …2479

Potem sie zczailem ze featury nie te i jest ich 5706… accu 55%

JRip(Ripper)

Usuniecie więcej .. - 2416

Potem bezsensowne i o interview o RT – 2195

Teraz bayes 54.8769 %(90.6% on train) ale ZeroR - 52.6436 %

A J48 na 10-k 51.1851 % (train 81.495 %)

Logistic 51.5953 %(10-fold) a na train 100%

POTEM POPRAWILEM #@ bo w ogole z 700 kolumn było pustych

Kolumn dalo 5400, to avoid overfitting I removed ones that occurred to rarely

Usunalem <4 I bylo 55.912 % zostalo 1185 feat

Usunalem <6 i było 56.6453 % (73 train) zostało 859 feat (dropping <5,4,3) was giving worse or the same results)

Logistic 52% 1-k(84)

Odpaliłem WRap, nb, bestfirst na train i dalo 0.688 merit

Po wyjebaniu tych z wrap zostaje 116+2 i NB daje 67.7819 %(70) a J48 62%(69.4%)

A logistic 66.8, (69.2%)

ZMIANA NA 3 WARTOŚCI UP/DOWN/NC

Treshold 0.1 w Wece: NB 46(52), LOG 47(53), J48 45 (54)

Sci : LOG 43,4% (48/3%), NB 45.8%(53%),

Potem usunalem puste wiersze i dodałem marking features again, dodałem stratyfikacje i pusiclem 30 run x 10 folds – 30 bo dawalo już tyle samo co 100

i wyszlo:

W wece log 55, nb 55(61) a SCIKIT: NB 56.41(56,6), LOG 53.22,

POTEM ZMIENILEM

Treshold: na 1/3 std (1/3) sigmy : 0.1223 %

Teraz ZeroR 36%

NB 0.465 (50.9%), log ()

Potem z meanem przesunalem, zeroR: 41%

NB 52.7% (0.597) LOG 0.492(0.541)

Potem zmieniłem tweet sentyment z 0/1 na 0.0-1.0

NB 52.7% (0.597) LOG 0.491(0.540)

REMOVE RT

Usunalem RT i z 2149 zrobilo się 2025

Removed rt because endormesment, moze sie z inim nie zgadzać i to nei jego słowa?

Teraz

NB 0.529(0.602), LOG 49.44(55.07%)

Nowe atrr to remove hujowo działają po usunieciu RT ale nie markowałem jeszcze raz…

**TODO**

Sprawdzić accu na pos i negatywnach?

polaczone w jeden tam gdzie ....

TODO dzielic tagi na slowa #afghanstrategy -> afghan strategy

remove not needed features - select features, mniej niż 50% foldsóws,

usunac tweety co nie wplywaja na rynek,

zobaczyc drzewko po czym wybiera atrybuty,.

zmienic na up, down, neutral with treshold, different comodities, pokazac na stronie features selection jaki maja wpływ

requirements z venv

RULES: JRIP, PART

LEMMATIZE PHRASES

Sprawdzać czy w cols\_to\_drop nie ma change – jako featurea

Uploadowanie własnych csv

Sciaganac index euro i pound

Pokazac wplytowe featuery

Sprawdzić wrapper może w tym nowym module? mlxtend

Assosietion znaleźć

Testy sentyment

Testy main model

Plik który przetwarza tweety

Feature selection z maxa

(0, 1, 10, 14, 15, 19, 20, 21, 35, 38, 40, 43, 49, 55, 58, 63, 64, 66, 68, 71, 73, 79, 86, 90, 91, 94, 100, 111, 112, 115, 120, 122, 124, 134, 143, 146, 147, 149, 150, 152, 154, 157, 160, 163, 170, 176, 181, 188, 193, 196, 209, 211, 217, 221, 223, 227, 228, 230, 233, 236, 244, 247, 248, 255, 258, 264, 273, 279, 285, 313, 317, 319, 328, 329, 335, 338, 350, 352, 353, 358, 360, 362, 364, 367, 368, 370, 376, 379, 380, 384, 395, 399, 402, 407, 411, 416, 417, 419, 424, 435, 440, 441, 445, 452, 454, 459, 468, 471, 478, 487, 488, 500, 502, 504, 512, 515, 516, 531, 533, 537, 544, 563, 565, 578, 596, 608, 614, 627, 642, 643, 645, 647, 652, 653, 659, 697, 700, 715, 722, 726, 727, 730, 764, 772, 777, 783, 788, 793, 796, 800)