SENTIMENT ANALYSIS FOR CLIMATE CHANGE

Alberto Formaggio 21145148 -
Data Mining NDBI023

WHAT IS SENTIMENT ANALYSIS?

Sentiment analysis is the process of analyzing online pieces of writing to determine the emotional tone they carry, whether **they're positive, negative, or neutral**.
In simple words, sentiment analysis helps to find the author's attitude towards a topic.
In our case, we are going to create a model to study the attitude of people towards Climate Change by scraping tweets from Twitter.

We must combat climate change. Indigenous ppl are protesting for our earth, our water. Yet this happens: https://t.co/EUDmjbgbkx

@crabtrem Yes! Imagine all the money wasted. Trump's plan to ditch payments for climate change will save us billions!!

PRE-PROCESSING

01

THE CLEANING PROCESS

Text Cleaning

Lowercasing. Punctuation, Mentions, Hashtags removal. Removal of stopwords

Words can have spelling errors that add noise to the data

Spell-checking

Lemmatization

Words get substituted by their lemma

Sentences are split into lists of tokens

Tokenization

DATA ANALYTICS

02

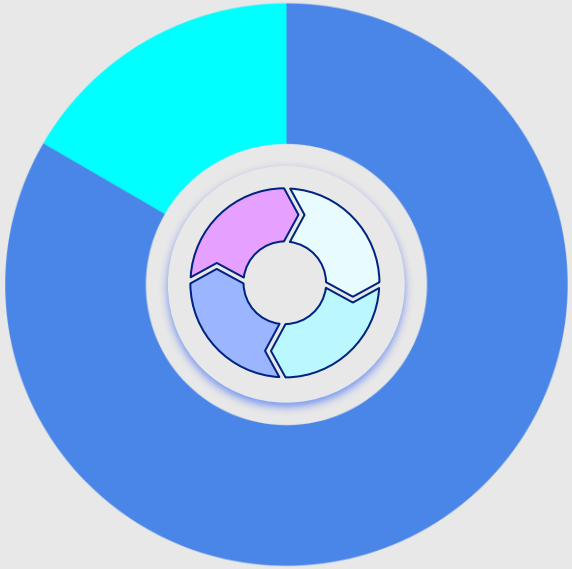
CLASS DISTRIBUTION

84.2% **POSITIVE**

Tweets that express support to Climate Change

15.8% **NEGATIVE**

Tweets of people that don't think Climate Change is an issue

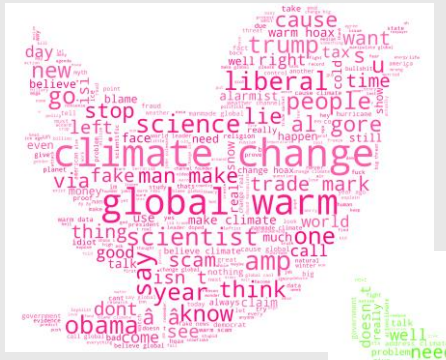


We can see that we have class unbalance in our training data, we must take this into account when training the classifiers.

WORD DISTRIBUTION

WordCloud was used to get useful insight into the word distribution. We can see here the distributions of the words separated by class.

Negative

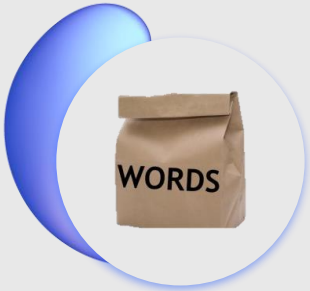


Positive

MINING PROCESS

03

TEXT PROCESSING APPROACHES

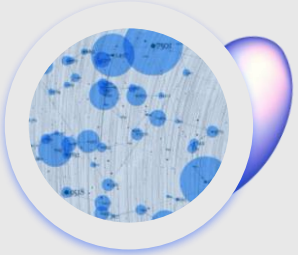


BAG OF WORDS

Bag of words is a measure that considers the number of times that a word (or bigram, trigram, etc.) appears in a sentence.

TF - IDF

TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents.



WORD EMBEDDING

Word embeddings are used to display words in the feature space in such a way that similar words are close to each other.

RESULTS WITH BAG OF WORDS

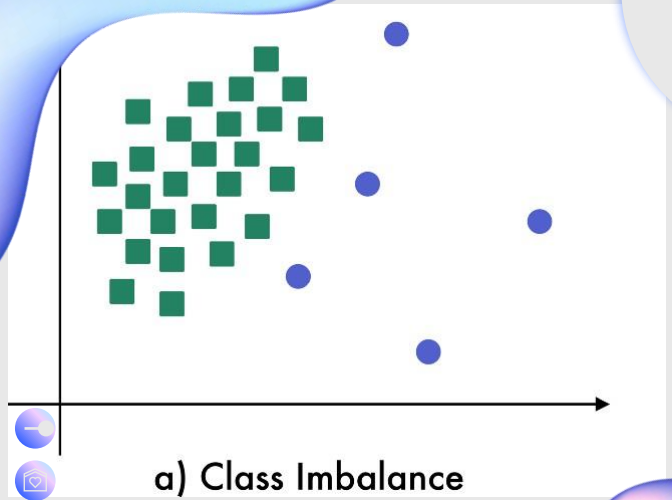
	ACCURACY	PRECISION	RECALL	F1-SCORE
MULTINOMIAL BAYES CLASSIFIER	87.6%	77%	79%	78%
LOGITSTIC REGRESSION	86.7%	75%	78%	77%
LINEAR SVM	87.2%	76%	78%	77%
NEURAL NETWORK	89.3%	78%	78%	78%

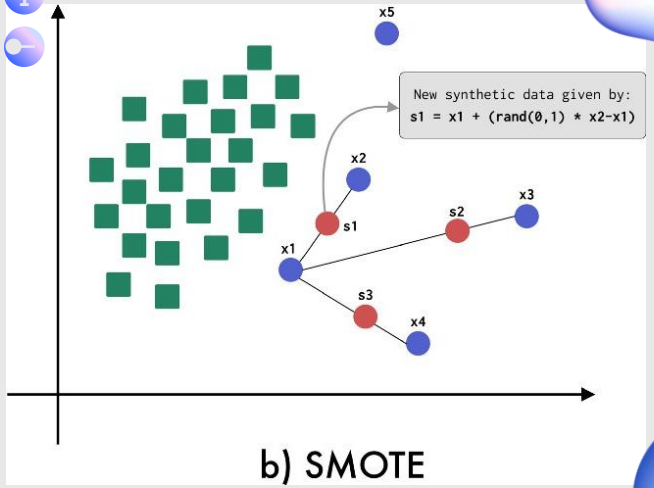
Bigrams and unigrams with a maximum of 15000 features were used. This number was chosen according to the computational power available. Probably the results can be improved with more powerful hardware.

RESULTS WITH TF-IDF

	ACCURACY	PRECISION	RECALL	F1-SCORE
MULTINOMIAL BAYES CLASSIFIER	89.7%	86%	76%	76%
LOGITSTIC REGRESSION	90.8%	85%	78%	81%
LINEAR SVM	91.2%	87%	78%	81%
NEURAL NETWORK	87.8%	77%	79%	78%

Bigrams and unigrams with a maximum of 15000 features were used. This number was chosen according to the computational power available. Probably the results can be improved with more powerful hardware.





HOW WAS CLASS IMBALANCE TACKLED IN THESE 2 MODELS?

When we have class imbalance, training the model without any change would lead to a model biased towards the majority class.
This problem can be overcomed as follows:

- Downsampling
- Oversampling

Oversampling with **SMOTE** led to the best performance.

Furthermore, the **macro average of the F1-score** metric was used to assess the goodness of the model rather than the accuracy

CHOICE OF THE BEST WORD EMBEDDING

We tested 2 different Word Embeddings in order to find the most suitable one for our task:

1) **Google News Word2Vec:** It contains vectors trained on 100 billion words from the Google News dataset. The resulting vectors have 300 features.
2) **Global-Vector Twitter:** It contains pre-trained vectors trained over tweets. The resulting vectors have 200 features.

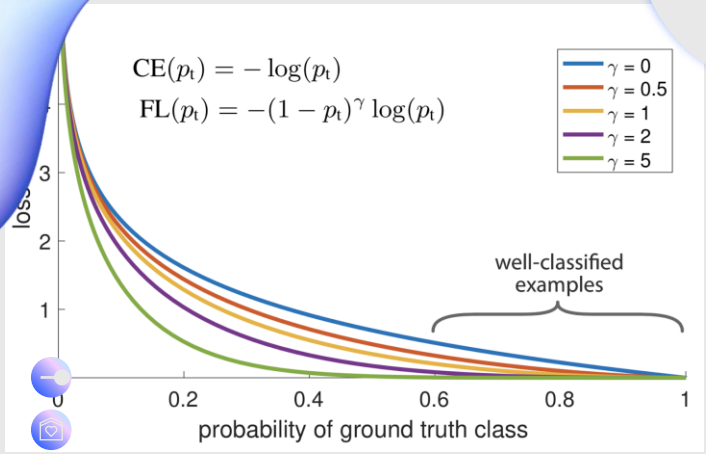
The results were obtained by applying a Logistic Regression classifier.

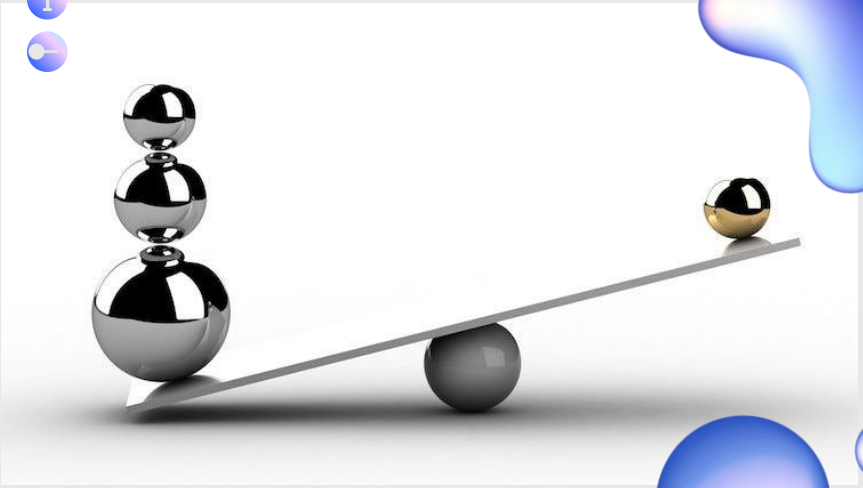
	ACCURACY	PRECISION	RECALL	F1-SCORE
WORD2VEC	84.0%	71%	75%	73%
GLOVE	81.3%	67%	70%	68%

RESULTS WITH WORD EMBEDDINGS

	ACCURACY	PRECISION	RECALL	F1-SCORE
LINEAR SVM	84.4%	72%	76%	73%
POLY-KERNEL SVM	86%	74%	79%	76%
NEURAL NETWORK	86.7%	75%	80%	77%
RNN	88.1%	78%	75%	76%

The parameters for non-neural network approaches were computed optimally by using Cross Validation (GridSearchCV).





HOW WAS CLASS INBALANCE HANDLED WITH WORD EMBEDDINGS?

Unfortunately SMOTE is not working well with Word Embeddings, so a new approach was needed.

I tried:

- Downsampling
- Change class weights
- Focal Loss

Out of these approaches we used:

- Focal Loss for NNs and RNNs
- No modification of data for the other models

COMPARISON WITH STATE-OF-THE-ART MODELS

	ACCURACY	PRECISION	RECALL	F1-SCORE
VADER (NO FINE TUNING)	57.8%	52%	53%	48%
DISTILBERT (BS=32, WITH PREPROCESSING)	87.9%	77%	84%	80%
DISTILBERT (BS=64, W/O PREPROCESSING)	92.8%	88%	84%	86%
DISTILBERT (BS=64, WD=0.01, W/O PREPROCESSING)	92.4%	86%	86%	86%

BS = Batch Size, WD = Weight decay.
DistilBERT = lighter and faster BERT implementation, appropriate when the hardware is limited

CONCLUSIONS

BEST PERFORMING MODEL: BERT

PROS:

- Good performance (86% F1-score)

CONS:

- High training time required
- Many resources needed
- Hard to implement

2° BEST MODEL: LINEAR SVM WITH TF-IDF

PROS:

- Fast training Time
- Easy to understand

CONS:

- Slightly less precise than BERT

THANKS

Do you have any questions?