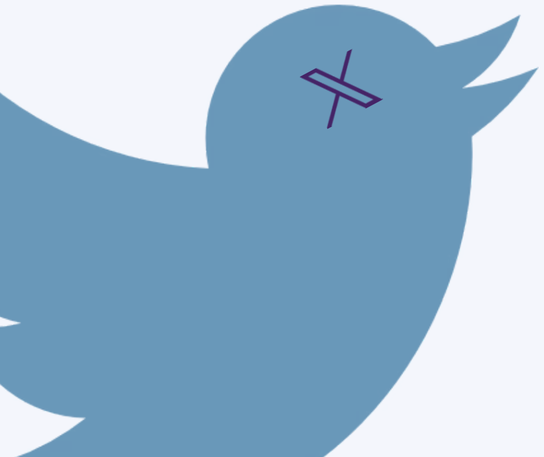
Sentiment Analysis Tool Using PySpark

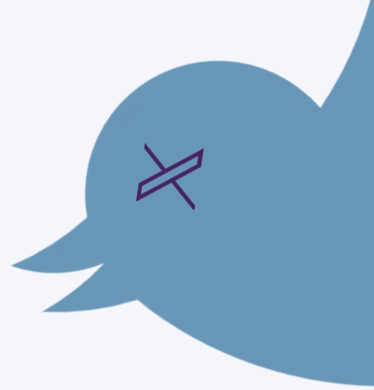
Unlocking Insights from Twitter Data



Riccardo Ruberto 1860609

Big Data Computing course A.A. 2022-2023

What is sentiment analysis ?



What is sentiment analysis ?

Is the process of evaluating and determining the emotional **tone** and **polarity** in text data

Analyzing sentiment in tweets

Analyzing sentiment in tweets

- provides insights into public opinions and emotions
- supports brand reputation management
- helps detect trends and emerging issues
- …

Data source

Sentiment140 dataset with 1.6million tweets

TARGET	**Polarity** of the tweet (0 = NEGATIVE, 2 = NEUTRAL, 4 = POSITIVE)
ID	**ID** of the tweet (AUTO-INCREMENT)
DATE	**DATE** of the tweet (DAYNAME MONTHNAME DAY HH:MM:SS TIMEZONE YEAR)
FLAG	Specific **QUERY** used (NO_QUERY if no query was used)
USER	**USER** that tweeted
TEXT	**Content** of the tweet

Data pre-processing

Cleaning and **structuring** the dataset.

- Conversion of **TARGET** to binary labels
  (0 = NEGATIVE, 1 = POSITIVE)

Ther are no **NEUTRAL** = 2 values in the dataset

Data pre-processing

Cleaning and **structuring** the dataset.

- Extraction of **HOUR** and **DAYNAME** from the **DATE**

The dataset contains dates for only 2 months of 2009, so the **YEAR** and **MONTH** are not useful

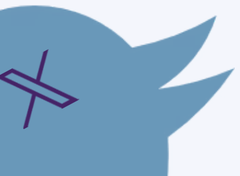
Data pre-processing

Cleaning and **structuring** the dataset.

- **FLAG, ID, USER** removal

FLAG has only one value within the dataset

ID and **USER** are not useful



Cleaned Data source

TARGET	**Polarity** of the tweet (0 = NEGATIVE, 1 = POSITIVE)
DAY_NAME	**Day of the week** (Mon, Tue, Wed…)
HOUR	**Hour** (HH format)
TEXT	**Content** of the tweet

Text pre-processing

Case normalization and **Trimming**

“Text To Normalize”

↓

“text to normalize”

Text pre-processing

Username and Link removal

“thanks to @Riccardo
http://github.com/RiccardoRobb !!!”

↓

“thanks to !!!”

Text pre-processing

Punctuation symbol removal

“thanks to !!!”

↓

“thanks to”

Text pre-processing

Tokenization

“thanks to”

↓

[“thanks“, “to“]

Tokenization is vital for processing and analyzing text effectively in natural language processing

Text pre-processing

Stopwords removal

[“example“, “of“, “stopword“, “removal“]

↓

[“example“, “stopword“, “removal“]

In order to reduce the dimensionality of text data
and focuses analysis on the more meaningful words

Text pre-processing

Stemming

Using **SnowballStemmer** for english

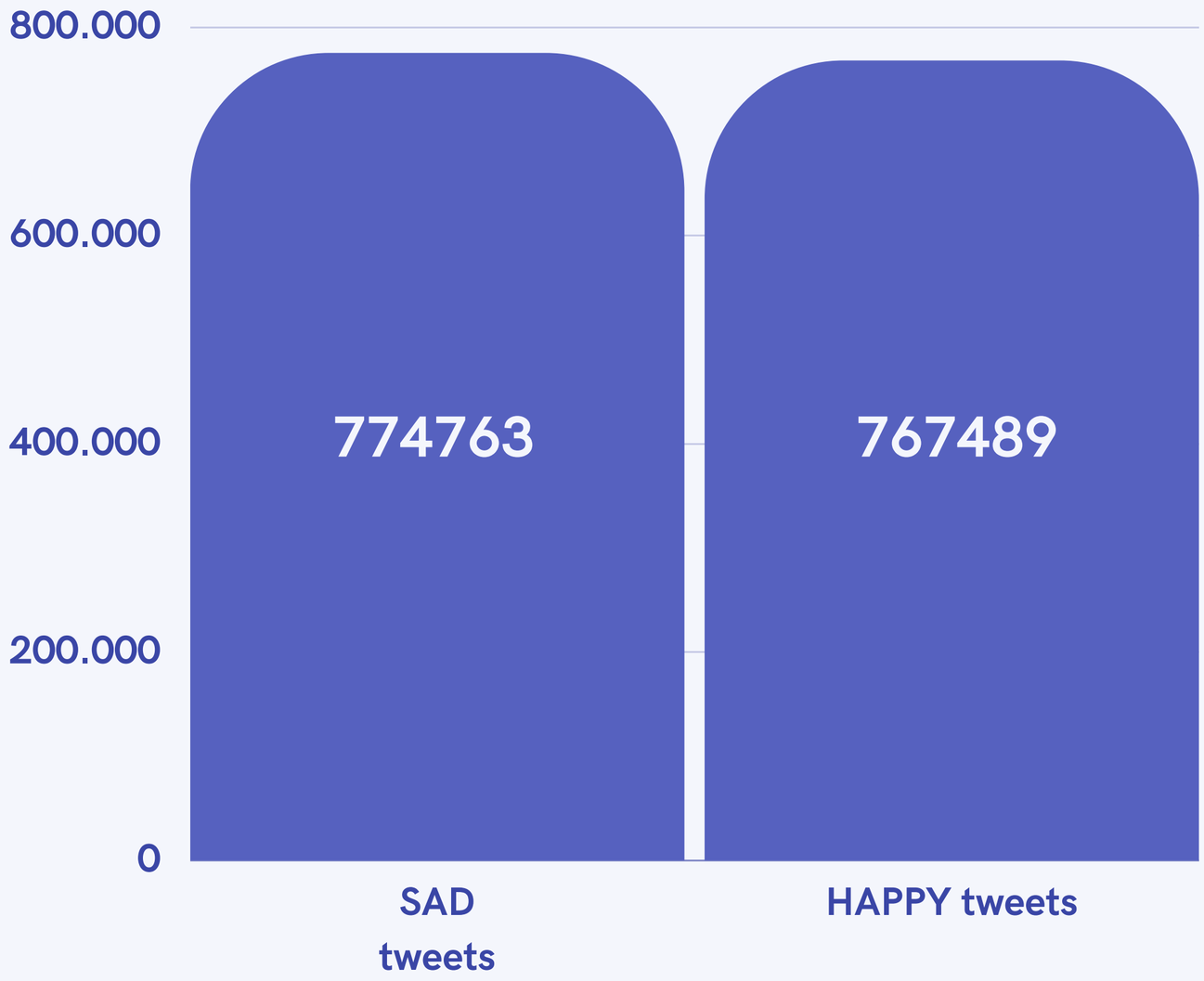
[“example“, “stopword“, “removal“]

↓

[“exampl“, “stopword“, “remov“]

Used to eliminate variations of words so that different forms of the same word are treated as one

Dataset balancing



Balanced removing random **SAD** rows

Embedding mapping

Gensim library GloVe

GloVe (**Global Vectors for Word Representation**) embeddings allows us to represent each word in our text data as a fixed-length numerical vector

It’s based on the fact that words occurring in similar contexts tend to have similar meaning

Dataset splitting

Test dataset = 20%

Train dataset = 80%

+

Weights assigment to
HOUR and **DAY_NAME**

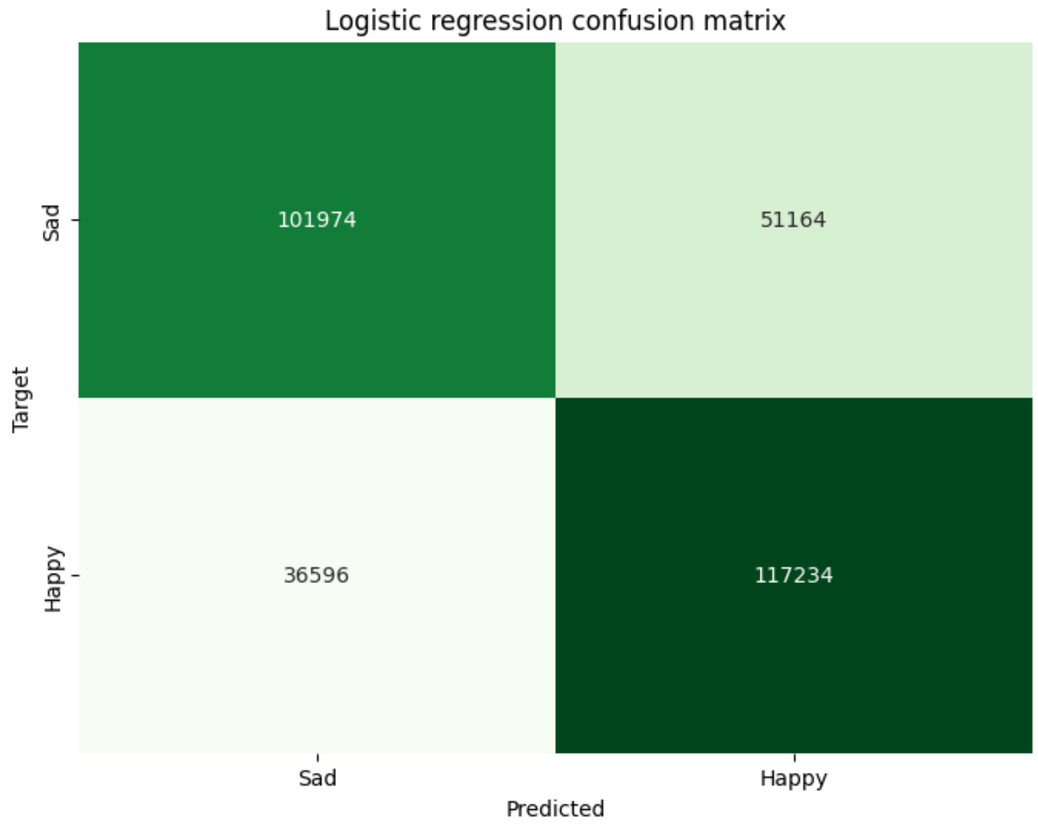
Assigning weights to data is particularly important because it helps create a more balanced, fair, and accurate model

PySpark models

- Logistic regression
- Support Vector machines
- Decision tree
- Random forest

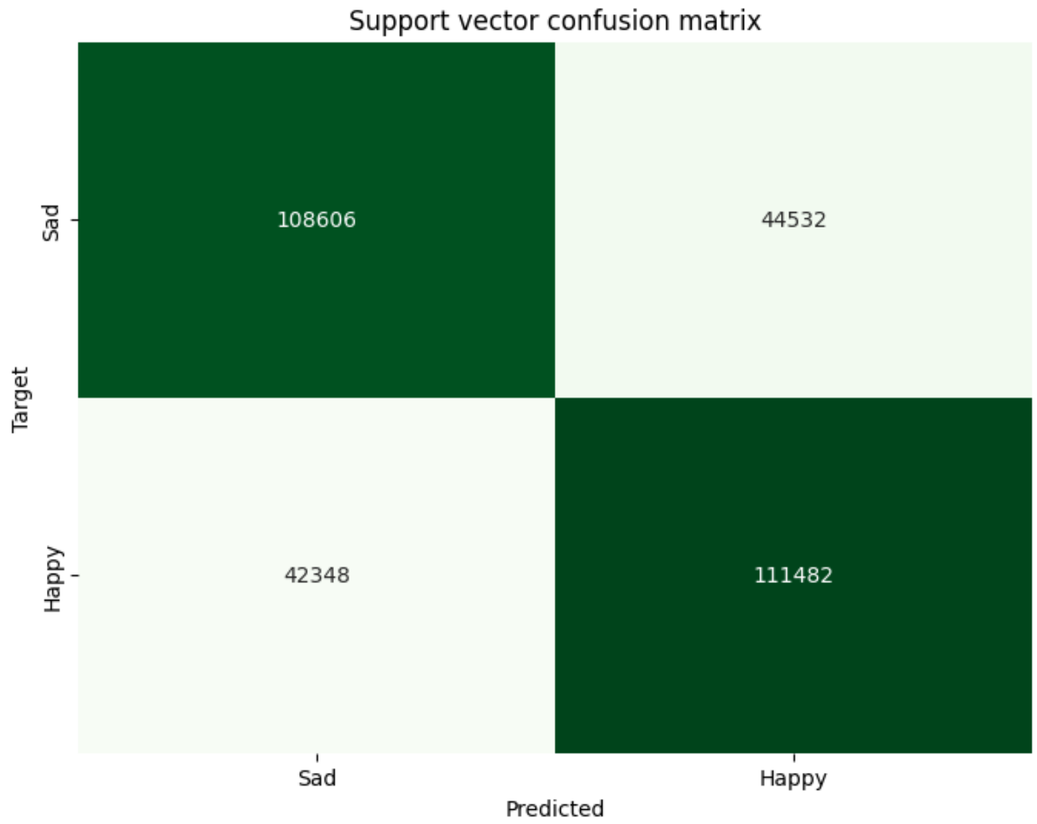
Logistic regression
results

Precision	0.69
Recall	0.76
F1 Score	0.73
Accuracy	0.71



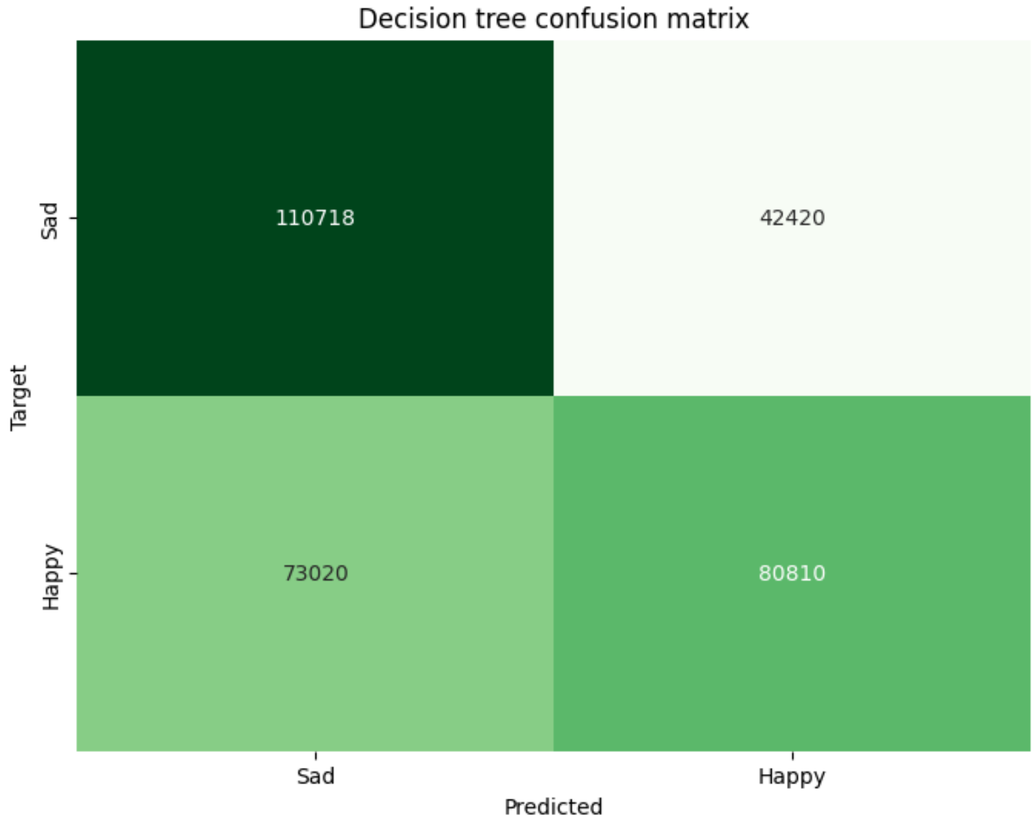
Support Vector
results

Precision	0.71
Recall	0.72
F1 Score	0.72
Accuracy	0.72



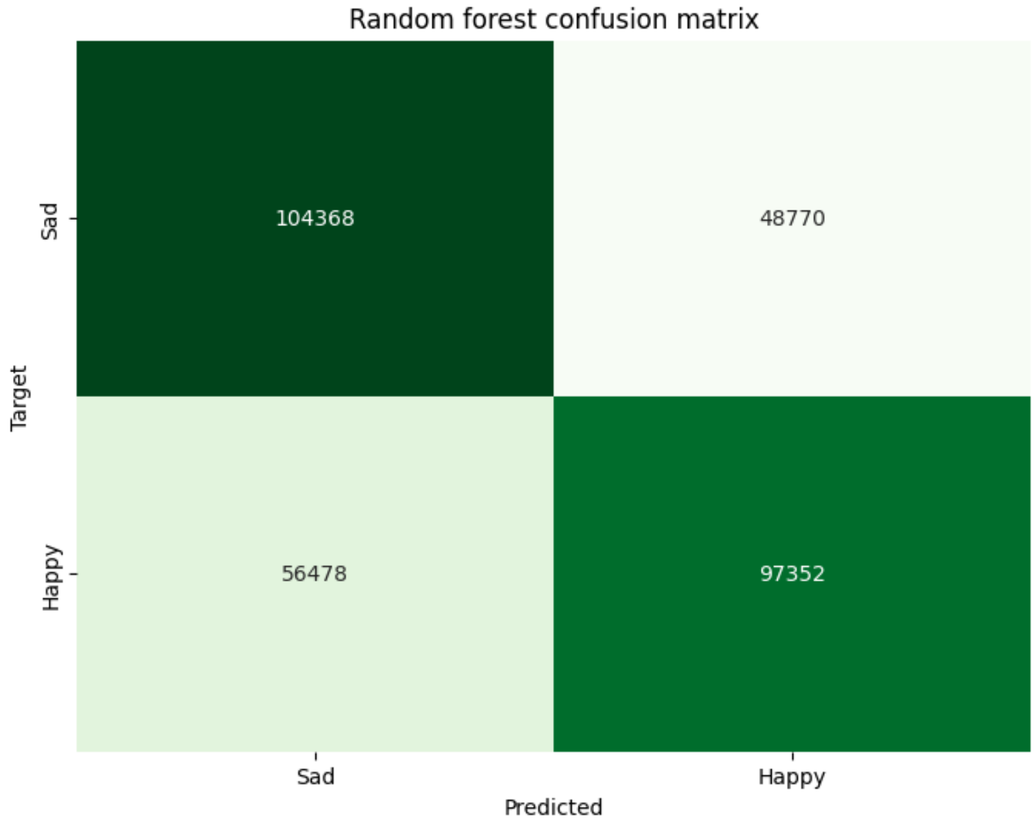
Decision Tree
results

Precision	0.65
Recall	0.52
F1 Score	0.58
Accuracy	0.62



Random forest
results

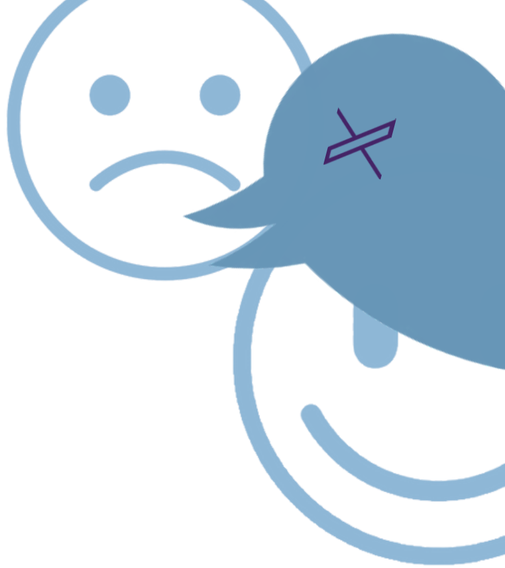
Precision	0.66
Recall	0.63
F1 Score	0.65
Accuracy	0.65



Firefox extension

live demo

Future work



- **Automatize feedback system**

  Append new record to the train dataset and re-train the model in background

- **Build a powerful model**

  To support typos and no sense phrases

- **Hyperparameter tuning**

  Try new combinations of tree depht Decision tree and number of trees Random forest