Born Classifier Project Report

Sentiment Analysis

By: Arina Lopukhina

PROJECT PLAN

1

EDA & Pre-processing

- studying the dataset content
- cleaning the text from redundancies

2

Sentiment analysis

- training Born Classifier to recognise Positive or Negative sentiment
- analysis of important tokens

3

Aspect edition

- using Born + Multiclass classifier to form dictionary for each aspect
- assign aspect to a specific text portion
- use trained Born for sentiment analysis

DATA

FABSA dataset overview

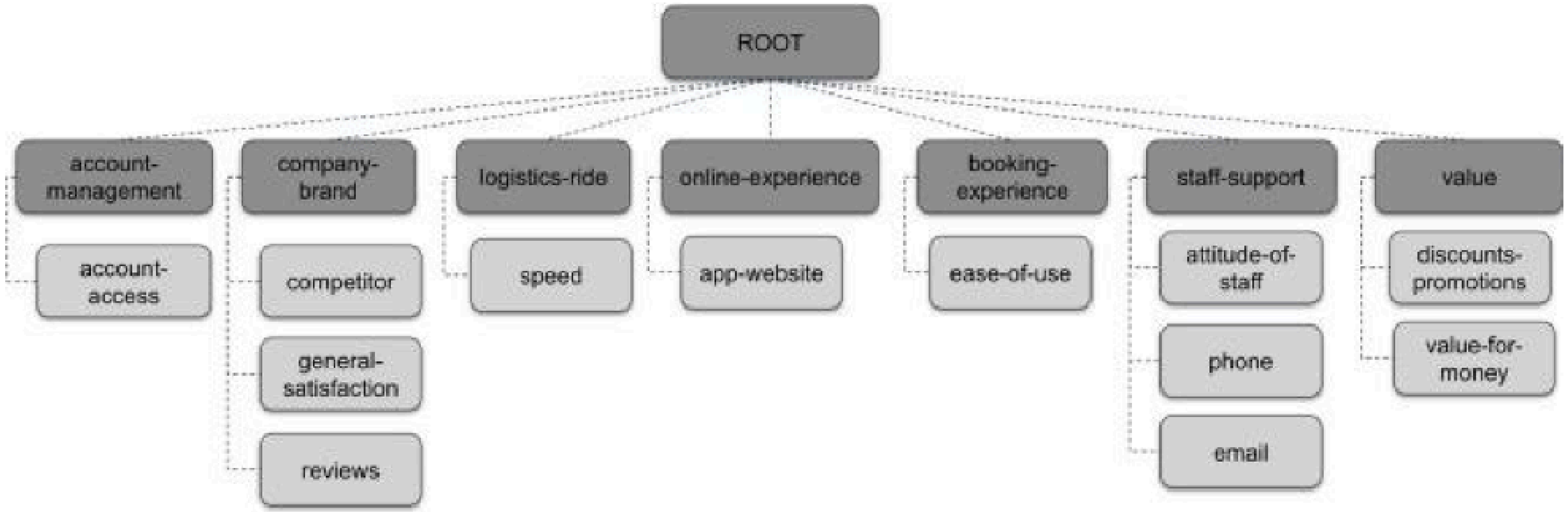
- Data scraped from Trustpilot, Apple Store, and Google Play, the latter being the prevalent source.
- Over 10k observations tagged with sentiment, industry, and source website.
- Already split in test and train datasets.
- See sample below:

	id	org_index	data_source	industry	text	labels	label_codes
0	301972057	600	Trustpilot	Price Comparison	My experience is only around the Parking forum...	[[Staff support: Attitude of staff, negative],...	['staff-support.attitude-of-staff.-1', 'compan...
1	301982453	514	Google Play	Banking	I love it so handy, plus I hate my bank so it ...	[[Company brand: General satisfaction, positiv...	['company-brand.general-satisfaction.1', 'comp...
2	301980653	369	Google Play	Ride Hailing	Sometimes it takes	[[Company brand: General satisfaction, negative]]	['company-brand.general-satisfaction.-1']
3	301979991	727	Apple Store	Fashion	This is the worst app I ordered my sneakers 2/...	[[Logistics rides: Speed, negative], [Online e...	['logistics-rides.speed.-1', 'online-experienc...
4	301984330	549	Google Play	Travel Booking	So easy & loads of info !	[[Company brand: General satisfaction, positive]]	['company-brand.general-satisfaction.1']

DATA

Aspects definition

- For further analysis, only the umbrella aspects were considered



DATA CHALLENGES

1. **Unbalanced sentiment and aspect classes**
   - 2:3 ratio between Positive and Negative
   - some aspects were as few as a few dozens of instances
2. **Strange aspect allocation**
   - the logic behind aspect assignment was not always fully clear -> training data quality might not have been so high.
3. **Many redundant typos, emojis, etc.**
   - extensive data cleaning needed

Exploratory data analysis

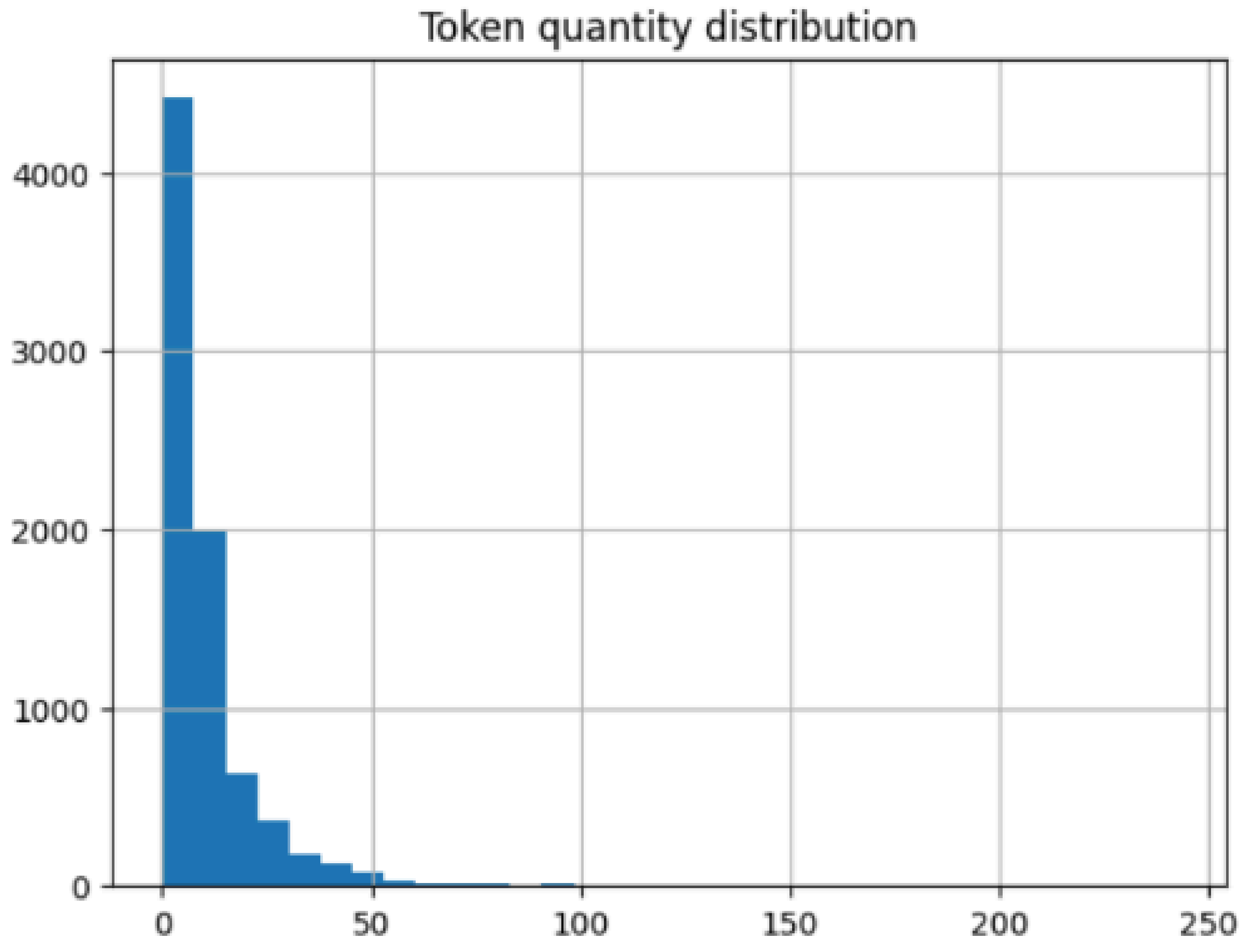
industry	count
Fashion	2161
Price Comparison	1157
Groceries	1021
Trading	1021
Travel Booking	973
Banking	913
Ride Hailing	383
Information Technology	141
Consulting	81
Streaming	79

industry	count
Fashion	419
Price Comparison	239
Travel Booking	211
Groceries	206
Trading	195
Banking	189
Ride Hailing	83
Information Technology	23
Consulting	13
Streaming	9

- in both test and train datasets, most common industry - Fashion
- industry breakdown is rather evenly split in both datasets

Exploratory data analysis

Most text samples were rather brief, containing less than 10 tokens. The longer reviews could contain as many as a little over 50 tokens, which is still limited.



Text Preprocessing

Actions performed to standardise textual input before vectorization:

1. Eliminate special characters such as emojis, punctuation, etc.
2. Lowercase all the input
3. Removing stop words
4. Lemmatizing with the helf of NLTK library
5. Leaving only words with the following POS tagging:

- JJ = Adjective
- RB = Adverb
- NN = Singular noun
- VB = Verb, base form
- PRP = Personal pronoun
- JJR = Comparative adjective
- JJS = Superlative adjective

6. Vectorize using TD-IDF method, allowing N-grams of up to 3 tokens + eliminate too rare and too frequent tokens.

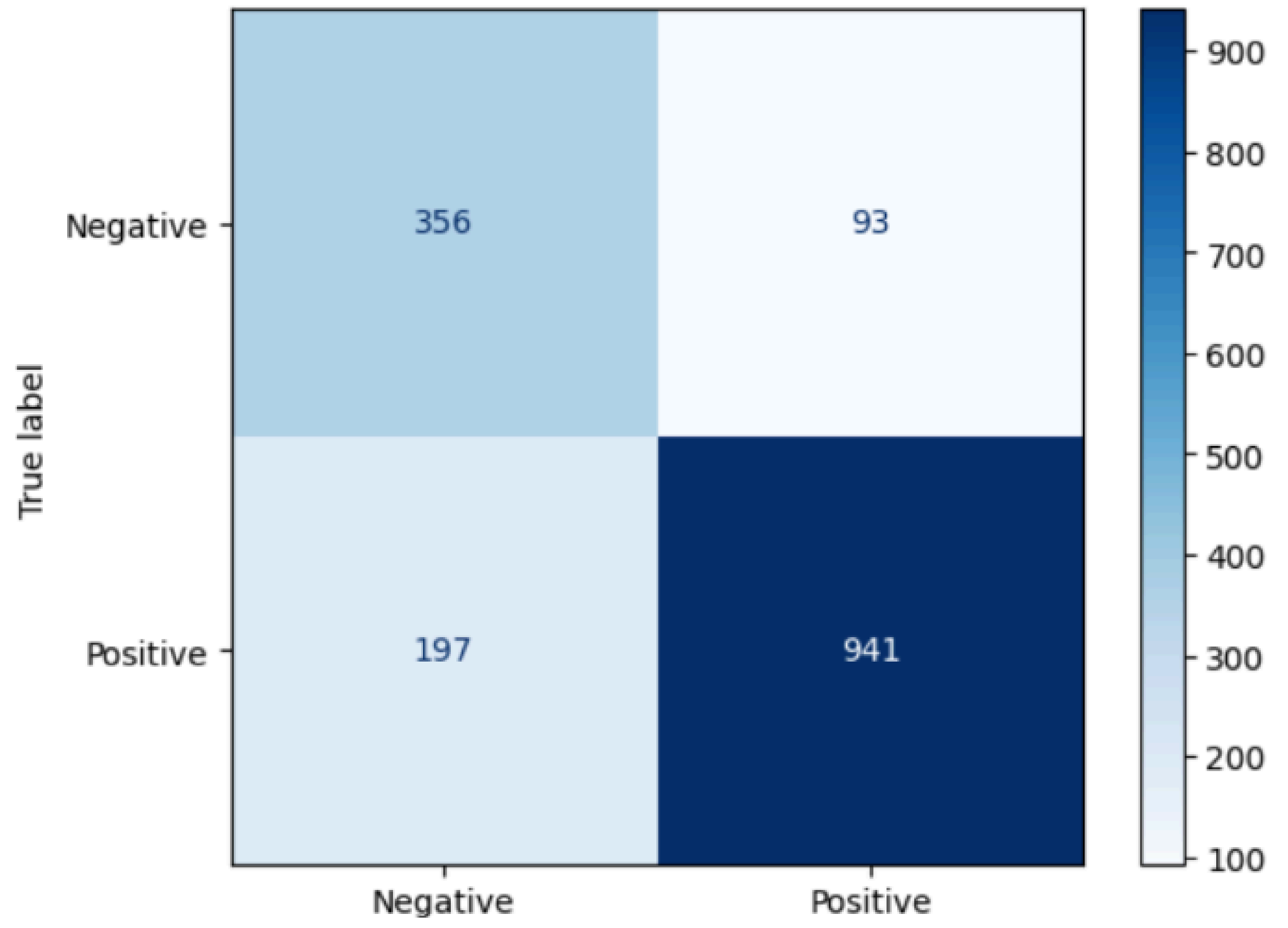
SENTIMENT ANALYSIS

Born Classifier

Approach:

- both classical probability model weights and the quantum probability model weights were applied
- quantum probability approach has yielded much better performance, reaching accuracy as high as 0.82.
- it was attempted to experiment with different weight combinations but no notable improvements were noted, so the a = 0.5, b= 1, h= 1 combination was used throughout the project.
- SVM and Logistic regression models were employed for comparison and showed slightly higher accuracy scores, but without dramatic improvments

	precision	recall	f1-score	support
0	0.64	0.79	0.71	449
1	0.91	0.83	0.87	1138
accuracy			0.82	1587
macro avg	0.78	0.81	0.79	1587
weighted avg	0.83	0.82	0.82	1587



Born Classifier

	Negative	Positive	Tokens
0	0.000161	0.000144	ability
1	0.006634	0.004409	able
2	0.004698	0.009808	abroad
3	0.001033	0.001281	absolutely
4	0.011493	0.005493	accept
5	0.000939	0.000797	access
6	0.000039	0.000037	accommodation
7	0.000006	0.000006	account

	Tokens	Negative
6622	terrible	0.052396
1550	crash	0.038397
6099	slow	0.038264
3111	horrible	0.037497
4907	poor	0.037376

	Tokens	Positive
2058	easy	0.175315
2877	great	0.126628
3890	love	0.105355
2450	fast	0.084642
2307	excellent	0.082300

- using **explain** function from the Born classifier package, it was possible to extract tokens which were more often associated with a given class

ASPECT-BASED SENTIMENT ANALYSIS

Defining dictionary per aspect

- Born classifier was utilized with the OnevsRestClassifier to create a classification model for each of the 7 umbrella aspects defined by the FABSA dataset creators (‘account management', 'company brand', 'logistics rides', 'online experience', 'purchase booking experience', 'staff support', 'value') using the train dataset.
- mirroring previous steps, **explain** was utilized to pull 100 most used words for each class (there were a lot of overlaps so expanding a dictionary negatively affected precision)
- precision was not ideal for most of the classes; the best one so far was ‘online experience’ one.

	precision	recall	f1-score	support
0	0.30	0.86	0.44	79
1	0.67	0.65	0.66	692
2	0.41	0.84	0.55	189
3	0.83	0.83	0.83	724
4	0.67	0.83	0.74	503
5	0.53	0.85	0.65	224
6	0.57	0.82	0.67	288
micro avg	0.62	0.79	0.69	2699
macro avg	0.57	0.81	0.65	2699
weighted avg	0.66	0.79	0.71	2699
samples avg	0.66	0.80	0.68	2699

Mapping part of document to aspect

0. Prepropresessing was redone from scratch, splitting text into smaller pieces based on **punctuation** and presence of **connector words** such as ‘but’, ‘however’, etc.

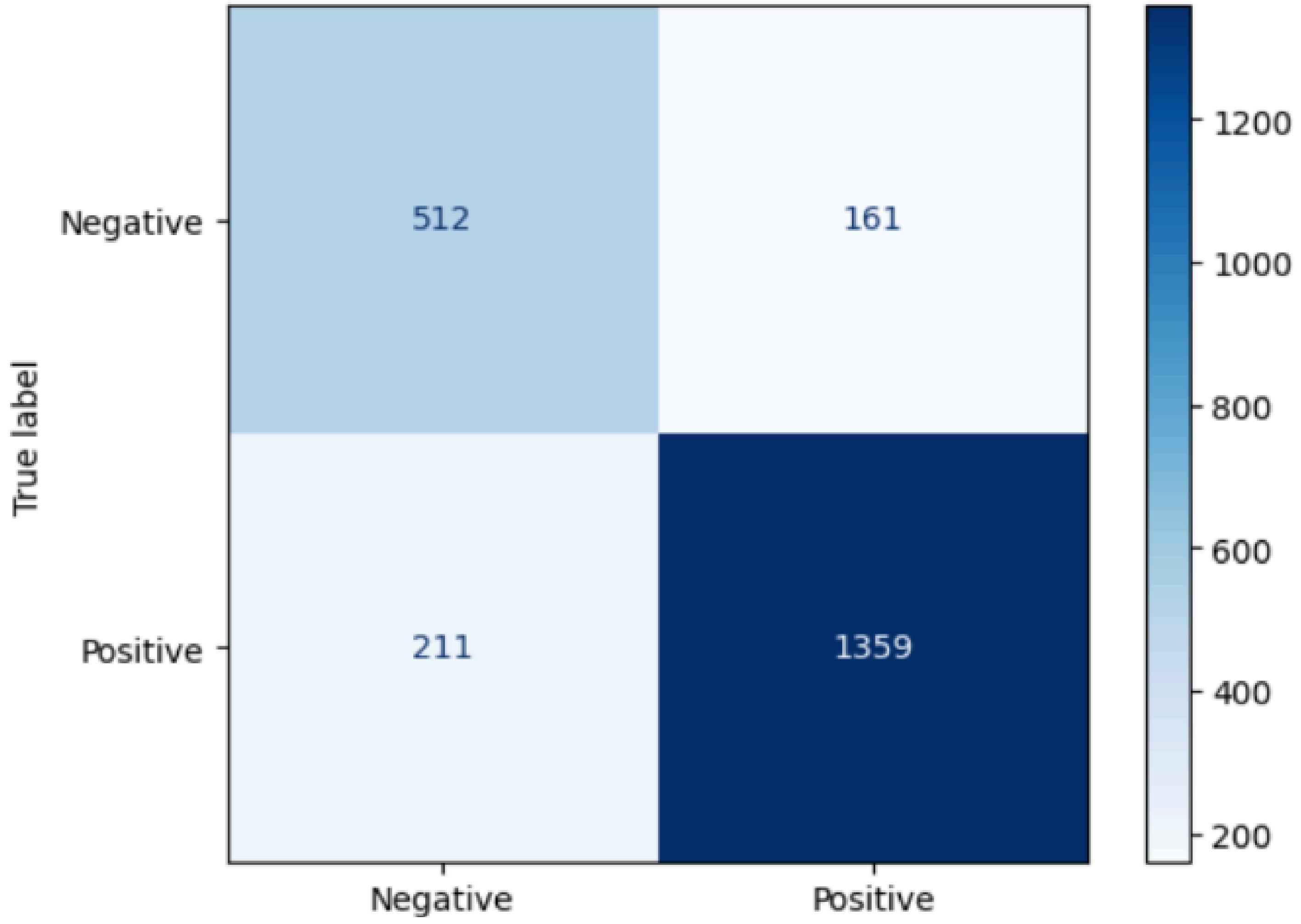
Aspects to texts is mapped so that:

1. If the same aspect appears more than once or if there is only one aspect per document, the entire text is associated with that aspect.
2. If an aspect appears only once and there are multiple aspects, the method checks if **keywords** associated with the aspect are present in the text.
3. If any keyword is found, the text is associated with that aspect provided it surpasses the **threshold** of containing at least 1 word from the dictionary.

Disclaimer: in a some instances, aspects could not be mapped due to absence of matches between dictionary and cleaned text, so such documents were removed before re-runing the trained classifier, resulting in a reduction of samples.

Predicting sentiment for aspects

	precision	recall	f1-score	support
Negative	0.71	0.61	0.66	224
Positive	0.88	0.92	0.90	714
accuracy			0.85	938
macro avg	0.79	0.77	0.78	938
weighted avg	0.84	0.85	0.84	938



1. Improved precision scores, especially for negative sentiment class.

2. Overall accuracy score increase by 0.03

KEY TAKE-AWAYS

1. Slight improvement in accuracy achieved together with a precision score increase.

2. Need to try other methods to reshape aspect dictionary or more advanced techniques, such as NNs (BiLSTM-CRF, transformers?) Perhaps limited document size and vocabulary hindered the progress.

3. Consider improving preprocessing steps.

Thank You