Text Mining and Sentiment Analysis Final Project

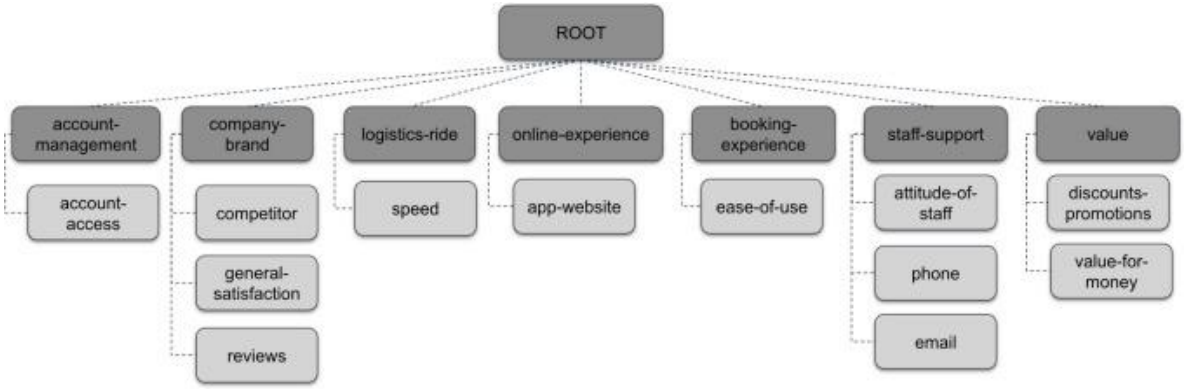
Explain Your Opinion: The Use of Born Classifier for Aspect-Based Sentiment Analysis

Arina Lopukhina (17169A) e-mail: arina.lopukhina@gmail.com

Abstract:
The project aims to perform ABSA on the FABSA dataset collected from Trustpilot, Google Store, and Apple store, containing user reviews across different sectors. The approach leverages feature extraction techniques to process textual data and applies a multi-label classification model to capture the nuanced aspects of feedback. To achieve this goal, Born classifier is employed to first assess only the sentiment of the text samples, and then the trained classifier is applied to detect different aspects. The classifier demonstrates robust predictive capability in several categories, though challenges persist in accurately predicting sentiments in less represented dimensions.

1. Introduction

Aspect-Based Sentiment Analysis (ABSA) is a natural language processing technique that identifies specific aspects or features within a text and determines the sentiment expressed toward each of those aspects. This project aims to explore the possibility of ABSA implementation using the Born classifier model. The dataset selected for this experiment is the new FABSA, a multi-domain dataset consisting of ~10,500 feedback reviews in English from across 10 domains (Fashion, Consulting, Travel Booking, Ride-hailing, Banking, Trading, Streaming, Price Comparison, Information Technology, and Groceries). The materials were collected from three main sources- Trustpilot, Google Play, and Apple Store, so they represent a mix of app reviews as well as feedback for different products and services. The dataset has been manually labeled to attribute the comment to one or more aspects, considering value, staff support, overall experience, logistics, company brand, and some more. The complete definition of different aspects is defined as follows:



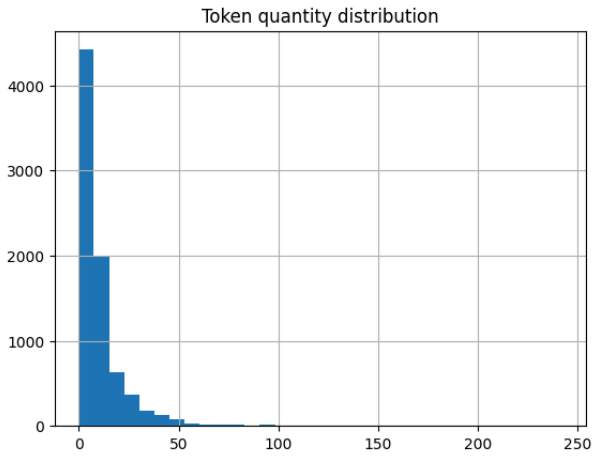
Additionally, the dataset consists of source information, unedited comments, assigned aspects, and assigned sentiment, which can be -1 for negative, 0 for neutral, and 1 for positive. There is a multi-label relationship as the same comment can be often attributed to more than one aspect. You can see the sample of the original dataset 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']

2. Data Preprocessing

The original dataset is already split into separate data frames for training and testing purposes. The training dataset consists of over 7000 rows, while the test dataset has a little over 2000 rows.

The data contained numerous emojis, typos, slang, and other linguistic elements that needed to be addressed before doing any analysis. Each cell of the text column has been stripped of URLs, digits, and other special characters, then the cases were normalized and stop words were removed. The text was tokenized into words, and each word is assigned a part of the speech tag. Finally, the text was lemmatized using the NLTK library. Overall, most comments had limited vocabulary and contained only a few tokens, as can be seen below:



Then, for future classifier training, a separate label column needed to be created. A binary variable was formed, defining whether a text was overall positive or negative. Additionally, unique aspect labels of each sample text were gathered in a separate column for future analysis and were used to create dummy variables for each aspect. The same exact processes were employed to the test dataset. Finally, X and y test and train variables were created and then

vectorized using the TF-IDF method as it was necessary to keep weights strictly above or equal to 0. Within the vectorization step, too common and too rare terms were filtered out using frequency threshold as well as n-grams were introduced to allow multi-token terms between 1 and 3 words long.

3. Born Classifier Methodology

As per Guidotti, E., & Ferrara, A. (2022), Born classifier is an approach for text classification that leverages principles from quantum mechanics, specifically the Born rule. Although the mechanism behind it is not fully understood, the Born rule defines the probability of a quantum state collapsing into a particular observable state. In the Copenhagen interpretation, the modulus squared of the inner product is interpreted as the (unnormalized) probability of the the wave function ψ to collapse in state s:

$$P(\psi \to s) = |\langle s|\psi\rangle|^2 = |\psi s|^2 \quad (1)$$

To put it into sentiment analysis terms, the classifier models documents as superpositions of words, treating text as a wave function where each word contributes to the overall probability distribution. Born classifier treats documents and classes as quantum objects, and the classification process involves computing transition probabilities between the document's wave function and each class's wave function. Therefore, the class with the highest probability is selected as the predicted category.

During the training phase, each class is represented as a wave function based on the conditional probabilities of words given the class. The conditional probabilities are computed from the training data, enabling the classifier to learn how likely a particular word is to appear in each class. Moreover, the Born classifier employs regularization to reduce the influence of irrelevant features in classification by introducing entropy-based reweighting, shrinking the contribution of irrelevant addends towards 0. The theory behind regularization of probabilities works as follows:

$$u_k = \left(\sum_j H_j \sqrt{P_{j|k} x_j}\right)^2 \quad (2)$$

where H stands for entropy (which we aspire to minimize), x is a feature vector, and P is the conditional probability of feature j given class k.

Lastly, equation (3) generalizes the model by introducing hyperparameters a, b, and h. This allows flexibility in how the classifier computes the probability of a class given a document:

$$u_k = \left(\sum_j H_j^h W_{kj}^a x_j^a\right)^{1/a} \text{ with } W_{jk} = \frac{P_{jk}}{\left(\sum_{j'} P_{j'k}\right)^b \left(\sum_{k'} P_{jk'}\right)^{1-b}} \quad (3)$$

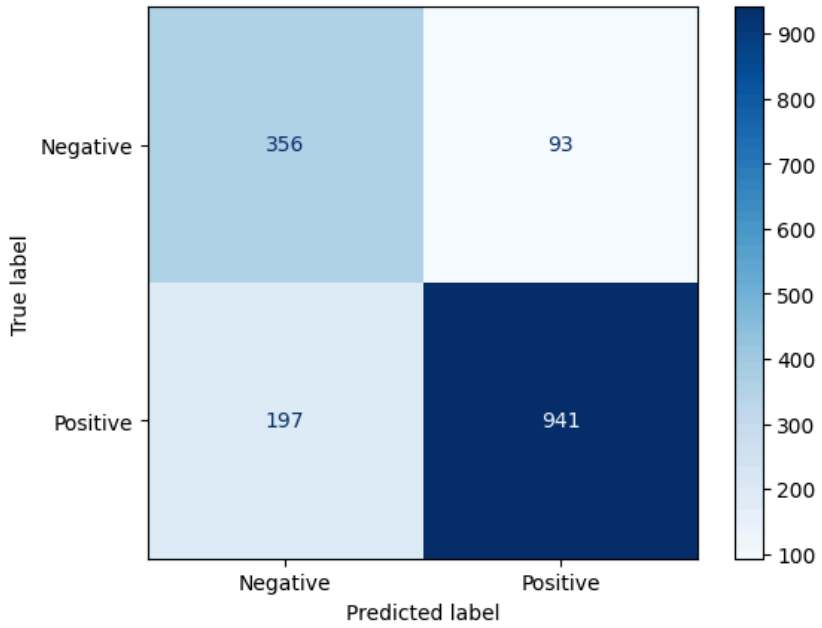
where *a* controls the weight of feature values, *b* affects the conditional probabilities, and *h* controls the importance of entropy-based reweighting.

In particular, when a=1, b=0, h=0, it reduces to a classical probability-based model, providing a comparison against the Born quantum approach. In the original model, which is used for the project implementation, we use weights as follows - a = 0.5, b = 1, h = 1.

4. Training Born Classifier

After the necessary preprocessing steps, both classical probability model as well as quantum probability model were employed for training the classifier to recognize positive and negative sentiment. The latter has performed much better, resulting in accuracy of 0.82. The model was better at recognizing positive attitudes, likely due to the larger sample size representation, although it is believed that Born classifier deals well with unbalanced classes.

	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



Other weight combinations were attempted, but at the end it was decided to continue with the model weights described in the original paper.

5. Token importance for sentiment definition

The initial results were investigated using the *explain()* attribute, which allowed us to understand which tokens had more weight in the decision-making process of the model. For every word, we could see its probability of collapsing to positive or negative class, hence we could define top-most-important tokens. It has shown that most important negative tokens were:

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

On the contrary, the following ones were attributed as most important positive words:

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

The examples above show a good level of distinction between positive and negative vocabulary, thus further suggesting that the classification outcome of Born is quite successful.

6. Defining aspects

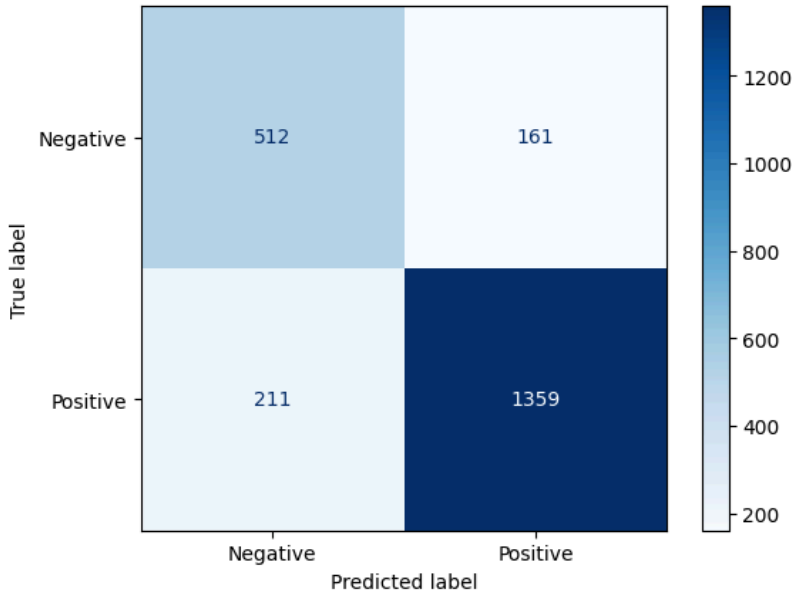
To add the aspect dimension to sentiment analysis, Born classifier was utilized to understand which tokens are the most important for defining a certain aspect so that we could create a dictionary and successfully map a relevant portion of text to the right label. The aspect labels were already assigned to each comment by the dataset creators (as follows - 'account management', 'company brand', 'logistics rides', 'online experience', 'purchase booking experience', 'staff support', 'value'), so there was no need to carry out additional unsupervised learning analysis to define a list of aspects. Consequently, it was necessary to define which portion of text contained details which could be attributed to one aspect or another.

To understand that, Born classifier was employed using the aspect label as predicted outcome and *OneVsRestClassifier,* which allowed the model to recognize each of the aspects simultaneously. Although for less represented classes the accuracy scores left more to be desired, we could then get the tokens that had the highest probability of being associated with a given aspect. They were later used to map cleaned text portions to the associated aspect labels. One issue encountered, however, was that sometimes due to imperfect aspect prediction there was no text-to-aspect mapping identified, so such samples needed to be removed from the dataframe used for further classification.

7. Predicting sentiment using trained Born classifier

Having defined text and aspect label mapping, the Born classifier (previously trained to define positive and negative sentiment using entire comments which could be linked to multiple aspects at once) was employed on the separate portions of text associated with certain aspects. This resulted into rather positive outcomes:

	precision	recall	f1-score	support
Negative	0.71	0.76	0.73	673
Positive	0.89	0.87	0.88	1570
accuracy			0.83	2243
macro avg	0.80	0.81	0.81	2243
weighted avg	0.84	0.83	0.84	2243



Notably, the previous steps helped visibly improve precision score for negative sentiment identification, which used to be only 0.64. However, the overall accuracy score of the model has not changed drastically and rose only by 0.01. Nonetheless, it was important to see any kind of improvement, and it is likely that finding a more optimal way to map aspects to text would have resulted in better model performance.

Sample outcome summary of the last classification model application can be observed below:

```
True Label: {'purchase booking experience': 1}, Predicted: {'purchase booking experience': 1}
True Label: {'staff support': 1, 'online experience': 0, 'purchase booking experience': 1, 'value': 0}, Predicted: {'staff support': 1, 'online experience': 1, 'purchase booking experience': 1, 'value': 1}
True Label: {'company brand': 1}, Predicted: {'company brand': 0}
```

8. Conclusion

Overall, Born classifier has demonstrated good results when applied for sentiment analysis purposes, and its performance is well in line with other classification algorithms such as SVM or Logistic Regression. The main challenges that have been identified, however, are the struggle to address multi-label context, as well as to deal with unbalanced classes. It appears that the dictionary-based approach to aspect-definition was not sophisticated enough to capture all the nuances, hence the re-application of the trained classifier had a limited impact. In future applications of the Born classifier, it would be interesting to further explore its potential to deal with multi-class predictions, perhaps as a layer in a neural network, combining it with other techniques. Lastly, perhaps, a more elegant alternative of employing meta-strategies for multi-class classification could be found, apart from the One-vs-Rest approach explored in the project. Nonetheless, despite the challenges and limitations, the project results provide an insightful outlook on the real-world data ABSA application.

Bibliography:

1. Arora, A., Gupta, R. K., & Sharma, D. K. *FABSA: A Dataset for Aspect-Based Sentiment Analysis*. Papers with Code. Retrieved from https://paperswithcode.com/dataset/fabsa.
2. Guidotti, E. (n.d.). *Born Classifier Documentation*. Retrieved from https://bornrule.eguidotti.com/sklearn/#bornrule.BornClassifier.explain.
3. Guidotti, E., & Ferrara, A. (2022). *Text Classification with Born's Rule*. NeurIPS Conference Paper. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2022/hash/c88d0c9bea6230b518ce71268c8e49e0-Abstract-Conference.html.
4. Rana, T. A., & Cheah, Y. N. (2016). Aspect extraction in sentiment analysis: comparative analysis and survey. *Artificial Intelligence Review, 46*(4), 459-483.