Classifying Beauty Products' Reviews

Group 11: 95617 Juliana Yang and 95618 Leonor Barreiros

When we give our opinion about something, there's a subjective, underlying sentiment associated - and it can be ranked. Sentiment Analysis is the NLP field dedicated to studying that. This article proposes a simple and effective algorithm for classifying beauty products' reviews in: =Poor= (worst), =Unsatisfactory=, =Good=, =VeryGood=, or =Excellent= (best).

	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6
Base. 1	0.3115	0.3215	0.3215	0.3090	0.3115	0.3300
Base. 2	0.4780	0.4815	0.4710	0.4590	0.4600	0.4630
JLLR	0.4955	0.5000	0.4870	0.4720	0.4860	0.4860

Table 1: Experimental results of 6 runs (accuracy) for the learned model, in comparison with the baselines.

1 Models and Experimental Setup

1.1 Proposed Model

We preprocess the data by applying lowercasing, lemmatization and tokenization (using, respectively, NLTK's WordNet Lemmatizer and Word Tokenizer), common approaches when dealing with unstructured data such as natural language - as explained in [4]. We performed feature engineering, by extracting new features from the data to improve performance. Each review has the following features:

- **Raw-count term-document vector:** used in constructing a term-document matrix, whose features will indicate how similar documents are;
- **Nº of adjectives:** (using NLTK's POS Tagger) - adjectives are "*(...) important indicators of opinions*" [3];
- **Presence (or not) of negative words ("no", "not", "n't"):** negations can change the polarity of a sentence (for example, "good" has a positive connotation, whereas "not good" has a negative one);
- **Nº of "!":** the use of exclamation marks may indicate the sentiment's intensity;
- **Nº of ":)" and ":(":** ":)" is perceived as positive, and ":(" as negative [1];
- **Nº of positive/negative/neutral words:** (determined using NLTK's VADER [1] [2]) if the sentence's meaning is compositional, then its sentiment may also be composed by the sentiments of its words;
- **Being (or not) a positive/negative sentence:** (also with VADER) the overall polarity of the sentence.

Finally, the model consists of a Logistic Regression (LR) classifier which predicts, given the features for reviews, their classification: we call it ***JLLR***.

1.2 Experimental setup

There are three steps to our research (which follows the scientific method): getting to know the field and formulating our goal and hypothesis, designing the experiments to test our model, and implementing and evaluating the system. The target of this research is to evaluate how well a model can identify the sentiments in a sentence. To do this, we suggest the aforementioned model, with the goal of classifying reviews better than a baseline. Therefore, our hypothesis is: ***the JLLR model labels reviews better than the baseline***.

We conducted our experiments on the given dataset, which contains 10000 samples - the training set. In each run, we extracted a development set of 2000 samples (20%), which simulates a test corpus and enables evaluation (we did this 6 times with different random states, to avoid overfitting). We didn't perform cross validation because we can learn from development sets and don't need to spend the extra resources on multiple rounds. We performed intrinsic evaluation, as we tested our system alone, firstly, with accuracy. After obtaining a reasonable value, we looked at other metrics: confusion matrix, accuracy by label, and manual observation.

We implemented several variations, with different preprocessing techniques (for example, stop words removal) and feature combinations (including some we ended up not using, such as the presence of "..."). We also evaluated other models, like Naive Bayes and Support Vector Classifiers. We conducted an iterative process of testing and reformulating the hypothesis, until obtaining the best model - the proposed one.

2 Results

In **Table 1** are the results of baseline 1 (Jaccard), baseline 2 (Support Vector Classifier and CountVectorizer) and JLLR - considering accuracy. Our model outperformed both baselines, in all runs.

In **Figure 1** is the confusion matrix for one of the development sets we evaluated with. Our model was better at detecting the most extreme classes, and mistakes were more frequent in neighbouring labels.

In **Figure 2** are the model's and the baselines' accuracy by label. Although only being the best in one of them, our model is best as it's the most stable. Baseline 1 performs poorly on the extreme layers and the =Good= one; and although baseline 2 has similar results to JLLR, it performs significantly worse on the =Unsatisfactory= and =Good= classes.

3 Discussion

Our model performed the best on the most extreme labels: =Poor= and =Excellent=, as they have the most correctly classified reviews. From the confusion matrix we can verify JLLR's low sensitivity in differentiating neighboring targets: which is responsible for the lower accuracy in the middle categories. If, instead of 5 labels, our model classified reviews according to only 3 classes, we expect obtaining significantly better results.

To better study the faults in our model, we analyzed the frequent mistakes it made.

Firstly, "good" and "nice" are learned as positive words.

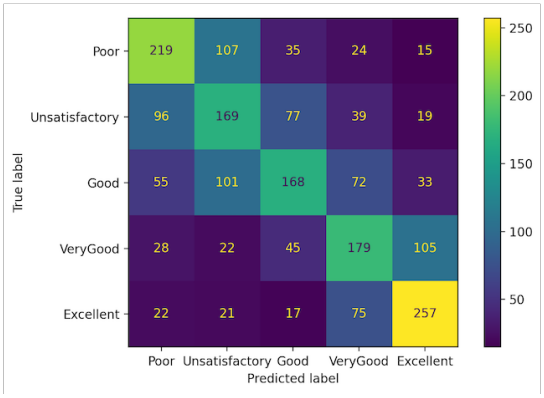

Figure 1: Confusion matrix for (one of the runs of) our model.

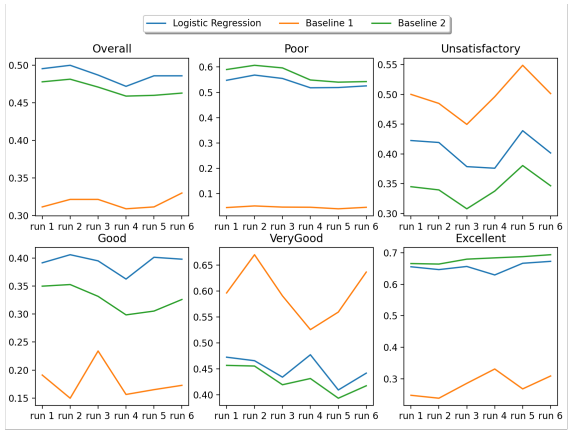

Figure 2: Our model's and the baselines' performance (considering accuracy) by label.

So, when JLLR finds a sentence with them (without negatives or "but"s), it labels it as =VeryGood=. However, there are =Good= (around 11% of the class's errors), =Poor= (around 6%) and =Excellent= (around 16%) samples which were incorrectly classified as =VeryGood=. For example, the review "Pretty good" was =Good=, "Color good" was =Poor=, and "Nice Set for the Price" was =Excellent=.

Secondly, negative words ("no", "not", "n't") are comprehended as having a negative polarity. The model, consequently, associates them with negative reviews (=Poor= and =Unsatisfactory=). There are =Excellent= (around 7% of the class's errors), =VeryGood= (around 8%), and =Good= (around 25%) sentences labeled negatively. "Doesn't Sting, Smells Wonderful" was =Excellent= (and not =Poor=, as it was labeled), "Not Going Back to My Old Hair Dryers!" was =VeryGood= (but was classified as =Unsatisfactory=), and "No complaints" was =Good= (but labeled as =Poor=).

Finally, although our model made some *honest mistakes*, the dataset it learned from also had some difficult characteristics. For example, "Pretty good" (sometimes, with different casing on some letters - which isn't detected since the model performs lowercasing) was labeled as =Poor=, =Good=, or =VeryGood=, which becomes incoherent when training. Another example is "Ok" (also with different combinations for casing), which was classified as =Unsatisfactory=, =Good=, or =VeryGood= by different reviewers.

Ours	Ours and model's (avg)	Ours and dataset's (avg)
0.3080	0.4495	0.3925

Table 2: Comparison of agreements between different annotations: ours, ours and the dataset's, and ours and our model's.

Moreover, there are some items whose label we don't agree with or aren't sure which would be the most correct one. The review "E.L.F Primer" (and others which were just the name of the product) was =Poor=, although there are no distinctive elements which indicate that sentiment. Another picturesque example is "good" which was classified as =VeryGood=.

So, we intend to prove that the task of classifying these reviews is subjective and, for that reason, difficult for a machine to perform with large success (i.e. higher than 50-60%). For that reason, we collected a small (20 sentences) random sample of one of the development sets and manually assigned a class to each of them. Then, we calculated inter-annotator agreement (Cohen's kappa coefficient) between our annotations, each of our annotations and the dataset's and each of our annotations and the model's, depicted in **Table 2**. Our agreement, and ours and the dataset's is Fair, however, our and our model's is Moderate. These values mean that performing this task is complex (because all levels weren't high) but, more importantly even, they show we agree more with our model's classification than the dataset's (real) labels.

4 Future Work

We intend to do a finer study of the dataset and evaluate whether it's possible to extract more features. For instance, by learning a pattern in our model's mistakes, and re-teaching it not to make them. Furthermore, since features aren't independent, we also wish to research which combination of features would result in the best performance. Finally, since the baselines outperformed our model in some classes, we aim to create a hybrid model which would combine the best characteristics of all of them.

Notes

1. **V**alence **A**ware **D**ictionary for s**E**ntiment **R**easoning is a rule-based tool developed for web-like text - a similar nature to our reviews'. If it were, for instance, built using text from books, it wouldn't be useful.

References

[1] Pollyanna Gonçalves, Matheus Araújo, Fabrício Benevenuto, and Meeyoung Cha. Comparing and combining sentiment analysis methods. In *Proceedings of the first ACM conference on Online social networks*, pages 27–38, 2013.

[2] Clayton Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225, 2014.

[3] Walaa Medhat, Ahmed Hassan, and Hoda Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113, 2014.

[4] S Sumathi and G Hannah Grace. A similarity measure for text document using term cardinality. In *IOP Conference Series: Materials Science and Engineering*, volume 1012, page 012059. IOP Publishing, 2021.