Using SVM to Classify Amazon/Yelp/IMDb Review Sentiments

Introduction

This project aims to create a classifier able to predict sentiments from a dataset of sentences derived from customer reviews from three different websites: amazon.com, imdb.com and yelp.com.

My method is based on the traditional Machine Learning approach, specifically supervised learning with Linear Support Vector Classification (Linear-SVC).

The dataset contains 3000 manually labelled instances, 500 positive and 500 negative from each website. I trained my classifier using the data in different ways and selecting different features in order to explore the results and be able to discuss the best scores. For instance, in one of my approaches I considered the corpus as a whole set of 3000 sentences, in another one I separated the genres and trained and predicted on the three sets separately. I will provide results coming both from character-based and word-based feature selection.

Finally, I will show some examples of sentences predicted incorrectly and will speculate on the possible reasons why the classifier fails to label them correctly.

The dataset is derived from the paper 'From Group to Individual Labels using Deep Features' (Kotzias et. al,. KDD 2015). I will introduce this paper, their goal, methodology and achievement in the next section. Their experiment results allowed me to compare the scores of the prediction accuracy of two completely different approaches, namely supervised and unsupervised learning.

Paper: *From Group to Individual Labels*

Overview

The study described in this paper aims at finding a good method to classify sentiments at the sentence level starting with a group of instances dataset, in this case the reviews from Amazon, Yelp and IMDb.

Reviews allow us to have fast and easy labels at the group level, either because the total sentiment of the review is already explicit or because it is relatively quick to label a review manually as positive or negative. It is instead much more time consuming to label each sentence within a review. Nonetheless, sentence sentiment happens to be very useful data, for instance when it comes to improving customer service.

A company might be interested in knowing specifically what the customers are dissatisfied with, and also users might benefit from performing detailed feature queries when looking for a particular quality in a product or service.

As shown in Figure 1, if a review is only globally labelled as positive or negative this kind of information gets lost.

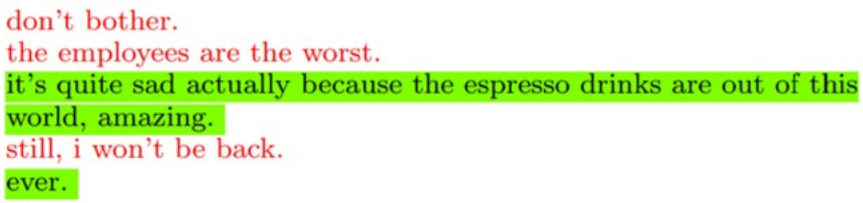

Fig 1

This is a globally negative review from the Yelp dataset, with a very positive sentence in the middle. The red sentences are labelled as negative and the green ones as positive (although we could argue that the sentence "ever." in this case is clearly negative, but this can safely be inferred only from the context).

This idea is applicable to a wider range of problems in different fields.

Method from Kotzias et. al.

Given that a review is often composed by positive and negative comments, they propose a method that is able to go from the review sentiment to sentence sentiment by using instance similarity and at the same time taking into account the review global label.

Since the reviews are already labelled and they make use of this data in their training, we can probably consider this as a half supervised/half unsupervised learning experiment. All the sentences have been also manually labelled, but only in order to evaluate their results and not for model training.

The core of their new method, which they named GICF (Group-Instance Cost Function), consists in having created a new cost function that measures how close the predicted values are to their corresponding real values and is optimized by stochastic gradient descent.

After having trained the model in a supervised way by using the document labels, they provide vector representations of the sentences (which are used in the cost function and which capture the semantic relationship of words) through embedding techniques (convolutional neural networks).

In a few words, their cost function leverages instance level similarity and group level label information.

For their experiment, they consider the three datasets of reviews separately. They clean the data by breaking the review into sentences and then sentences into words. They get rid of HTML markup, and map numbers, symbols and infrequent words to their specific generic token.

The three datasets are then processed with slightly different methods (example: the size of the convolutional network).

Results from Kotzias et. al.

For the evaluation of their results they compare their accuracy level with the two following baseline methods.:

1- An L2-regularized logistic regression classifier on a bag of words representation at the document level, then they applied to bag of words representation of bag of words sentences.
2- Same logistic regression but with embedding vectors as features.

As shown in Figure 2, their results demonstrate that the approach they propose offers a better accuracy level across all the datasets than the two baselines.

	Amazon	IMDb	Yelp
Logistic w/ BOW	79.0%	76.2%	75.1%
Logistic w/ Embeddings	54.3%	57.9%	66.5%
GICF w/Embeddings	**88.2%**	**86.0%**	**86.3%**

Fig 2

The logistic regression with bag of words obtains much higher results than the logistic with embeddings, but still can't achieve the GICF accuracy. They also compare their method to a previous one (Socher et al.), and once again their accuracy is significantly higher.

They have therefore shown that their proposed method is successful when transferring information from the document level to the sentence level when predicting review sentiments.

They also manage to get a good accuracy score when they evaluate their model as a group classifier, namely when they utilize the aggregation function on the predicted sentence score, in order to predict the global review sentiment.

My Project

For my project I created four different Python scripts that process the data in different ways.

In two of them I consider the 3000 sentences as a single corpus, and in the others I train and predict on the three separate sets of reviews (1000 sentences for each set). The sentences have been selected randomly from larger datasets of reviews.

The following are three examples of sentences in the dataset, each one from a different website and their corresponding labels:

- *I have to jiggle the plug to get it to line up right to get decent volume.* *NEGATIVE*
- *Not to mention the combination of pears, almonds and bacon is a big winner!* *POSITIVE*
- *The movie had you on the edge of your seat and made you somewhat afraid to go to your car at the end of the night.* *POSITIVE*

Note that the labels are either positive (1) or negative (0), no neutral scores are present.

The starting format of my dataset is [sentence \t score \n], so I proceed normalizing the data by getting rid of tab and new line symbols and by reducing characters to lower case. Unlike the cited paper, I don't get rid of punctuation/symbols or infrequent words.

When I use the whole corpus, I also reshuffle the sentences in the document every time when running the code, so the results slightly change each time.

Since I want to select features at the character level and also at the word level I proceed in the following way:

- In two of the scripts I create a dictionary with the frequency of character-based ngrams (namely, in my results scores, bigrams, trigrams and 4grams). I apply this to the whole corpus and on the separate sets.
- In the other two scripts I simply tokenize the sentences word by word and create a dictionary with their frequency. I apply this to the whole corpus and on the separate sets.

In all cases, I split the data in 80% training sentences and 20% test sentences.

Then I use Scikit-learn to create a pipeline that:

1- maps the feature-value mappings to vectors by using DictVectorizer;
2- normalizes samples individually to unit norm (l2 Normalizer);
3- implements TF-IDF;
4- trains the linear SVC.

Using TF-IDF is important because it shows the relevance of each feature by giving a value that increases proportionally to the number of times a feature appears in the document but is balanced by the frequency of the feature in the corpus. As you will see in the results section, I also compare the score of the prediction where TF-IDF is not used in the training.

I chose Linear Support Vector Classification, which is an implementation of Support Vector Classification for the case of a linear kernel.

Finally, I predict, evaluate the accuracy and provide a confusion matrix.

Results

The graph in Figure 3 shows the level of accuracy achieved when I train and predict on the three sets of genres separately. I compare three results for each set, the word-by-word approach, the character based one (specifically, the trigram-based) and the accuracy achieved by the experiment of Kotzias et. al. using their GICF method.

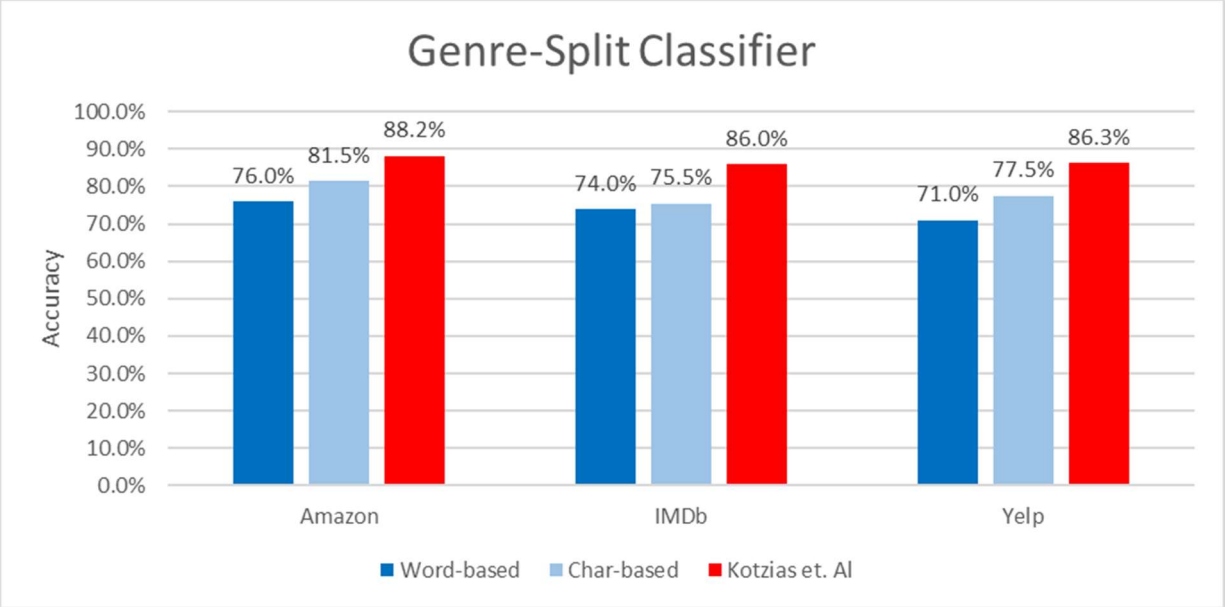

Fig 3

As it can be seen, between my two approaches, the character-based gives better results in the case of Amazon and Yelp datasets and only slightly better result in the case of the IMDb one. My score is still significantly lower than the GICF method score, but it looks respectable when compared with the two baselines used in Kotzias's paper and shown in Figure 2.

The graph in Figure 4 shows the different accuracy scores obtained when training the corpus of 3000 (shuffled) sentences as a whole, without any distinction of genre, on different features: bigrams of characters, trigrams of characters, 4grams of characters, word-by-word, and trigrams of characters without using TF-IDF.

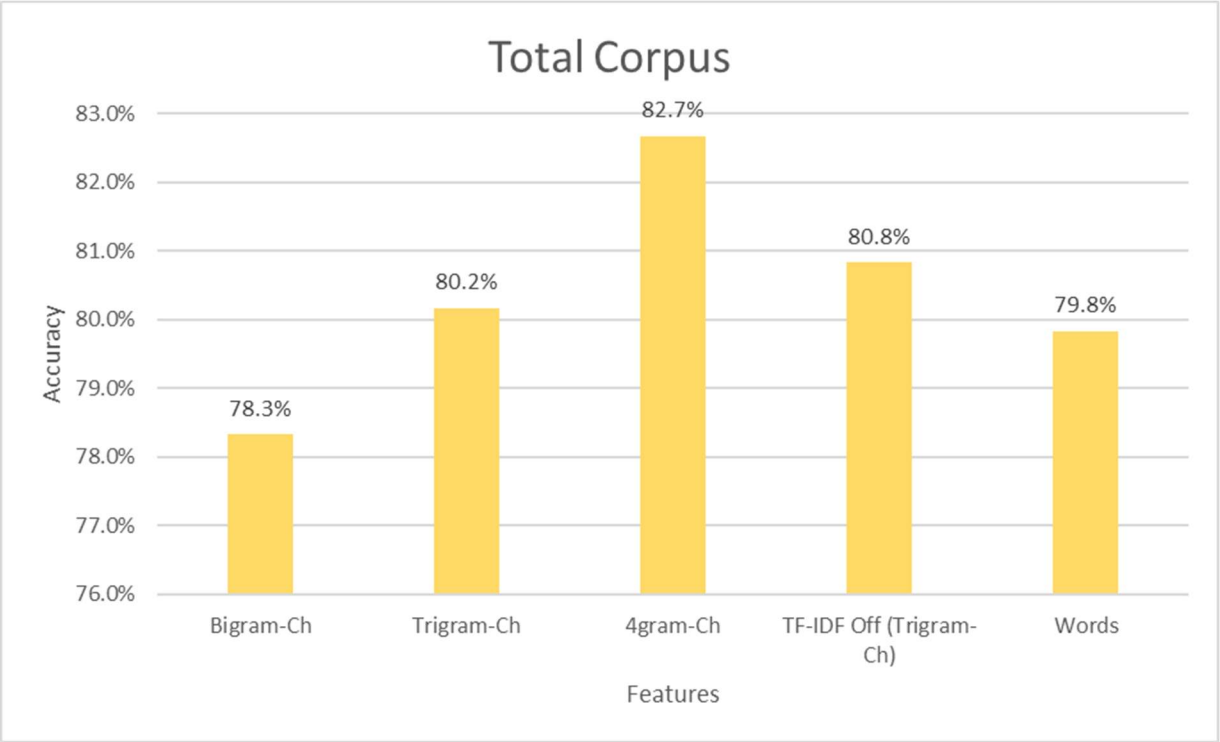

Fig 4

The best score in this case goes to the 4gram character-based classification. It is nevertheless notable that the word-based approach performs worse than almost all the others (except the bigram). It reveals itself to be even less accurate than the trigram-based classification that does not use TF-IDF.

Curiously enough, using TF-IDF on the trigrams of characters does not improve the accuracy score, even though the difference is not big. It would be interesting to see how well it performs on the other features, as a further study.

Analysis of Some Predictions

The analysis of the incorrect predictions shows some interesting and somehow amusing outcomes that are worth being discussed. I will therefore report some of the examples that I find relevant from the linguistic point of view.

From the word-based classification of the whole dataset, one of the incorrect sentences is the following:

It's so bad it's actually worth seeing just for that reason.

This sentence has been manually labelled as negative but it has been wrongly predicted as positive. It is maybe inferable that, since the words “worth seeing” without the negation are usually associated with only positively connoted reviews, the classifier safely predicts it as a positive review. The words “so bad” were maybe not enough to overcome the ambiguity that human language together with a sarcastic approach are able to create. Someone might argue that this sentence can also be considered as a positive review.

The next example is:

A couple of months later, I returned and had an amazing meal.

This sentence from the Yelp dataset is manually labelled as positive, but my classifier labels it as negative. Can this happen because of the word “returned” which is usually strongly negative when used in the context of “returning an item to the store”? Since this result belongs to the mixed genres approach, I assumed that the Amazon reviews influenced the classifier decision and checked the result in the genre- separated corpus, only to find that even in that case the sentence is incorrectly labelled. It is only the separate-genre character based (trigram) approach that manages to get it right.

It would be interesting to find out if this happens because of the lemmatization occurring when considering ngrams instead of words (the word “returned” doesn’t appear often in the whole corpus, but the word “return” does).

Just as a speculation (that might or might not be true), a tokenization that takes into account the lemma instead of the word might also influence the score in the case of words belonging to different languages but sharing the same root (as “delicioso” in one of the reviews).

The following sentences are instead random examples of incorrect classification from which I cannot really draw any linguistic conclusion from, showing that not every decision is explainable with human logic:

The staff was very attentive.

(Manually labelled as positive, classified as negative)

This place should honestly be blown up.

(Manually labelled as negative, classified as positive)

I vomited in the bathroom mid lunch.

(Manually labelled as negative, classified as positive)

And finally a curious example:

This short film certainly pulls no punches.

(Manually labelled as negative, classified as positive)

This sentence has been manually labelled as negative, but this judgment is quite debatable. In my personal view this sentence is positively connoted, because the user is saying that the film really got their message across. The classifier, though, labels it as positive. This example can maybe show that Machine Learning can help overcome human arbitrariness.

Conclusion

The classifier used for this project manages to achieve its best accuracy (82.7%) when trained with the 4gram character-based feature approach on the whole corpus of 3000 sentences, without the genre separation. It is still an improvable score, especially when compared to the accuracy scores derived from the paper of Kotzias et. al. with their semi-unsupervised method.

The analysis of some incorrectly classified sentences shows that linguistic intuition can explain only to a certain extent why the classifier fails, nevertheless several ideas about how to improve the scores can be derived.

It might be relevant to explore other kinds of tokenization, for instance by taking into account the lemma of the word or by exploring other similar avenues.

References

- Dimitrios Kotzias, Misha Denil, Nando de Freitas, and Padhraic Smyth. 2015. From Group to Individual Labels Using Deep Features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD '15). ACM, New York, NY, USA, 597-606. DOI: https://doi.org/10.1145/2783258.2783380
- Liu, B. (2011). *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers: Toronto.