**Project Report**

**“Los Ancianos”**

William Hanson

CS 487

New Mexico State University

Las Cruces, NM USA

[billh@nmsu.edu](mailto:billh@nmsu.edu)

Sanford “Jay” Johnston

CS 487

New Mexico State University

Las Cruces, NM USA

[spaztech@](mailto:spaztech@)nmsu.edu

Eloy “Frank” Macha

CS 519

New Mexico State University

Las Cruces, NM USA

[efmacha@nmsu.edu](mailto:efmacha@nmsu.edu)

# Introduction and Motivation

It’s no secret that the internet has become one of the most important ways that people use to both perform research and purchase goods and services. One of the key tools that potential buyers use is online reviews. An item or movie with an overwhelming number of positive reviews is typically considered a high-quality item and tends to attract the attention of potential buyers, while one with mostly negative reviews will be much less likely to be considered. Similarly, online reviews are one way for a company to receive user feedback on products. As noted in a recent Forbes Technology Council post, 97% of customers use online reviews as a way to find businesses. [1]

However, it is not uncommon to see reviews where the “star” or numerical rating does not match the actual review feedback. A 1-star (poor) rating may have glowing positive feedback, and vice versa. [2] These mis-marked feedback comments make the task of discovering and properly evaluating user feedback more difficult, possibly resulting in faulty purchasing decisions.

Similarly, there is a large problem with fake reviews – either reviews generated *en masse* by bots, reviews for other products (e.g., you are looking at a hammer, but the review is for a Bluetooth headset), or companies willing to outright pay for fake positive reviews designed to boost sales or negative reviews to sabotage competitors. As reported by CBS News, a study by the website Fakespot.com noted that 30% of Amazon reviews are either fake or unreliable.[3]

# Problem Definition

With many thousands, or even millions of online reviews on sites like Amazon.com, IMDB.com, yelp.com and others, an accurate review will influence purchase decisions, movie-watching decisions, or other engagement decisions. Mis-marked, fake or misleading reviews can have significant implications to those involved in the decision, and a manual review to validate these review ratings would be untenable for most businesses with significant numbers of products for sale. The problem at hand is essentially a classification task: How can we gauge whether a given review is positive or negative?

# Dataset

For the project, we obtained a curated dataset of text reviews, already classified as positive or negative. For our project, the “IMDB50K” dataset meets these requirements.[[1]](#footnote-1) This dataset contains 50,000 highly polar (positive/negative) movie reviews, already split between training (25,000) and testing (25,000) subsets. Reviews were evenly-balanced between positive and negative reviews in both sets. The only preprocessing necessary was to change the classification labels from ‘pos’ and ‘neg’ to (1, -1).

# Approach and Solution

The challenge we are attempting to solve is to accurately classify these online reviews as either positive or negative with a high degree of accuracy. In other words, we are performing sentiment analysis with an aim at classifying review ratings.

Given the course structure, we decided to focus on mainly using classification approaches from the course for this task. Our challenge, then was to try to achieve better than 0.9 accuracy for sentiment classification using “older” tools such as Logistic Regression, Multinomial Naïve Bayes, Multi-Layer Perceptron, Random Forests, and SVM. Because it would be unlikely that any single classifier would provide 0.9 accuracy on its own, we chose to use an “ensemble” approach, where each classifier would make individual predictions, and then final predications would be made by “majority vote” of the five classifiers.

The basic approach was to first tokenize and vectorize the reviews, and then to run those preprocessed vectors through their respective classifiers. Once all classifiers made their predictions, then the five sets of predictions were combined via majority vote.

Since we planned to use an ensemble voting approach for final predictions, we made a conscious decision to develop both vectorizing/tokenizing approaches and classifiers independently. This is because voting ensemble approaches work better when the predictions are independent from each other. By using separate approaches, this allowed the final predictions to be as independent as possible, given we were all working from the same data. This also allowed members of the team to choose classifiers and text processing methods that best met their individual coding needs.

Accordingly, several preprocessing methods were used to tokenize and vectorize the reviews. With that said, all shared common steps, including cleaning text, use of stop words, tokenizing and vectorizing.

Text cleaning generally involved removing special characters, html tags, single-character words, and changing all text to lower case. This also required removal of stop words, which are common words that do not convey relevant meaning for classification. Two different sets of stop words were used, the NLTK English stop word list, and a custom list developed from the review text. Additionally, some approaches used either stemming or lemmatization to further clean the data.

The most complex approach included cleaning, lemmatization, parts-of-speech tagging, sentiment embedding using VADER, and then embedding using doc2vec. The simplest approach used the Keras tokenizer, which includes limited stripping of unwanted characters. Other approaches took a middle ground via text cleaning, lemmatization and parts-of-speech tagging.

Once that was complete, reviews were vectorized, either via the Keras Tokenizer or the sklearn TFIDF Vectorizer. One set of important choices in this process was the use of minimum and maximum degrees of freedom to eliminate words that were either common to most documents or were present in only a very few. This allowed the classifiers to focus on words most likely to differentiate between positive and negative reviews.

Another important tuning choice was the use of n-grams, which capture meanings that would be missed by only looking at single words, e.g. “not good” vs “not” and “good.” After trying several variations, a combination of bigrams and unigrams gave the best performance for our classifiers.

Due to hardware limitations, a two-stage approach was used, with stage 1 (preprocessing) included reading in the data, cleaning it, and then tokenizing and vectorization. Once that was complete, the vectorized text was written to disk for stage 2. Stage 2 then read in the text vectors from disk, ran the appropriate classifier and wrote the predictions to disk. The final stage took all the individual predictions in and then returned final predictions based on a majority vote.

# Results

The final results met our accuracy goal, albeit not by much. The accuracy and f1 results of the five classifiers as well as the majority vote are shown below. Four of the five individual classifiers turned in results in the upper 80s, the Logistic Regression classifier performed best and the Random Forest classifier trailed at a still-respectable 0.84 accuracy/f1. Despite the different approaches to preparation and classification, it is apparent that there was not a lot of independence in these classifiers given the very small (0.01) bump in overall accuracy from voting. Had these results been completely independent, we would have expected the final accuracy to be around 0.98, or at the very least in the mid-90s. As such, the final result of 0.90 was a bit disappointing, even though it met our goal.

|  |  |  |
| --- | --- | --- |
|  | Test Accuracy | Test f1 |
| Log Reg | 0.89 | 0.89 |
| MLP | 0.86 | 0.86 |
| MNB | 0.86 | 0.86 |
| SVM | 0.88 | 0.88 |
| RF | 0.84 | 0.84 |
| Majority | 0.90 | 0.90 |

Table 1: Classifier Performance

Overall, run times for the preprocessing step reflected the complexity of the individual pre-processing steps. The Random Forest approach was by far the most complex, and took a whopping 1305 seconds to complete, and the resulting vector files were a hefty 9 Gb+. The two simplest approaches completed in 22 and 31 seconds, respectively.

Classifier runs also spanned a wide range, from as low as 1 second (Multinomial Naïve Bayes) to almost one hour for the SVM classifier. Run times are summarized in the chart below. Given the combination of run time and performance, the Logistic Regression classifier worked best on this data.

|  |  |  |
| --- | --- | --- |
|  | Preproc Time | Classify Time |
| Log Reg | 243 | 7 |
| MLP | 22 | 85 |
| MNB | 227 | 1 |
| RF | 1305 | 903 |
| SVM | 31 | 3434 |

Looking at the error rate of the voting results revealed the expected pattern. As anticipated, unanimous decisions were both most common (74%) and the most accurate at about 0.96. Decisions that were essentially split decisions (3:2) were the least common (11%) and least accurate (0.62). Decisions that were 4:1 splits made up 15% of the results and had an accuracy of 0.79. The results are summarized in the chart below.

As part of the analysis, we looked briefly at misclassified reviews to see if there were any immediate changes that could be made. Here is an example of a positive review that was incorrectly classified:

*I do not see what is the whole deal about this movie. Patricia Kaas sings, yes, and that makes the film charming, but singing is not enough. I mean, if you want to get all benefits of this one, go buy Patricia's CD and enjoy! The plot is simple (if to omit the "dreams"); the main characters seem to love each other instantly, and as well instantly forget their other acquaintances. Relationships appear almost "mandatory", and the little "detective story" thrown in just makes things worse. What at first appeared to be a perfect movie ending, is first screwed up and then comes out as just a dream. I measure movies by how well I would remember them, and for this one, I already started to forget the details.*

Here is an example of a negative review that was incorrectly classified:

*The real surprise of this effortlessly lightweight movie is how such a top notch cast got assembled for what is nothing more than a hammy uninspiring affair. Presumably it was a proverbial snowball rolling down a hill gathering pace and size and shape. One can imagine that by the time Miranda Richardson got contacted by her agent, the conversation went along the lines of: 'Do you want to shoot a movie in Dublin scripted by Neil Jordan? Michael Caine and Michael Gambon are already in!' This is a dull 'comedy' that sees Michael Caine and Dylan Moran try and pull off a well-planned hustle where Moran must imitate a London gangland boss (whose arrival is imminent) to collect a sizeable sum of cash from local kingpin Michael Gambon. The rest is simply a forgettable romp that is thankfully over quite quickly. Moran is mildly amusing in places but on this evidence is better suited to life on the small screen <text deleted>. Gambon actually steals the show, and anyone who has caught some of his performances in the likes of Have I Got News For You will know that he is a wonderfully funny man. But overall the result is disappointing, and it seems a lifetime ago that Neil Jordan was making quality movies of the likes of Mona Lisa.*

Even reading these reviews is somewhat confusing – both mix negative and positive comments, and the bottom-line decision seems to be buried in the verbiage. Even then, a reader coming at these reviews cold could be forgiven for reading these reviews as either negative, positive, or neutral. It is clear that a more nuanced approach to classification is required to correctly categorize reviews like these.

# Conclusions and Further Steps

This project had two objectives (besides earning a good grade). The first was to see if an ensemble of classifiers could achieve accuracy of over 0.90 on a sentiment classification task. Our approach combining five basic classifiers (Logistic Regression, Multi-Layer Perceptron, Multinomial Naïve Bayes, Random Forests and SVM) in a majority vote did achieve our goal – although not by much.

A brief look at misclassified reviews showed that at least the reviews we looked at were quite vague and rambling, making even human classification a difficult task. This was probably one of the reasons why the voting results did not greatly improve overall accuracy – any classifier would have trouble with these kinds of reviews. Clearly, a more sophisticated set of classifiers would be required to significantly improve accuracy.

Our second objective was to allow each member of the team to pursue specific tasks which would aid us in our professional lives. Subjectively, we believe that this project allowed us to succeed here as well, although such a judgement is clearly subjective.

The scope of an undergrad/MS semester project left limited room for additional “what if” and sensitivity analysis. For example, one step that could have been taken would be to “mix and match” pre-processing and classification approaches to see if we could gain better performance.

Similarly, while some tagging and embedding approaches were used, there are several tools (like spaCy and BERT) that could have been used to our advantage.

Another approach would be to further tune the classifiers to improve performance. One thing that we observed is, not surprisingly, that classifier performance improved slightly as larger portions of the dataset were used. Due to hardware limitations, initial tuning of the classifiers was performed using first, 5% of the data, and then 20% of the data. Since performance changed with increased size, it is likely that some additional hyperparameter tuning using the full set of training data would yield optimal results.

We learned that LSTM or CNN may yield better results, however, the necessary self-study to make these classifiers work robustly was beyond the scope and time limits of this course. Such an effort is probably better suited to an MS Thesis. We were able to get a CNN network approach built, but the results were disappointing and we were unable to markedly improve the results given time constraints.

In summary, *Los Ancianos* were able to meet our personal and group objectives through this project, and we appreciated the challenge. We only wish there was more time and opportunity to apply other approaches that we discovered during the course of the project.

**REFERENCES**

[1] Cory Capoccia. 2018. “Online Reviews are the Best Thing That Ever Happened to Small Business.” *Forbes Technology Council*. April 11, 2018. [https://www.forbes.com/sites/forbestechcouncil/2018/04/11/online-reviews-are-the-best-thing-that-ever-happened-to-small-businesses/](https://www.forbes.com/sites/forbestechcouncil/2018/04/11/online-reviews-are-the-best-thing-that-ever-happened-to-small-businesses/#2eb45c63740a)

[2] Nishit Shrestha & Fatma Nasoz. 2019. “Deep Learning Sentiment Analysis of Amazon.com Reviews and Ratings.” International Journal of Soft Computing, Artificial Intelligence and Applications 8, 1 (Feb 2019), 1-15. <http://aircconline.com/ijscai/V8N1/8119ijscai01.pdf>

[3] Aimee Picchi. 2019. “Buyer Beware: Scourge of fake reviews hitting Amazon, Walmart and other major retailers.” CBS.com. <https://www.cbsnews.com/news/buyer-beware-a-scourge-of-fake-online-reviews-is-hitting-amazon-walmart-and-other-major-retailers/>

1. http://ai.stanford.edu/~amaas/data/sentiment/index.html [↑](#footnote-ref-1)