Project 3 Report: Movie Review Sentiment Analysis

CS 598 Practical Statistical Learning, Fall 2023

Team Members

- **Kurt Tuohy** *(ktuohy)*: contraction expansion, token t-test, token exploration
- **Neal Ryan** *(nealpr2)*: notebook configuration, initial tokenization, lasso token regularization
- **Alelign Faris** *(faris2)*: mymain.py

Technical Details

We followed a similar approach to the instructors in Campuswire posts 626 and 628. The only deviation was expanding contractions (based on THIS work). The advantage of this is that negation is better captured. For example, "I don't enjoy" and "I didn't enjoy" both get tokenized as "not enjoy", increasing their predictive power while remaining interpretable.

Data Preprocessing

Contraction expansion is the first step performed and converts all contractions to their full English equivalents (don't -> do not).

Next, tokenization is performed, with a token window of 1-4. Apostrophes are retained, and other punctuation marks are treated as word boundaries and removed. Stop words from the custom list in post 626 are removed. Minimum term frequency is set to 0.1% to avoid one-off use cases. Maximum term frequency is set to 50% to avoid any inadvertent stop words ('the' or similar).

A t-test is then performed to explore tokens. This tests for whether tokens are from the "positive" or "negative" distribution of tokens, as these two populations are the most helpful for prediction of the same sentiment. The highest (absolute value) t-test statistic is used to capture the 2000 most predictive tokens from both positive and negative reviews. We fold in 9 additional tokens which appear only in positive reviews or only in negative reviews.

Lasso (L1) regularization is then implemented to reduce the vocabulary size to 1000 tokens with a C of approximately 0.045764.

Models Used

Logistic regression is used to make the final assessment of sentiment. This technique is extremely valuable in cases of binary classification due to the activation functions underlying shape (transitioning quickly between asymptotic end values). An l2 penalty is used, with a C of approximately 21.5443469.

Data Exploration

Similar values to those reported in Campuswire post 628 were found. The values all make enough sense and are fairly intuitive predictors of sentiment.

After regularization, the highest absolute value coefficients of the L1 model are shown below:

'7 10', '4 10', '3 10', '8 10', '3 out of 10', '4 out of 10', 'not recommend', 'well worth', '7 out', 'waste', 'mst3k', 'definitely worth', 'refreshing', 'disappointment', 'poorly', 'worst', '10 10', 'awful', 'not funny', 'unfunny', '2 10', 'redeeming', 'miscast', 'forgettable', 'dull'

It's perhaps not surprising that review scores like 7/10 (with the / removed due to punctuation deletion) are very predictive of sentiment, and the top 6 tokens reflect that. Most of the other tokens are very interpretable, with the exception of 'mst3k' which seems to refer to Mystery Science Theater 3000.

The inclusion of films being funny (or more accurately not funny for the prediction of negative sentiment) was interesting, and not a thought that immediately came to mind, but makes sense. It leads to the hypothesis that if categories were included (or mined through latent semantic analysis), we could likely perform even better with weights like "funny" being used for comedy, "suspenseful" being used in thrillers, or "casting" being used for dramas (e.g. weighting the importance of certain words in based on genre).

Performance Metrics

Weighted mean absolute error (WMAE)

Split	AUC
1	0.9616726083797218
2	0.9616009217810427
3	0.9610771609287114
4	0.9621485005750404
5	0.96203883948773

Execution time

Split	Run time (seconds)
1	186.57749891281128
2	187.39333772659302
3	187.07028603553772
4	191.3993124961853
5	187.2207088470459

Computer Specs

- CPU: Intel i7-7700
- Cores/threads: 4/8
- Clock/boost speed: 3.6/4.2GHz
- Cache Size: L1: 128 KB, L2: 1 MB, L3: 8 MB
- 32Gb Memory
- GPU not used