Austin Anderson
Hw 3: Analysis

For this assignment I extracted features by initially using the feature extraction provided by sklearn's CountVectorizer and TfidfVectorizer. I then wrote a custom analyzer to extract features. I finally combined the best aspects of both the custom analyzer and built-in analyzer for my best results. Results are tabulated in the tables on page 2. Accuracy for training data analysis was reported for each variation of feature extraction. All the training data was used for the tests and accuracy was reported using sklearn's accuracy_score function. Kaggle tests were run on only select feature extraction methods as the daily submissions were limited to two.

Built-in Exaction

I initially used the baseline vectorizers. I looked at using words, characters, and n-gram characters in word-boundaries. Examined this for both the Count and Tfidf vectorizers. The results can be seen in the table on page 2. The word based analysis performed significantly better than the characters or character in word-boundary analysis (runs 1 to 3 to 4). From there, I looked at using the n-gram analysis and various data editing features. Generally, the number of n-grams improved accuracy at the cost of increased run-time (runs 7,8,9). Accuracy results were found to taper between 8 and 10 n-grams (runs 9,18). Most tests were conducted using 8 n-grams as that produced highly accurate results without a huge hit to computation time. From there, several data preprocessing features were examined specifically the use of stop words, the stripping of accents, and putting bounds on term document frequencies. The use of stop words significantly reduced computation time at the cost of accuracy (runs 9,10). Similarly, bounding word document frequency to reject words occurring bellow a lower limit reduced accuracy (runs 9, 13, 14, 15). It was observed that highly accurate countvectorizer runs put emphasis on words like morgana, olivia, and sherlock as they were reported in the top ten feature results. Looking at the data, these names were found to tightly correlate to spoilers in the training data, but refer to shows and characters not mentioned in the test data. While the lower limit of document frequency did reduce accuracy, it did remove these overfit words at least from the top 10 results. Stripping accents was found to reduce accuracy (runs 1,5). The TfidfVectorizer was also explored as it was thought that the normalization it provides might help reduce the overfitting observed in the countvectorizer. It was found that while the Tfidf produced top ten feature results that did not typically include the overfit names like morgana or olivia, it always performed worse than the countvectorizer (runs throughout). This was true for both the training accuracy and the kaggle accuracy.

Custom Extraction

Custom extraction was accomplished by using the same two built-in vectorizers, but passing them a custom analyzer function that performed feature extraction. The custom features developed were whole sentence extraction, single word extraction, 2-gram, 3-gram, custom stop words, high frequency word stemmers, part of speech analysis, and IMDb genre extraction. Generally Tfidf was found again to perform worse than Count. The use of 3-grams was superior to 2-grams or single word as is evident from the results in the table (runs 2 to 4 to 6). The use of stop words was found to reduce accuracy somewhat for the countvectorizer (run 6 to 8). Word stemmers for the forms of “kill” and “death” were developed as they were found to appear frequently in the top-ten features. The use of these word stem extractors did not significantly impact performance using count one way or another (run 6 to 13). The parts of speech analysis used NTLK to estimate what part of speech each word was and passed this result as a feature. This feature extraction significantly increased computation time and did not positively impact performance (run 14 to 17). The IMDb genre extraction looked at the page associated with an entry to return its genres as reported by IMDb. This was done using data generated from IMDb ahead of time and accessed off-line at runtime using a dictionary structure. The use of genres and exclusion of whole sentences did result in a tie for the best run (run 21). It is noted that while th first run looks the highest, it is simply the trivial case of the training exactly matching sentences.

Combination Extraction

The built-in countvectorizer was found to outperform the custom analyzer in Kaggle scores (0.667 compared to 0.64). As it was believed the IMDb genres still had the richest information for classifying, they were combined with the built-in feature extraction by simply appending the genere strings to the end of the sentences. This resulted in a 0.999 accuracy for the training set and 0.696 accuracy on Kaggle.

Austin Anderson
Hw 3: Analysis

Built In Analyzer Results

Built In Analyzer

Run #\Feature	Vectorizer	Analyzer Type	Strip Accents	Ngram	Stop Words	Max Df	Min Df	Genre	Run Time	Train Accuracy	Kaggle Accuracy	Comments
1	Count	Word	Default	Default	Default	Default	Default	0	0.41	0.856	0.64	
2	TfiDf	Word	Default	Default	Default	Default	Default	0	0.42	0.807		
3	Count	char	Default	Default	Default	Default	Default	0	0.69	0.47		
4	Count	char_wb	Default	Default	Default	Default	Default	0	0.99	0.47		
5	Count	Word	ascii	Default	Default	Default	Default	0	0.44	0.838		
6	TfiDf	Word	ascii	Default	Default	Default	Default	0	0.45	0.807		
7	Count	Word	Default	1,2	Default	Default	Default	0	1.6	0.993		
8	Count	Word	Default	1,4	Default	Default	Default	0	6.02	0.996		
9	Count	Word	Default	1,8	Default	Default	Default	0	15.04	0.997	0.667	
10	Count	Word	Default	1,8	English	Default	Default	0	5.73	0.991		
11	TfiDf	Word	Default	1,8	English	Default	Default	0	5.84	0.973		
12	TfiDf	Word	Default	1,8	Default	Default	Default	0	14.61	0.967		
13	Count	Word	Default	1,8	Default	Default	0.0001	0	15.85	0.963		Less overfit
14	Count	Word	Default	1,8	Default	Default	0.00001	0	15.28	0.997		Overfit
15	Count	Word	Default	1,8	Default	Default	0.00005	0	15.31	0.9966		Overfit
16	Count	Word	Default	1,8	Default	0.999	Default	0	14.09	0.997		Overfit
17	Count	Word	Default	1,8	Default	0.999	0.001	0	11.15	0.971		Overfit
18	Count	Word	Default	1,10	Default	Default	Default	0	19.06	0.997		
19	Count	Word	Default	1,8	Default	Default	Default	1	16.81	0.999	0.69648	Best

Custom Analyzer Results

Custom Analyzer

Run #\Feature	Vectorizer	Sentence	Word	2-Gram	3-Gram	Stop Words	Kill Stem	Death Stem	Part of Speech	IMDb Genre	Train Accuracy	Run Time	Kaggle Accuracy	Comment
1	Count	1	0	0	0	0	0	0	0	0	0.999	0.45		Just Exact Matching
2	Count	1	1	0	0	0	0	0	0	0	0.935	0.75		
3	TfIDf	1	1	0	0	0	0	0	0	0	0.826	0.77		
4	Count	1	1	1	0	0	0	0	0	0	0.997	2.1		
5	TfiDf	1	1	1	0	0	0	0	0	0	0.922	2.11		
6	Count	1	1	1	1	0	0	0	0	0	0.998	4.13		
7	TfiDf	1	1	1	1	0	0	0	0	0	0.962	4.27		
8	Count	1	1	1	1	1	0	0	0	0	0.991	0.72		
9	TfiDf	1	1	1	1	1	0	0	0	0	0.972	0.7		
10	Count	1	1	1	1	0	1	0	0	0	0.998	4.211		
11	Count	1	1	1	1	0	0	1	0	0	0.998	4.08		
12	Count	1	1	1	1	1	1	1	0	0	0.992	0.68		
13	Count	1	1	1	1	0	1	1	0	0	0.997	3.95		
14	Count	1	1	1	1	1	1	1	0	1	0.909	0.71		
15	TfiDf	1	1	1	1	1	1	1	0	1	0.866	0.72		
16	TfiDf	1	1	1	1	0	0	0	0	1	0.945	4.24	0.64	
17	Count	1	1	1	1	1	1	1	1	1	0.75	264.73		
18	TfiDf	1	1	1	1	1	1	1	1	1	0.819	264.73		
19	TfiDf	1	1	1	1	0	0	0	1	1	0.92	269.33		
20	Count	0	1	1	1	1	1	1	0	1	0.859	0.63		
21	Count	0	1	1	1	0	0	0	0	1	0.998	4.12		
22	TfiDf	0	1	1	1	0	0	0	0	1	0.944	4.17		

Kaggle Team Name: Austin
Username: austinsteamboat
CU Identikey: amanders
(sorry about the difference between identikey and username)
Display Name: Spider Man
Best Kaggle Score: 0.69648