Genre Classification through Token Analysis of Movie Subtitle Files

Edmond C. Malone

Computer Science Department

Troy University, Alabama

Abstract

Machine learning is used in various industries to aid with making predictions and insights with the use of existing data that can help reduce manual effort. This paper will demonstrate an additional path to classify movie genres with only common words. The domain focus of this project is the movie industry. The data source is an SRT file and we will use that file to classify a genre of a movie. The application will test the the data against a few popular machine learning algorithms that are best to to solve multi-classification problems.

I. Introduction

Categorizing movies makes it easier for viewers to determine what they like and want to see. Genres contain four elements of a movie: story, setting, and plot which are equal to the genre of any film. Movie genres can be determined with various other elements besides the previously mentioned. Movie text and patterns can also be used to determine the genre of movies. By exploring commonly used words in each genre of movie we will be able to classify movies with only common word data in each genre and classify each movie with the correct genre.

II. Methods

Data Set 2.1

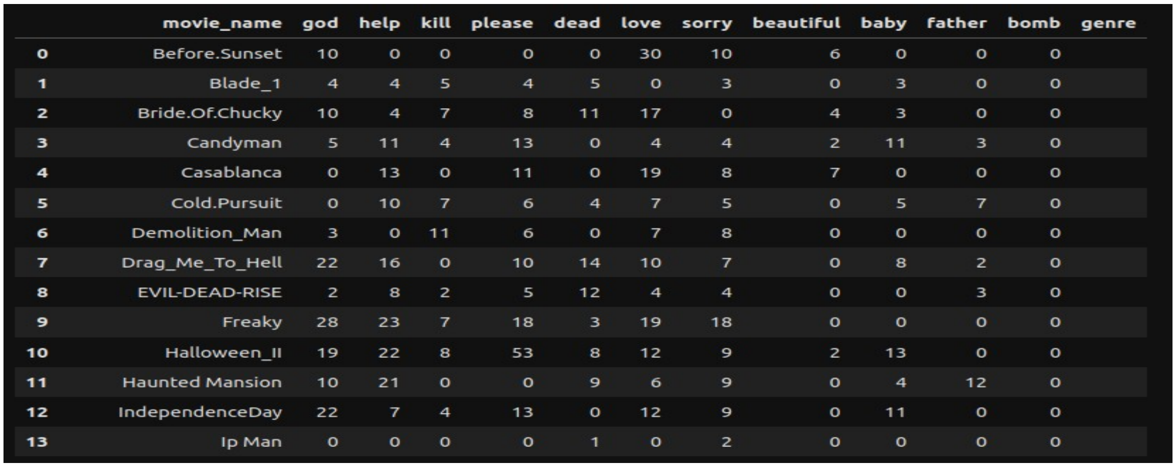
The data that was used for this project is SubRip Subtitle better known as SRT File. The SRT file is a plain text file that contains information that relates to how subtitles should appear on the screen. The file consists of the time sequential number for the subtitle and, the start and end time of the text to ensure the text matches the audio, the text that will be displayed on the screen. This data was gathered from subscene.com. The genres that are used in this paper are Action, Horror, and Romance. To utilize the SRT file to classify movie genres the data will need to be preprocessed. At this time I have acquired 20 SRT files for various genres of Action, Horror, and Romance movies.

Movie name extraction: This process is to extract the movie's actual name from the file path that was selected. The purpose of this step is to keep the data in the training set clean without any full file path but only the movie name. This step is not necessary but it helps to keep the movie name column header data clean.

Create a word bank of common words: To fully identify and create a word bank of commonly used words for each movie genre we will have to manually identify the common words in each genre and use that word as a column header for column names. Each table row will represent the individual movie and the number of frequent uses of the word in the given movie.

Perform upper to lowercase to text: To complete this task I used various of libraries such as pysrt which allowed me to parse the desired text section of an SRT file. The library's beautiful soup allowed me to separate the text from the HTML tags. I applied the lower method on the words to make it easier to verify the uppercase words and lowercase are identified as the same text and will not be separated.

I tokenized the corpus text and applied frequency distribution on the top 300 words words. This step allowed me to gain useful information and insight into common words that are frequently used in each genre and helped me to create a word bank and obtain the data for each cell for a movie. For the movies that do not have a value to that reflects the header name a zero will be placed in the cell field. This will be an accurate description of the values not present in the corpus.



	movie_name	god	help	kill	please	dead	love	sorry	beautiful	baby	father	bomb	genre
0	Before.Sunset	10	0	0	0	0	30	10	6	0	0	0	
1	Blade_1	4	4	5	4	5	0	3	0	3	0	0	
2	Bride.Of.Chucky	10	4	7	8	11	17	0	4	3	0	0	
3	Candyman	5	11	4	13	0	4	4	2	11	3	0	
4	Casablanca	0	13	0	11	0	19	8	7	0	0	0	
5	Cold.Pursuit	0	10	7	6	4	7	5	0	5	7	0	
6	Demolition_Man	3	0	11	6	0	7	8	0	0	0	0	
7	Drag_Me_To_Hell	22	16	0	10	14	10	7	0	8	2	0	
8	EVIL-DEAD-RISE	2	8	2	5	12	4	4	0	0	3	0	
9	Freaky	28	23	7	18	3	19	18	0	0	0	0	
10	Halloween_II	19	22	8	53	8	12	9	2	13	0	0	
11	Haunted Mansion	10	21	0	0	9	6	9	0	4	12	0	
12	IndependenceDay	22	7	4	13	0	12	9	0	11	0	0	
13	Ip Man	0	0	0	0	1	0	2	0	0	0	0	

Figure 2. Data table of various movies which list the sum of frequently used words in a movie.

Exploratory Data Analysis 2.2

Exploratory data analysis is an approach to analyze a dataset and to summarize the main points. Usually with visual methods. With this analysis, I used a bar graph to identify the frequently used words that are contained in various movies and to correlate the sum of each word for each movie to a genre.

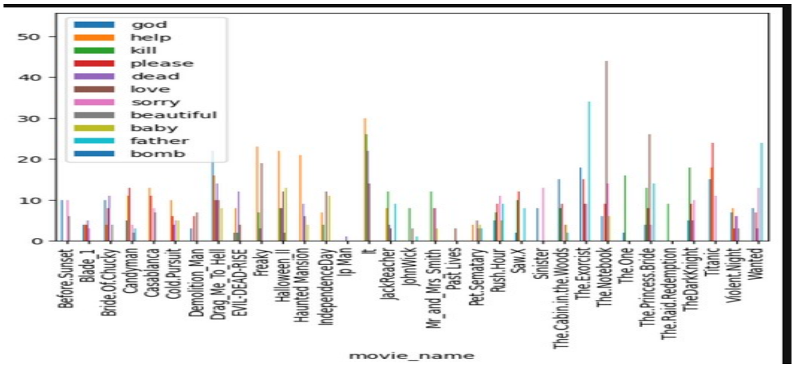
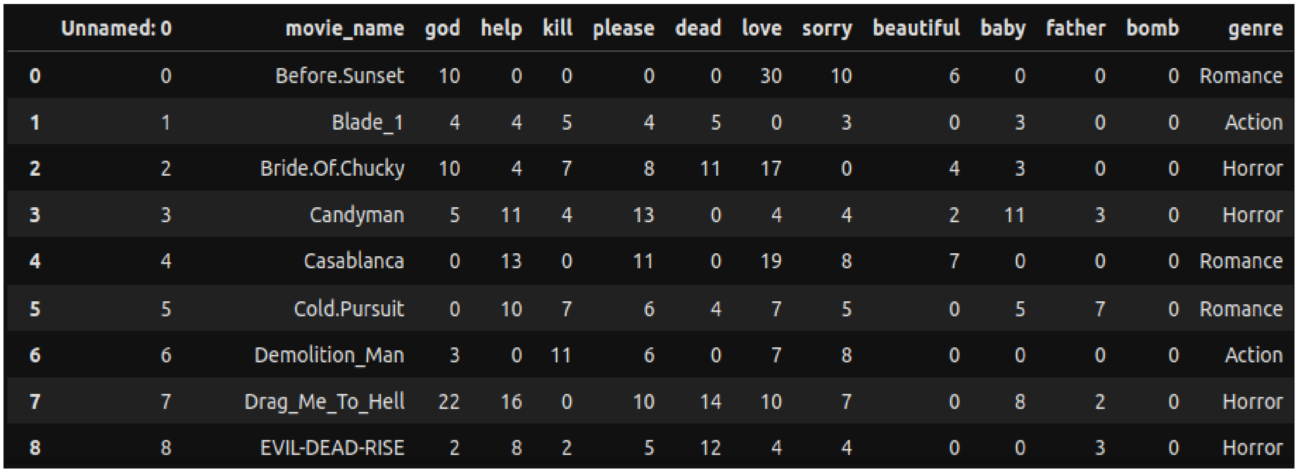


Figure 3. Bar graph of the sum of frequently used words in a movie

Data labeling 2.3

Because the SRT file does not know or contain the genre of the movie we will have to manually add the target values to the genre column. Before you can train the model to get the predictions I will have to label each movie row with its respective genre. I was able to gather each genre for each movie from the IMDB online database.



	Unnamed: 0	movie_name	god	help	kill	please	dead	love	sorry	beautiful	baby	father	bomb	genre
0	0	Before.Sunset	10	0	0	0	0	30	10	6	0	0	0	Romance
1	1	Blade_1	4	4	5	4	5	0	3	0	3	0	0	Action
2	2	Bride.Of.Chucky	10	4	7	8	11	17	0	4	3	0	0	Horror
3	3	Candyman	5	11	4	13	0	4	4	2	11	3	0	Horror
4	4	Casablanca	0	13	0	11	0	19	8	7	0	0	0	Romance
5	5	Cold.Pursuit	0	10	7	6	4	7	5	0	5	7	0	Romance
6	6	Demolition_Man	3	0	11	6	0	7	8	0	0	0	0	Action
7	7	Drag_Me_To_Hell	22	16	0	10	14	10	7	0	8	2	0	Horror
8	8	EVIL-DEAD-RISE	2	8	2	5	12	4	4	0	0	3	0	Horror

Figure 4. Data Table of the completed genre column

We import the training data and split the data into 80% training and 20% test data. Doing this would give you better results. From my analysis, I noticed if the data set is plentiful and the training set is greater than 50 % you will get better results than not having sufficient data and the training set is less than the test set. We will be using two classification machine learning algorithms. Decision tree and K-Nearest Neighbor

Decision Tree 2.4

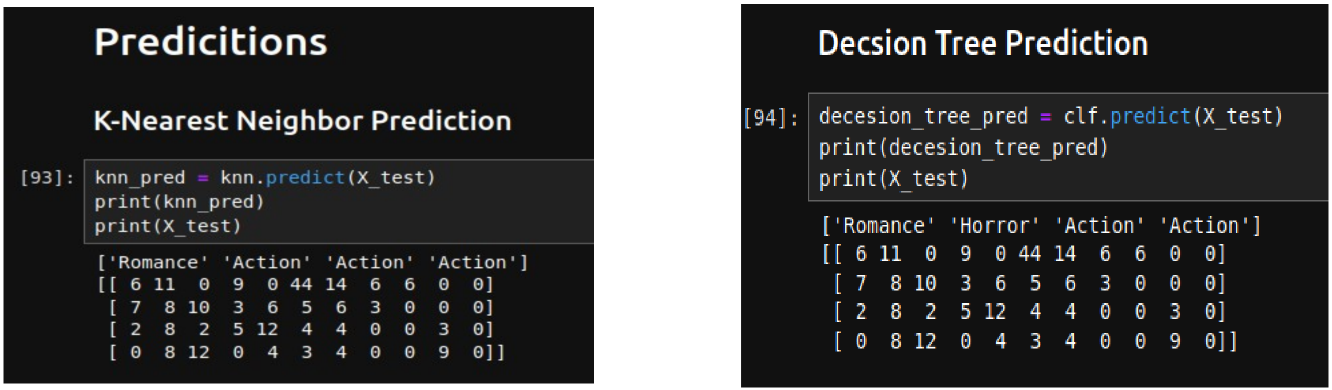
The Decision Tree is one of the useful supervised learning algorithms that is used for both classification and regression. It builds a reverse tree structure where each node is a feature and the leaf nodes are the results of the algorithm. The tree is created by iterating splitting the training data into subsets that are based on the values of their attributes until its criteria is met.

K nearest neighbor 2.5

The K-nearest neighbor is another classification algorithm in the machine learning arsenal. KNN predicts the label of a data point by calculating the distance between the input data point and all of the training examples, by using a chosen distance method.

Results 2.6

We used the same training data and test data for both algorithms. The results of both algorithms were different depending on the size of the training data and test data. The K-nearest neighbor algorithm proved to outperform the decision tree algorithm on various predictions.



Predicitions

K-Nearest Neighbor Prediction

```
[93]: knn_pred = knn.predict(X_test)
      print(knn_pred)
      print(X_test)

      ['Romance' 'Action' 'Action' 'Action']
      [[ 6 11  0  9  0 44 14  6  6  0  0]
       [ 7  8 10  3  6  5  6  3  0  0  0]
       [ 2  8  2  5 12  4  4  0  0  3  0]
       [ 0  8 12  0  4  3  4  0  0  9  0]]
```

Decsion Tree Prediction

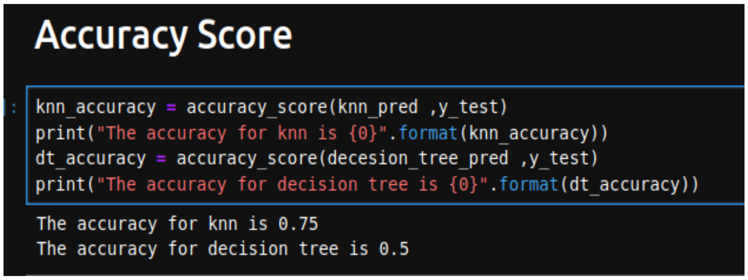
```
[94]: decesion_tree_pred = clf.predict(X_test)
      print(decesion_tree_pred)
      print(X_test)

      ['Romance' 'Horror' 'Action' 'Action']
      [[ 6 11  0  9  0 44 14  6  6  0  0]
       [ 7  8 10  3  6  5  6  3  0  0  0]
       [ 2  8  2  5 12  4  4  0  0  3  0]
       [ 0  8 12  0  4  3  4  0  0  9  0]]
```

Figure 5. Predictions of KNN and Decision Tree.

Accuracy 2.7

To compute the accuracy of the predicted labels the test set must exactly match the corresponding set of labels.



Accuracy Score

```
knn_accuracy = accuracy_score(knn_pred ,y_test)
print("The accuracy for knn is {0}".format(knn_accuracy))
dt_accuracy = accuracy_score(decesion_tree_pred ,y_test)
print("The accuracy for decision tree is {0}".format(dt_accuracy))
```

```
The accuracy for knn is 0.75
The accuracy for decision tree is 0.5
```

Figure 6. The output of the accuracy score

III. Future Work

There are some improvements we can add to increase the accuracy of the prediction. One of the approaches is to obtain more SRT files to create a training set to produce better results. The second approach can complement the previous approach by analyzing to determine the more common words that span the same genre.

IV . Conclusion

Our proposed method is to use common words from various genres to categorize movie genres. This research has proven to be successful. With providing decent accuracy scores, the analysis has proven this method can potentially be helpful in categorizing movie genres.

V . References

[1] https://subscene.com/

[2] https://www.nltk.org/

[3] https://scikit-learn.org/

[4] https://pypi.org/project/beautifulsoup4/

[5] https://www.geeksforgeeks.org/decision-tree/

[6] https://blog.hubspot.com/marketing/srt-file

[7] https://monkeylearn.com/unstructured-data/

[8] https://milnepublishing.geneseo.edu/exploring-movie-construction-and-production/chapter/2-what-is-genre-and-how-is-it-determined/

[9] https://machinelearningmastery.com/types-of-classification-in-machine-learning/

[10] https://www.ibm.com/topics/machine-learning

[11] https://www.educative.io/answers/what-is-the-accuracyscore-function-in-sklearn

[12] https://www.kdnuggets.com/2022/07/knearest-neighbors-scikitlearn.html

[13] https://www.imdb.com/