Automatic Genre classification of Movies based on movie plot

Siddharth Modala Kartik Shetty

Department of Computer Science, Rutgers University

Project for CS 530, Under professor Casimier Kulikowski

1. **Introduction/Abstract**

In this project we attempt to build an automatic movie genre classifier based on textual analysis of the plot of a movie. Currently, tagging of movies into genres is done manually and requires the person to watch the entire movie which is a time taking process. Internet movie database (IMDB) uses user reviews from trusted sources to tag each movie to its genre; this is a time taking process. Using AI techniques such as one v/s all Naive Bayesian and KNN we wish to automate this task and help speed up the process.  We are currently classifying [Adventure, Horror, Thriller, Crime and Documentary] and sub-genres [War and Sci-Fi]. In our approach we are trying to build a feature set consisting of the most frequently occurring words across all genres and use them as parameters whose values will help us model our classifier.  The Bayesian classifier that we built helped us identify the main genre and sub-genre but failed in multi label classification. Multi label classification was done with KNN classifier but the results were only satisfactory. We found that the feature set was not able to clearly distinguish the genres and we needed some more information to neatly segregate and group the genres. Use of more mature feature set and more robust classification methods would likely improve our results.

1. **Problem Description**

Movie genres help one in deciding if he/she wants to watch a particular movie. It tells the person what kind of movie it is. Tagging of genres to movie is done manually. To manually tag a movie, one must know what the movie is about. This can be done in a couple of ways. Based on the trailer of the movie, one can guess what a movie is about and tag the appropriate genre to it. This method may not always give us the right answer as trailers tend to provide incomplete information. Another way to classify the movies is by analyzing the scenes of the movie. This can be done by actually watching the entire movie or by reading the plot of the movie to understand what the movie is about. We attempt to automate this process of reading the movie plot and classifying it into particular genres.

Movies belonging to one genre have a similar theme. We look to exploit this property. Movies of the same genre tend to have similar words in their plots. We use the frequency of these words to determine if a movie belongs to a particular genre or not. The number of words a movie plot has in common with a genre is directly proportional to the probability that the movie belongs to the genre.

This approach requires natural language processing and some classification method to classify a movie. We need to identify each word and group similar words together. Based on the frequency of occurrence of these words we classify them.

1. **Prior work and Motivation**

In our research we found few projects which attempts to classify movies based on its audio video content and user reviews **[1][2][3]**. Classifying movie based on its video is a tedious task and requires image processing techniques that consume more time. Some of these methods classify a movie into only one genre or just specify all the genres the movie might belong to.  We also found that none tried to classify the movies into sub-genre or classify it based on the subject to which they belong to.

Several websites provide users the facilities to search and watch movies online. In order to improve the user experience proper categorization of movies is need. Hence, the task of automatic classification of movies based on its content is an important one as it would help segregate the movies based on the user preference. We attempt to identify the main genre of a movie and also try to classify movies into sub genres. We found one v/s all Naive Bayesian classifier as the most suitable one as it is a powerful and flexible in representing complex probability distributions of multiple random variables. The high flexibility allows us to use a large feature set with varying values and get good results.

1. **Data Collection and processing**

**Collection:**

We extracted the required information from IMDB’s Web site. During the initial phase of the project we thought of using the API provided by IMDB to download the required content. We were able to extract the movie names, genre information and movie URL’s of the top 500 movies belonging to each genre from IMDB.  IMDB provided a huge text file of approximately 300 Mb which contained plot data of all the movies that it had in the data base. The format that they used in the writing the plot text file and size of the single file made it difficult to parse and extract the information of the movies that we had selected. We then thought of dividing the file into chunks of smaller size and search in it but that method also did not work. Finally we decided to implement a web crawler that would use the URL that we had downloaded earlier to extract the information from the IMDB web pages and save it into csv (comma separated file format) which can be readily consumed.

Using the web crawler we collected movie names, genre information and plot data of 2500 movies (500 from each genre under consideration). On similar lines we also gathered information of 400 movies which exclusively contained war and sci-fi as their main subject.

**Processing:**

The data collected was first processed to create the bag of words (referred as BOW from now on). To create the BOW we consider all the data from our movie set. We use the movie plot to create tokens. Each individual word forms a token. While creating tokens we ignore digits (numbers) as they do not contribute to the genre of the movie. On tokenizing we found 340680 different tokens. There are many other words that do not contribute in deciding which genre the movie belongs to. Some of these are called stop words. Stop words are words which are present in almost all plots and do not contribute to the genre of the plot. Examples are articles, conjunctions, etc. We remove stop words by using the NLTK stop word list. After removing the stop words we were left with 186449 words.

We now had a set of most useful words.  But using the above words as it is in our feature set would make the classification task impossible because of the sheer size of the feature set. We needed a way to reduce its size. We followed the process of stemming and lemmatization which basically tries to reduce similar words to a common stem which represents all the words that can be formed from them. Stemming would not lead to any loss of information as it just helps in combining similar words into a single token. We use the NLTK stemmer on the remaining list to get words of different forms in the same base. This greatly helped us in reducing the size of the set we now had 15618 words. In order to build our feature set we needed only unique words and their frequency of occurrence. Using NLTK we found the frequency distribution of the words. For Bayesian approach we are considered words with frequency above 8 to form our BOW (we had 3461 stems in BOW). For KNN we used two feature sets one with word frequencies more than 8 and other more than 15(size of BOW was 2184).

We also find the in Term frequency- Inverse Document frequency. This is given by the formula:

**tf-idf(w,x) = x \* log (m/t\_w)**

Where, x is the frequency of the word w; m is the total number of training samples; and t\_w is the number of samples in which the word occurs.

These values of frequency are used for our calculations. We test the samples only across the words present in our BOW.

The following flow chart explains the steps followed for the classification.

Movie Name

Genre

Movie Plot

Text Processing Using NLTK.

Tokenizing, Stemming, Frequency Distribution

Html Web Page

Web Crawler

Movie Information

(csv format)

(name, genre, plot)

Binary labels

Resultlts

Resultlts

Knn Classifier

One vs. All Bayesian

**Figure - 1**

1. **Implementation**

We used NLTK **[4]** and python for this project. NLTK has many in-built functions which help in natural language processing. These functions are used for data processing. The classifiers were implemented using python.

We have used two methods to classify our data, namely, one v/s all Bayesian Classifier and multi-label KNN classifier.

* 1. **One v/s all Bayesian classifier**

Naïve Bayesian is a binary classifier. It classifies a data set into one of the two classes. It uses probability to perform this classification. For each word in the plot of the testing sample, the classifier creates the probability of the word belonging to that genre or not. Shown in the table below are the 10 most important features for the crime genre.

|  |  |  |  |
| --- | --- | --- | --- |
| Stem | Frequency | Label | Contribution |
| boss | 2.7080 | Crime : f | 16.8 : 1.0 |
| minut | 16.7636 | Crime : f | 9.8 : 1.0 |
| suicid | 12.6404 | Crime : f | 9.8: 1.0 |
| hannib | 26.2550 | Crime : f | 8.0 : 1.0 |
| commit | 11.5681 | Crime : f | 8.0 : 1.0 |
| low | 4.0253 | Crime : f | 8.0 : 1.0 |
| manner | 21.6593 | Crime : f | 8.0 : 1.0 |
| week | 3.6888 | Crime : f | 8.0 : 1.0 |
| narcot | 4.4308 | Crime : f | 8.0 : 1.0 |
| drug | 5.5451 | Crime : f | 8.0 : 1.0 |

**Figure- 2**

For multiclass classification we use the variation called One v/s All. We group all movies of a particular genre together as the positive samples and the rest as negative sample and then train the classifier for this genre. A classifier for each genre is created this way. We can do this because assigning a movie to a particular genre is a binary decision. Given a test set, we classify the movie using the classifier of each genre. The movie belongs to the genres that return a true value for the classification. To decide the main genre of the movie, we calculate probability of the movie belonging to a particular genre and the genre with the highest probability is the main genre of the movie.

* 1. **Multi-label KNN**

We used KNN for multi label classification. The Bayesian approach we used earlier was good at single label classification. We choose KNN because of the way our feature set was made. If we use the values of the feature set to represent a point in the n dimensional hyperspace then the points that belong to movies of same genre should be nearer to each other as they tend to have same type of words. Based on this assumption we implemented KNN classification. We encoded the class labels as a binary vector as shown below

if genre set contains = [“Thriller”, “Horror”, “Documentary”, ”Crime”, ”Adventure”] then the class label of an Adventure, Thriller movie will be encoded as [1,0,0,0,1].

In KNN we used two approaches.

**Uniform distance approach:** In this approach we assign uniform weights to each of the neighbor and then use voting to decide the class label which would be given to the test set.

**Euclidean Distance:** In this approach we assigned weights which are inversely proportional to the Euclidean distance between the test data and its neighbors. This method will give more weight to the point closer to the test data than the ones which are farther from it. This method was more successful at classifying the data. The results of this classification are shown in Results section.

* 1. **Sub-genre classification**

We created a separate training and testing test set to classify an already classified movie into one of the two sub-genres, war or fantasy. Below is the result of the training which shows the 10 most important words that contribute to the assignment.

|  |  |  |  |
| --- | --- | --- | --- |
| Stem | Frequency | Label | Contribution |
| offic | 1.0986 | war : fantas | 14.8 : 1.0 |
| attack | 1.9459 | war : fantas | 13.1 : 1.0 |
| militari | 1.6094 | war : fantas | 13.1 : 1.0 |
| american | 1.0986 | war : fantas | 13.0 : 1.0 |
| us | 2.1972 | war : fantas | 10.4 : 1.0 |
| german | 1.3862 | war : fantas | 10.3 : 1.0 |
| Colonel | 1.9459 | war : fantas | 8.6 : 1.0 |
| camp = | 1.6094 | war : fantas | 8.5 : 1.0 |
| magic = | 1.6094 | fantas : war | 8.2 : 1.0 |
| dark = | 1.6094 | fantas : war | 7.6 : 1.0 |

**Figure-3**

There was a lack of data available for sub genres and so we trained only for two genres. But one v/s all Bayesian classification can also be applied to classify the movie among multiple sub genres. But this classification is only multi-class.

1. **Results and Analysis**

**6.1 Bayesian classifier**

For one v/s all Bayesian classifier we performed two tests. One when the bag of words was created without using Term-Frequency- Inverse Document Frequency ratio (TF\_IDF) and one using it. The results for the various genres are:

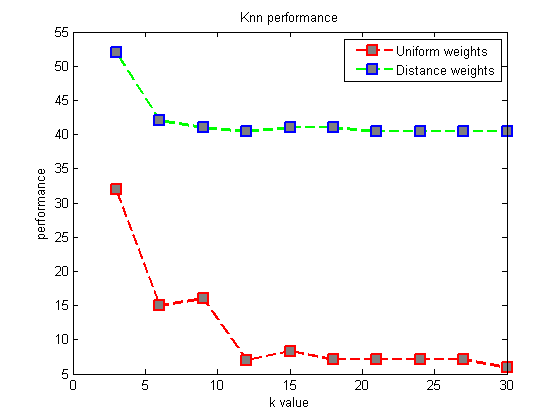
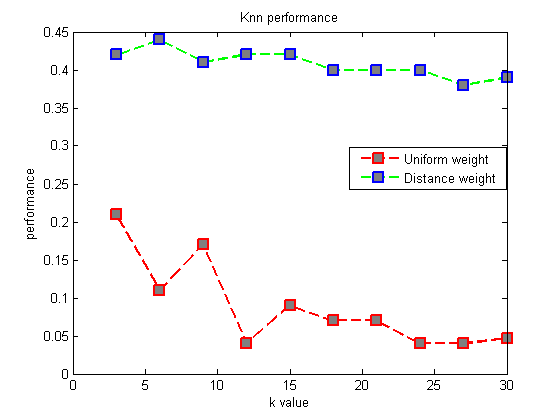
|  |  |  |
| --- | --- | --- |
| Genre | Without tf-idf | With tf-idf |
| Horror | 66% | 69% |
| Documentary | 81% | 85% |
| Adventure | 67% | 72% |
| Thriller | 58% | 64% |
| Crime | 56% | 58% |

**Figure-4**

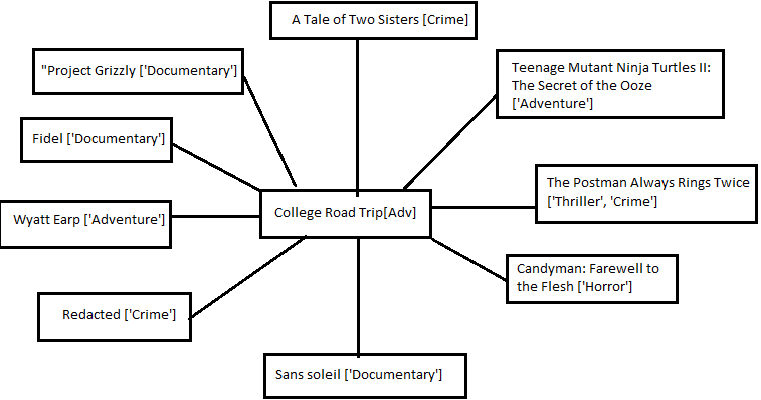
As noticed the addition of tf-idf has not produced a great variation in the results. Genres which are more distinct have produced better accuracy percentage (ex. Documentary) and the genres which tend to have an overlap have produced less (ex. Crime). Also note that the above result is classification of the movie to individual genres. When each movie is tested for all the genres it belongs to, the accuracy of getting all genres right is around **25-30%**. This is not good for practical applications.

One of the reasons for this is the lack of data. The classifier performs better when determining if a movie does not belong to a particular genre as compared to determining if it belongs. More data will improve the classifier as more words belonging to a particular genre will be present in the feature set.

**6.2 Multi-label KNN**

The results for KNN were not as expected. We thought the feature set will be very good at representing the genres in n dimensional hyperspace. But that was not the case. The following graphs show the accuracy of KNN with different values of k.

To find the optimal value of k for which the classifier had the maximum accuracy we ran tests from k value ranging from 3 to 30. As you can see in the above graphs when we use uniform weight the accuracy is not good and it decreases as the value of k increases. When we use Euclidean distance as the weight then we got satisfactory results. The graph on top is with a smaller feature set with words of frequency > 15 while the one in the bottom is with words of frequency > 8. We found that the maximum accuracy that we got was **52%** in correctly labeling all possible labels of the movie. Though this method was better than the Bayesian approach in multi label classification its results were not up to mark. In order to understand the reason behind it we analyzed the neighbors of a sample test set and found that the feature set poorly represented the genre in n-dimensional space.



**Figure-6**

In the above figure we can see that the test data “College Road Trip” which is an adventure movie is surrounded by all sorts of genres. In fact only 2/9 movies in its surrounding where actually Adventure movies.

**6.3 Sub-genre classification**

We used only two sub genres to classify data into sub genres. The lack of available data is the reason why we could not perform the classification across multiple sub-genres. The results were good. The correct movie between war and fantasy was predicted **86%** of the time.

Since this classification is only multi-class and not multi label, one v/s all Bayesian classification works well to classify sub genres. It is not multi label, because, given a genre, it should belong to only one sub-genre. The result obtained was satisfactory. This was due to the fact that sub genres are more separate as compared to genres, and so the classification gave better results.

1. **Conclusion**

In this project we attempted to classify movies into main genres and sub genres. We were fairly successful in doing it using Naïve Bayesian approach. KNN approach performed reasonably well in multi label classification. But the overall results were satisfactory. We need to dig into the text to find new relationship among the words that can represent the genre better. Use of other classification methods like SVM and Neural Networks might also improve the results.

1. **Future Work**

* Improve the feature set by doing semantic analysis of text. Using bi-grams and tri-grams in the feature set.
* Implement other classification methods like SVM, Neural Networks
* Synopsis provided basic abstract of the movie, but if we use the movie sub titles or movie script for analysis then we have more information about the content and it would help classify better.
* Possible impact beyond the course would be, if we are able to quantifying the content of the movie like 30% action and 40% adventure then we can use this information to recommending movies to users.

1. **References**

**[1]** Movie Classification Using Visual Effect Features, Hui-Yu Huang**,** Weir-Sheng, Wen-Hsing Hsu

**[2]** Issues in Decision tree classification of film genres using plot feature, Joshua Tanenbaum

**[3]** Classifying Movie Scripts by Genre with a MEMM Using NLP-Based Features, Alex Blackstock, Matt Spitz

**[4]** scikit-learn toolkit (<http://scikit.org/stable/modules/multiclass>)

**[5]** Natural language toolkit. (<http://www.nltk.org>).