Using movie scripts and text mining to predict movie genres

Rasmus Johns

January 10, 2018

Abstract

In this paper, the effectiveness of using text mining to predict genres for movie scripts is described. The model which proved most successful was a combination of a multi-layer perceptron and a decision tree; this combination scored 11.1% accuracy. Analyzing the result, the model was found to have potential for significant improvements.

1 Introduction

When selecting a movie to watch for the weekend, most often the genres of the movie will play a crucial part. The genres indicate what kind of experience the viewer can expect; some movies provide comfort, joy, and laughter, whereas others instill a feeling of deep fear and terror. This wide spectrum of movies can discretely be separated into what we know as genres.

To humans, genres can seem intuitive. Even if it is difficult to pinpoint what determines a movie's genres, we often have an innate understanding of what kind of experience we are having while watching the movie. For instance, when asked if *Indiana Jones* is an adventure movie, the answer is likely Yes. Yet, many will struggle to explain why they believe the movie should be classified as such. Indeed, most people would not autonomously quote the the International Movie Database (IMDb), which states that an adventure movie is a movie that *"Should contain numerous consecutive and inter-related scenes of characters participating in hazardous or exciting experiences for a specific goal. Not to be confused with Action, and should only sometimes be supplied with it"* [1].

In this paper, the option of using machine learning to determine movie genres is explored. The work is largely based on two previous works: first, a recent master's thesis attempting to classify movie genres based on subtitle data; second, an older work classifying texts into categories such as news or fiction.

2 Theory

This section presents theory necessary to understand the project's methodology.

2.1 Tf-idf

The individual characters constructing a word in textual data does not contain much inherent information about the word's meaning. Therefore, a machine learning model has to utilize a smart algorithm to, in some sense, understand textual data. Term Frequency - Inverse Document Frequency (tf-idf) is one such method. Tf-idf makes textual data more understandable by not focusing not on the characters; instead, tf-idf uses the frequencies of words gain information.

Tf-idf consists of two parts: term frequency and inverse document frequency, which can expressed as

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{1}$$

where t is a term and d is a document in the corpus D. The term frequency and inverse document frequency can be calculated as

$$td(t, d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{max\{f_{t',d} : t' \in d\}} \tag{2}$$

$$idf(t, D) = log \frac{N}{|\{d \in D : t \in d\}|} \tag{3}$$

where f is a frequency and N is the number of documents in the corpus. [2]

2.2 Stop words

In textual data, some words carry more meaning than others. From the sentence: "*When I walked in the park I saw a terrorist*" the crucial information lies in the words $\{walked, park, saw, terrorist\}$ – all other words could be considered *stop words*. Stop words are words that, when viewed from a general perspective, contain no information. [3]

Today, lists with stop words are easily accessible.

2.3 Models

When considering machine learning as a solution to a problem, a crucial decision is the model selection. Depending on the problem at hand and the data, it is likely that some models will produce superior results. In this section, all the presented models will have the trait of being multi-label, meaning they are capable of predicting multiple labels for each instance of test data.

2.3.1 Multi-layer perceptron

A multi-layer perceptron (MLP) model is an acyclic layered graph consisting of nodes and weights. Input is fed through the graph through the first layer, yielding an output from the final layer. In a fully connected network, every node is connected to other nodes in adjacent layers.

When training an MLP, the training is traditionally done by backward propagation of errors (backpropagation). In short, backpropagation is gradient descent; it calculates the gradient of the MLP's error function with respect to its weights. [4]

2.3.2 Decision trees

Decision trees are often referred to as white-box models – the way they do predictions is understandable. A decision tree works by modeling a set of conditions from the training data, where each condition can be asked as a question to determine the label of test data. [5] An example can be seen in Figure 1.

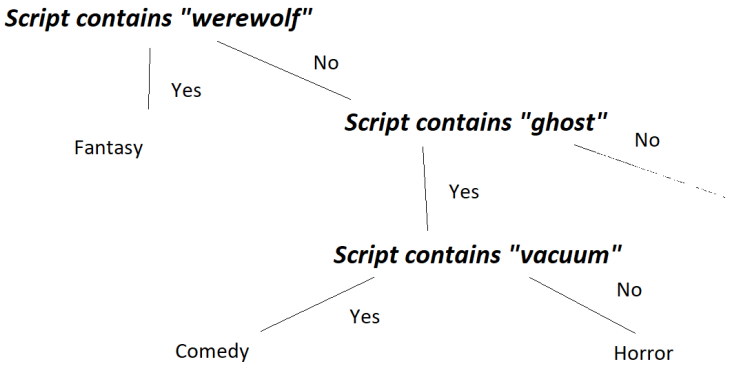

Figure 1: A decision tree predicting a single label for a script.

When training a decision tree, several parameters can be tuned. For instance, the tree may be restricted to a specific depth or width. Another key decision is selecting a splitting criterion. The splitting criterion is used during the training phase to select the set of conditions to be used in the model. Depending on which splitting criterion is being used, a different kind of outcome can be expected. [5]

2.3.3 k-nearest neighbor

Unlike most models, k-nearest neighbor does not have a training phase. Instead, k-nearest neighbor saves all training data in order to classify new data. The algorithm classifies new data by checking which k data points in the training data resembles the test data the most. These k data points in the training data classify the new data by voting – each data point will vote at its label.

An extension of the k-nearest algorithm is to use weighted k-nearest neighbor, meaning every data point's vote is weighted by its distance to the test data.

To use this algorithm to its full potential, one must determine a good magnitude of k, which traditionally is an odd number. A large k will result in underfitting, meaning the model will not fit the data, whereas a low k is likely to result in overfitting: a failure to generalize. [6]

2.4 Accuracy, precision and recall

The two most common measurements of precision are accuracy and recall.

Accuracy is measured as a model's ability to predict correctly. It can be expressed as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

where TP, TN, FP, and FN are explained in Table 1.

	Predicted true	Predicted false
Actual true	TP	FP
Actual false	FN	TN

Table 1: Explanation of TP, TN, FP, and FN.

Precision measures a model's ability to not label positive samples as negative. Precision is expressed as

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Recall measures a model's ability to find all true positives. In a case of classifying genres, recall is the ability to find all genres. Recall is expressed as

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

3 Method

This section provides the information required to reproduce the results of the paper.

3.1 Collecting raw data

The data used in this paper was scraped from the International Script Movie Database (IMSDb). IMSDb contains 1167 scripts of movies from various genres. The scripts ranged from modern blockbusters to old western movies. For this project, fourteen genres were picked from IMSDb to be scraped (see Table 2).

Genre	Number of scripts
Action	326
Adventure	191
Animation	40
Comedy	383
Drama	628
Family	51
Fantasy	128
Horror	152
Musical	26
Mystery	120
Romance	206
Sci-Fi	171
Thriller	400
Western	17
Total	1167

Table 2: Amount of scripts from each genre.

Once these scripts had been identified, the scraper navigated to each separate script and checked that it existed. If it did, potential metadata was stripped and the script was saved away; a sample from such a script can be seen in Appendix A.

3.2 Preprocessing scripts

The methodology behind the preprocessing was based on Lee's work *Text-based video genre classification using multiple feature categories and categorization methods*. Using subtitle data for movies, Lee showed that it was possible to classify genres using text mining. [7]

First, the scripts were broken down into tokens using the Python module Natural Language Toolkit (NLTK). The tokens were then lemmatized and filtered through NLTKs list of stopwords. Furthermore, in an attempt to purge the scripts from names of people, the StanfordNERTagger was used to classify each word and remove names.

3.3 Building model input data using tf-idf

With the collection of tokens, a tf-idf algorithm from the python module scikit-learn was utilized. The goal of the tf-idf algorithm was to produce output for a classifier. Therefore, the number of words within various document frequencies was evaluated (see Figure 2 and Figure 3).

From these figures, it was clear that many words, in this case refered to as tokens, had low document frequency; consequently, the tf-idf algorithm was set to ignore all tokens with a document frequency lower than 4%. This change sim-

plified output of the tf-idf drastically, seeing how most tokens had a document frequency lower than 4%.

There were primarily two reasons as to why such a large proportion of tokens were found in this interval. First, IMSDb has a tiny amount of scripts written in their original language; for instance, the script *Un-Singe-en-Hiver* in French existed in the database. The second, more general reason, is that scriptwriters have great imagination when writing dialogue. Words in dialogue often get twisted when characters have a runny nose, an exotic accent or a mouth full of food. Another reason which could have contributed to the statistics of tokens is if the web crawling failed to retrieve a script and saved something else as a script.

Then, to reduce the number of tokens carrying low amounts of information, the algorithm was also set to ignore tokens with a higher document frequency than 50%. This change could be seen as a filter for script specific stop words.

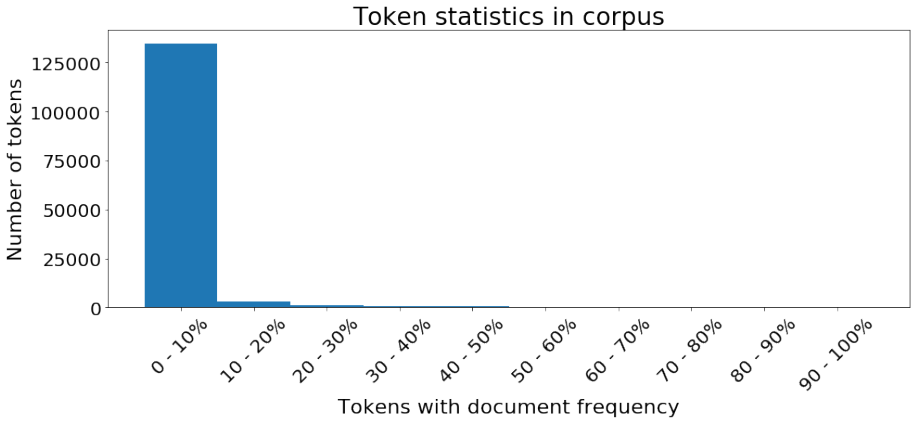

Figure 2: A complete picture of token statistics in the corpus.

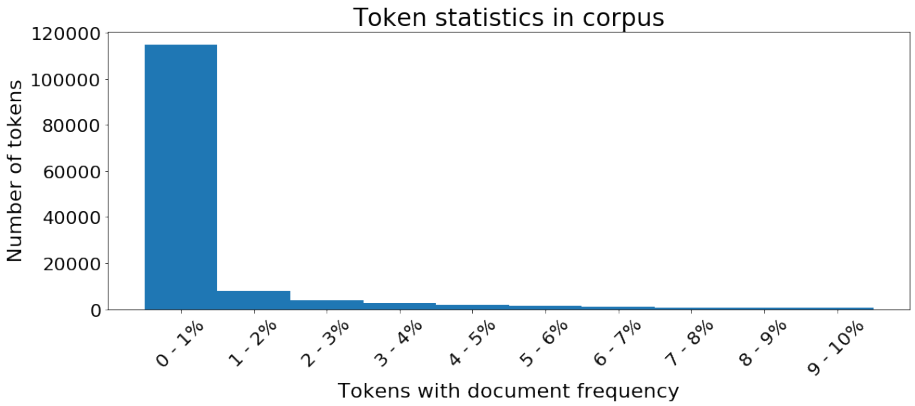

Figure 3: A zoomed in picture of token statistics in the corpus.

The tf-idf algorithm yielded a matrix where each row was a vector consisting of 11550 tf-idf scores, calculated according to Equation 1, for a specific script. The positions of these vectors were then shuffled. Finally, the matrix was split into a training set and a test set of equal size. The distribution of genres within the training and test data can be seen in Figure 4 and Figure 5.

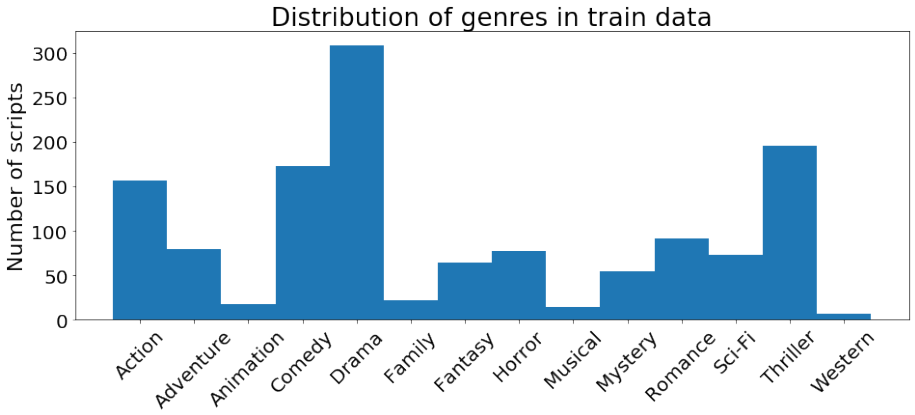

Figure 4: Distribution of genres in train data.

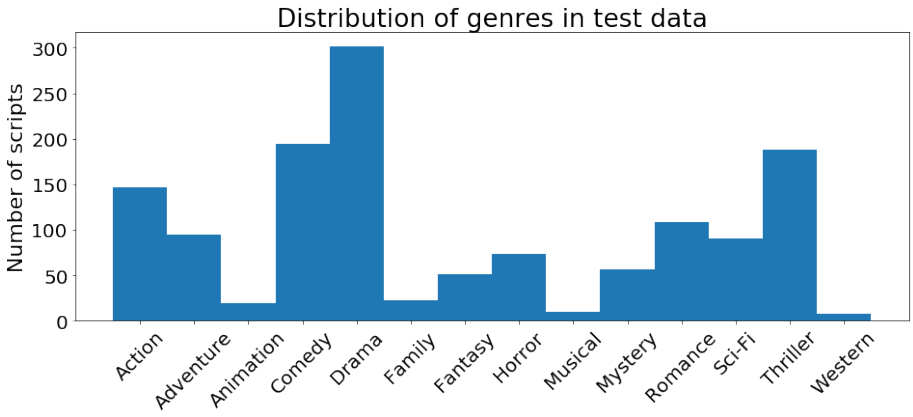

Figure 5: Distribution of genres in test data.

3.4 Building a model

Using scikit-learn's library of multi-label classification algorithms, three models were tested: MLP, decision tree, and k-nearest neightbour.

Due to the nature of the task, all models faced an issue: scripts always have at least one genre and the models used in the project had no way of integrating this knowledge. However, the decision tree model suffered far less from being incapable of choosing a genre. As the decision tree showed more promise in this regard, it was chosen as a second layer to the two other models: when the MLP or k-nearest neighbor algorithm could not select at least one genre, a decision tree attempted to do it instead.

Since scikit has solid models that work out of the box, no significant changes were made. After attempting several tweaks, the parameters in Table 3 were chosen. In addition, the architecture of the multi-layer perceptron was very inspired by the similar work, by *Kessler et al.*, called *Automatic Detection of Text Genre*, which classified genres of news articles [8].

Model	**Parameters deviating from scikit-standard**
Decision tree	None
MLP	$alpha = 10^{-5}$, $solver = lbfgs$
k-nearest neighbor	$k = 3$

Table 3: Models used in the project.

3.5 Accuracy, precision and recall

When training these models, accurate predictions were defined as replication of the golden standard. Getting only one genre wrong counted as a misclassification.

The precision and recall were calculated slightly different than explained in Equation 5 and Equation 6. As suggested by M. L. Zhang and Z. H. Zhou in *A Review on Multi-Label Learning Algorithms*, recall and precision should be used thoughtfully in multi-label problems. [9] Based on their recommendation, $Precision_{micro}$ and $Recall_{micro}$ were implemented. These could be expressed as

$$Precision_{micro} = \frac{\sum_{j=1}^{q} TP_j}{\sum_{j=1}^{q} TP_j \sum_{j=1}^{q} FP_j} \tag{7}$$

$$Recall_{micro} = \frac{\sum_{j=1}^{q} TP_j}{\sum_{j=1}^{q} TP_j \sum_{j=1}^{q} FN_j} \tag{8}$$

where q is the number of labels and j is the index of a specific label.

By using these measurements, another level of depth was added to the evaluation of the model. Instead of solely focusing on the end result, the model's ability to do decisions on a micro level was also accounted for.

4 Result

From the models' result in Table 4, it can be concluded that the MLP, which was combined with a decision tree for the few entries the MLP failed to give a genre, scored highest with 11.1% accuracy. Moreover, the MLP had the highest precision and recall scores.

Model	**Accuracy**	**Micro precision**	**Micro recall**
Decision tree	7.6%	41.5%	37.1%
MLP	11.1%	50.1%	40.0%
k-nearest neighbor	6.6%	43.7%	32.5%

Table 4: The result of models used in the project.

Since the MLP outclassed the other classifiers, the results henceforth will be products of this model.

4.1 Genre distribution

Comparing the distribution of genres in the MLP's classification with the test data, one can tell it is a good match (see Figure 5 and Figure 6). For a more

in-depth look at predictions, see Appendix B and Appendix C for sample predictions.

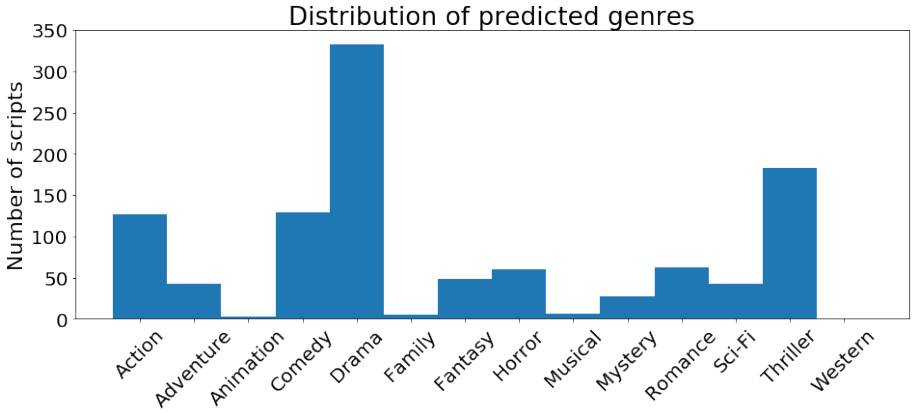

Figure 6: Distribution of genres in MLP predictions.

4.2 Accuracy

As mentioned, the MLP scored 11.1% accuracy. Looking more closely at the accuracy, it can be observed that the model performed better for scripts with fewer genres. The relation between the number of genres and the MLP's accuracy can be studied in Figure 7 and Figure 8.

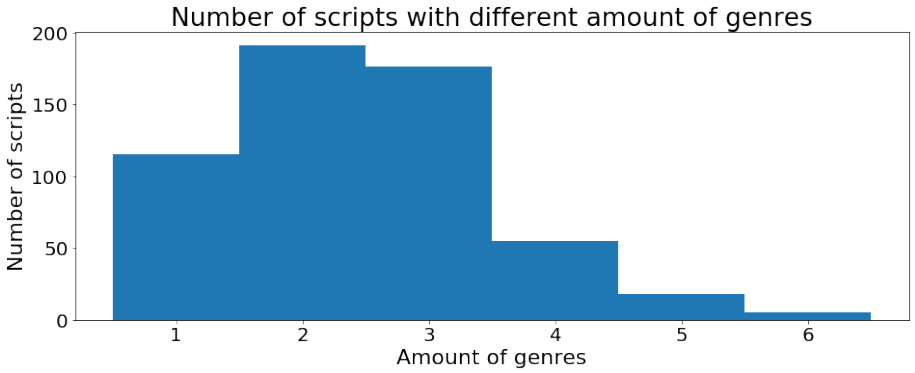

Figure 7: Number of scripts with different amount of genres.

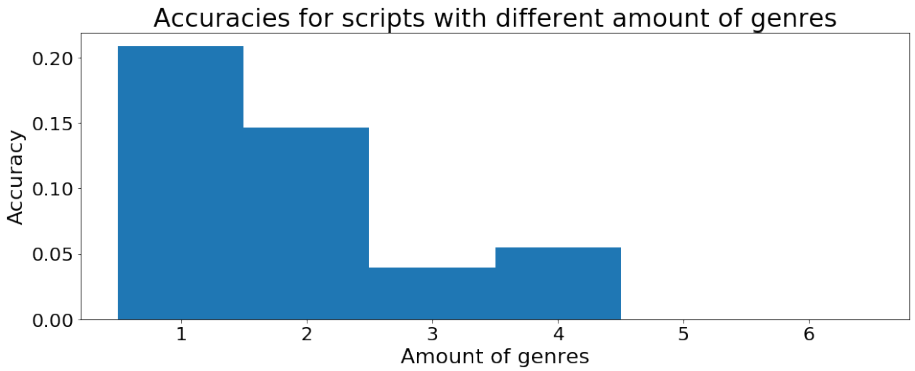

Figure 8: Prediction accuracies for scripts with different amount of genres.

As seen in Figure 8, the accuracy declined for scripts with multiple genres. However, if the model was allowed to mispredict one genre for scripts with more than one genre, such as predicting $\{Drama, Thriller\}$ instead of $\{Drama, Thriller, Action\}$, the accuracy rose to 35.2%. Furthermore, if the accuracy was calculated on a label based level, similar to the precision and recall, the accuracy rose to 82.7%.

5 Discussion

Looking at the model's result, the low final accuracy stands out. However, accuracy alone might not the best measurement of a multi-label model. After all, as seen in Section 4.2, the model had 35.2% when it was allowed to get one genre wrong. It could be argued that the model was learning good behaviors, but was not given enough data to work out all details. Looking at the micro precision and micro recall, it is clear that the model does good decisions; it just struggles with combining its knowledge of individual labels to multi-label classifications.

This theory is supported by the fact that the model was not fed much data; in fact, one thousand scripts to be shared between training and test data is far from sufficient for a problem of this complexity. Unfortunately, the low number of scripts at IMSDb was discovered too late into the project, as it was only when the web crawling finished it was noticed. The low amount of data also stands out when comparing this work to Lee's work, which had 100 times more data and achieved 90% accuracy (although subtitle data may also be superior due to its richness in metadata, such as the speed of subtitles)[7].

Furthermore, only when debugging the low accuracy was the questionable golden standard at IMSDb uncovered. Scripts uploaded to IMSDb are given genres by IMSDb themselves; as a result, IMSDb deviates from the phonetically related and more established site IMDb. For instance, the movie *Jurassic Park (1993)* is labeled as $\{Adventure, SciFi, Thriller\}$ at IMDb, but $\{Action,$

$Adventure, Horror, SciFi, Thriller\}$ at IMSdb. Overall, IMSDb are very liberal with the number of genres they give a script.

It is essential to see Appendix C to observe how the model incorrectly predicts some scripts. Most of these predictions are, although deviating slightly from the golden standard, really good. It could be argued that nine out of the fifteen scripts found in Appendix C are correct.

Another interesting field of study is the distribution of predicted genres. Comparing Figure 5 to Figure 6, primarily three genres stand out: *Animation*, *Musical* and *Western*. The reason why the classifier most likely struggles with animated scripts is that the genre is not necessarily linked to the content of the script; the movie titled *9*, which is an animated sci-fi drama, does not share a lot of script content with Aladdin. Similarly, scripts for musicals are though since they do not include the actual music. Another issue with musicals is their tendency to have ten minute long scenes which on paper are described as "*And then they notice, they can dance together!*". Finally, the model may struggle with western movies since there was not enough data in this category – looking at Table 2, it is clear that the entire data set only contains 17 western scripts.

Finally, it is important to understand this method of predicting genres has limitations – the scripts are not always final and even if they are, they lack information about the movie; for instance, scripts do not indicate if it is an animated movie, who the actors are, what kind of aspect-ratio is being used or if the movie is shot using mostly wide shots or close-ups.

6 Conclusion

The goal of the project, predicting genres for movies using scripts, is reachable. Although the project did achieve an accuracy lower than expected, analysis of the accuracy showed that the result is quite good. With more time and resources, it would have been interesting to do a study and see if the model is better than people in general at predicting genres. Low scale testing suggests that it is. Furthermore, it would have been smart to study the source of data, IMSDb, more before allocating time to crawl the site. This is one of the main lessons learned the project. Another lesson is to not be scared of initial bad results – it can turn out that they are better expected.

In general, the project has brought attention to the scope and magnitude of large-scale machine learning projects. Going from an idea to finding sources of information, retrieving and preprocessing data, building models, analyzing results to then realizing something in the preprocessing needs to be tweaked and redoing the whole thing, takes time and dedication.

References

[1] International Movie Database Genre Definitions. https://help.imdb.com/article/contribution/titles/genres/GZDRMS6R742JRGAG. Accessed: 2018-01-05.

[2] KyoJoong Oh, S. S. K., Chae-Gyun Lim & Choi, H.-J. Research trend analysis using word similarities and clusters. *International Journal of Multimedia and Ubiquitous Engineering* **8** (2013). URL http://www.sersc.org/journals/IJMUE/vol8_no1_2013/17.pdf.

[3] AntoineBlanchard. Research trend analysis using word similarities and clusters. *World Patent Information* **29**, 308–316 (2007). URL https://www.sciencedirect.com/science/journal/01722190.

[4] Hassoun, M. H. *Fundamentals of Artificial Neural Networks* (MIT Press, Cambridge, MA, USA, 1995), 1st edn.

[5] Quinlan, J. R. Induction of decision trees. *Mach. Learn.* **1**, 81–106 (1986). URL http://dx.doi.org/10.1023/A:1022643204877.

[6] Wang, L., Khan, L. & Thuraisingham, B. An effective evidence theory based k-nearest neighbor (knn) classification **1**, 797–801 (2008).

[7] Lee, C. *Text-based video genre classification using multiple feature categories and categorization method*. Master's thesis, Radboud University, Department of Communication and Information Sciences (2017). URL http://theses.ubn.ru.nl/bitstream/handle/123456789/5021/Chris%20van%20der%20Lee%20s4000528%20ReMA%20scriptie%202017.pdf?sequence=1.

[8] Kessler, B., Numberg, G. & Schütze, H. Automatic detection of text genre. In *Proceedings of the Eighth Conference on European Chapter of the Association for Computational Linguistics*, EACL '97, 32–38 (Association for Computational Linguistics, Stroudsburg, PA, USA, 1997). URL https://doi.org/10.3115/979617.979622.

[9] Zhang, M. L. & Zhou, Z. H. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* **26**, 1819–1837 (2014).

Appendix A Sample from the script: Alien (1976)

INTERIOR - MULTI-PURPOSE ROOM
The entire crew – STANDARD, ROBY, BROUSSARD, MELKONIS, HUNTER, and FAUST – are all seated around a table, with Standard at the head.
MELKONIS
If it's an S.O.S., we're morally obligated to investigate.
BROUSSARD
Right.
HUNTER
I don't know. Seems to me we came on this trip to make some credit, not to go off on some kind of side trip.
BROUSSARD
(excited)
Forget the credit; what we have here is a chance to be the first men to contact a nonhuman intelligence.
ROBY
If there is some kind of alien intelligence down on that planetoid, it'd be a serious mistake for us to blunder in unequipped.

Appendix B Sample of correctly predicted scripts

```
Alien Resurrection - ['Action', 'Horror', 'Sci-Fi', 'Thriller']
Man on the Moon - ['Comedy', 'Drama']
The Verdict - ['Drama']
Mud - ['Drama']
Life As A House - ['Comedy', 'Drama']
Life of David Gale, The - ['Drama', 'Thriller']
RKO 281 - ['Drama']
Lord of the Rings Return of the King - ['Action', 'Adventure', 'Fantasy']
Grapes of Wrath, The - ['Drama']
War Horse - ['Drama']
Last Station, The - ['Drama']
Commando - ['Action', 'Thriller']
Hellbound Hellraiser II - ['Horror', 'Thriller']
Tomorrow Never Dies - ['Action', 'Adventure', 'Thriller']
Indiana Jones and the Raiders of the Lost Ark - ['Action', 'Adventure']
Man Who Wasn't There, The - ['Comedy', 'Drama']
Matrix Reloaded, The - ['Action', 'Sci-Fi', 'Thriller']
```

Appendix C Sample of incorrectly predicted scripts

```
After.Life - ['Drama', 'Mystery', 'Thriller']
Hall Pass - ['Comedy']
Roommate, The - ['Action', 'Drama']
Un Singe en Hiver - ['Comedy']
Total Recall - ['Action', 'Thriller']
Chronicles of Narnia The Lion, the Witch and the Wardrobe - ['Sci-Fi']
War of the Worlds - ['Thriller']
Cruel Intentions - ['Thriller']
Dallas Buyers Club - ['Comedy', 'Drama']
Tremors - ['Action', 'Drama', 'Thriller']
S. Darko - ['Sci-Fi']
Avengers, The (2012) - ['Action', 'Drama']
Amityville Asylum, The - ['Drama']
The Abyss - ['Action', 'Drama']
Minority Report - ['Drama', 'Thriller']
```