**CSP 571- Data Preparation and Analysis**

Department of Computer Science

Illinois Institute of Technology

**Predicting Movie Genre**

Harshaa Bajaj, Vinod Rao

(A20378813), (A20369838)

**Instructor:** Adam McElhinney

## **Problem Statement**

Can we auto-classify a movie based on the information we have about it?

**Goal:**

Explore existing relationships between the listed genre of a movie and a set of attributes related to the movie to help make prediction of the genre for the upcoming movies.

**Approach:**

Use machine learning algorithms to understand the associations between the dependent and independent variables and make predictions. With focus on being able to make predictions for the genre of a movie, we would also like to compare multiple models and select the most efficient one.

**Related Research**

The below listed material includes theoretical and methodological contributions to similar topic,

* A study by Karl Persson showed that random forest yield consistently better performance than support vector machines for predicting user ratings.[1]

Although the study was not directly related to predicting genre, it gave a good insight into the methods of handling other variables

* A research paper by Andrew McGregor Olney focuses on Likability-Based (predictions based on user ratings), Content-Based (predictions based on plot keywords) and Synopsis-Based (predictions based on plot summary) based modelling.[2]
* Ka-Wing Ho in his project successfully implemented One-Vs-All approach with SVM, KNN, Parametric mixture model and Neural network.[3]

This paper helped to understand which models suit the current dataset and required output

**Data Description**

The table below shows the methods and attributes used from the IMDbPy module to collect the respective variable information about a movie

|  |  |  |
| --- | --- | --- |
| **Item** | **Method/ Attribute** | **Description** |
| Release Year | get('year') | For TV titles this is the broadcast year of the first episode. For movie titles this year is the year of general release |
| MPAA | get('mpaa') | The Motion Picture Association of America provides the IMDb with its ratings reasons. Example, G, PG, PG-13, R, etc., |
| Poster URL | get('full-size cover url') | URL to an official poster |
| Kind | get('kind') | Categories of titles. Example, Films,TV movies,TV series,Music videos, etc., |
| Plot Keywords | get(‘plot\_keywords’) | Word (or group of connected words) attached to a title to describe any notable object, concept, style or action |
| Storyline | get(‘plot\_outline’) | Detailed description of the entire plot |
| Titles | Item["title"] | Displayed titles according to their original country-of-origin language |
| IMDB Score | Item["rating"] | Weighted average of all the individual votes cast by IMDb registered users |
| Director(s) | Item["director"] | List of individuals that have played an role in bringing a film's creative and dramatic vision to the screen |
| Writer(s) | Item["writer"] | List of individuals who have written the original script |
| Cast | Item["cast"] | List of individuals who appear in footage |
| Run time | Item["runtime"] | Duration in minutes |
| Certification | Item["certification"] | Ratings certificates given to titles within each country. Example, Spain: T,7,X |
| IMDb URL | Item["url"] | Link to the movie on IMDb |
| Genre | Item["genre"] | IMDb classified genres. Example, Drama, Talk-Show, Romance, etc |

Where Item is the object returned by get\_movie() with IMDB ID (Identifier from IMDb)as input parameter.

**Data Preparation**

**Cleaning**

* The initial dataset contained 18,87,166 records with 8,840 unique combinatorial genres.
* Genre: 2,30,725 null genres were detected and removed
* IMDb ID: 55,111 null movie ids were identified and removed.
  + Using regular expressions identified if the ID values match the url id - 17 non matching records were detected and removed
  + 6,53,639 duplicated IDs were observed and thus removed
* Release Year: Checked if it is in range (1874, 2117), source- IMDb website
* IMDB Score: Checked for valid range through 0-10

**Transformation**

* ID, title, genre, director, cast, writer, plot summary, plot keywords were considered for further analysis
* Null value rows were dropped and a final subset consisting of 6,48,715 records was obtained
* All the columns were changed to lowercase and tokenized
* 153 stop words and punctuations were removed from the text words
* Lemmatization was performed on the remaining words
* 29 unique genres were identified and TF-IDF *( Term Frequency-Inverse Document Frequency )* scores were generated for each of them
* Formula used for TF-IDF score:
  + Term Frequency \* log(Document Frequency/Size of Corpus) \* (-1)

**Exploratory Modelling**

Five different modelling techniques were attempted. This section contains the working logic of these models towards predicting the genre

1. **Cosine Similarity**

The cosine similarity between two documents in the Vector Space is the comparison between them considering the magnitude of each word count (tf-idf) of each document and the angle between the documents.

Three resultant scores can be obtained:

* Similar scores
  + score vectors are in same direction
  + angle between them is near 0 deg
  + cos of angle is near 1 i.e, 100%
* Unrelated scores
  + score vectors are orthogonal
  + angle between them is near 90 deg
  + cos of angle is near 0 i.e, 0%
* Opposite scores
  + score vectors in opposite direction
  + angle between them is near 180 deg
  + cos of angle is near -1 i.e, -100%

1. **K-Nearest Neighbors**

To classify a document X, the k-Nearest Neighbor classifier algorithm ranks the documents neighbors among the training document vectors, and uses the class labels of the k most similar neighbors to predict the class of the new document. The classes of these neighbors are weighted using the similarity of each neighbor to X, where similarity is measured by Euclidean distance or the cosine value between two document vectors.

1. **Logistic Regression**

Input: tfidf vectors: xi = [xi,1 ,..., xi,j, ..., xi,d]T

The values yi ∈ {+1,−1} are class labels: (+1) or nonmembership (−1) of the vector in the category.

Conditional probability is given by:

p(y = +1|β, xi) = ψ(βT xi)

= ψ(Σj βjxi,j)

with logistic link function ψ(r) = exp(r)/1 + exp(r)

p(y = +1|xi) is the estimate of the probability that the ith document belongs to the category.

The decision to assign the category can be based on comparing the estimate to a threshold or by computing which decision gives optimal expected utility

1. **Naive Bayes**

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. There are three variations to it:

* + Gaussian naive Bayes
    - When values associated with each class follow Gaussian distribution.
  + Bernoulli naive Bayes
    - Features are independent booleans (binary variables) describing inputs.
  + Multinomial naive Bayes
    - With a multinomial event model, samples (feature vectors) represent the frequencies with which certain events have been generated or counting the number of times event was observed in a particular instance.

Multinomial naive Bayes works well for document classification as follows,

Let the set of classes be denoted by C. Let N be the size of vocabulary. Then MNB assigns a test document ti to the class that has the highest probability Pr(c|ti), which, using Bayes' rule, is given by:

Pr( c|ti ) = Pr( c ) Pr( tiIc ) Pr( ti ), c 𝞊 C

The class prior Pr(c) can be estimated by dividing the number of documents belonging to class c by the total number of documents.

Pr( tilc ) is the probability of obtaining a document like ti in class c and is calculated as:

Pr( tilc ) = (Σn fni )! ㅠn Pr (wn|c)f  fni !

where fni is the count of word n in our test document ti and Pr(w ni|c) the probability of word n given class c.

The latter probability is estimated from the training documents as:

est Pr (wni|c) = 1+ Fnc  (N +ΣN Fxc)

where Fxc is the count of word x in all the training documents belonging to class c

1. **One vs all SVM**

* SVM manipulates documents to represent them as points in a high dimensional space and then finds a hyper-plane that optimally separates the categories
* In One vs all, each classifier is trained to recognize one particular category versus all other categories.
* For each category the classifiers are trained on a labeled set where the documents from the corresponding category make up positive examples and rest of the documents become negative examples.
* In order to classify new data, each classifier in the ensemble is used to produce a confidence value of a point belonging to the corresponding category and the category with greatest confidence is selected

**Model Comparison**

Input: 29 TFIDF vectors representing 29 genres generated across the training set consisting of 6,48,715 unique movie records

Test Set: TFIDF scores generated from 8018 unique movie records

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Precision** | **Recall** | **F-1** | **Accuracy** |
| Cosine Similarity | 0.74 | 0.71 | 0.72 | 0.70 |
| KNN | 0.00 | 0.06 | 0.01 | 0.060 |
| Logistic Regression | 0.19 | 0.10 | 0.07 | 0.099 |
| MNB | 0.58 | 0.47 | 0.49 | 0.472 |
| SVM | 0.02 | 0.00 | 0.00 | 0.001 |

**Conclusion**

Cosine Similarity seems to performing better when compared to other models

**Future Work**

* Tune parameters before modelling for better performance
* Gather more textual data for robust modelling, example include user reviews.

**References**

1. <http://www.diva-portal.org/smash/get/diva2:821533/FULLTEXT01.pdf>
2. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3573840/>
3. <http://cs229.stanford.edu/proj2011/Ho-MoviesGenresClassificationBySynopsis.pdf>