Movie Genre Multi-label Classification from Plot Descriptions

HARVARD School of Engineering and Applied Sciences

Andrew Lund, Nicholas Morgan, Amay Umradia, Charles Webb
CS109B - Data Science II - Spring 2018 Final Project

IACS INSTITUTE FOR APPLIED COMPUTATIONAL SCIENCE AT HARVARD UNIVERSITY

Introduction

The overarching goal of this project is to **build a machine learning pipeline from scratch** in order to predict movie genres based on their plot descriptions, with the following goals:

1. Build a dataset of 1000 movies by scraping movie information from The Movie Database (TMDB) and the Internet Movie Database (IMDB).
2. Apply bag-of-words with term frequency inverse document frequency (TFIDF), word2vec (w2v), and doc2vec (d2v) transformations to movie plots.
3. Use conventional machine-learning techniques to classify movies as one or more genres using the transformed plots.

Project Question

"Can we build a robust movie plot and genre dataset through web scraping, then apply bag-of-words and word2vec transformations to plot descriptions to accurately classify movie genres using conventional machine learning multi-label classification techniques?"

Data Scraping & Cleaning

Data scraped using "tmdbsimple" and "imdb" python libraries.

Sources:
1. The Movie Database
2. The Internet Movie Database

Data Description & Design Considerations:

- 1000 movies
- English-language only
- From "top-rated" TMDB section
- Each classified as one or more genres (Figures 2 & 3)
- TMDB genres used as global response variable
- Only first IMDB plot scraped from list of plots
- Genres converted to binary representation (Figure 1)

Title: The Godfather
String Genres: ['Drama', 'Crime']
TMDB Genres: [18, 80]
Binarized Genres: [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

Title: Schindler's List
String Genres: ['Drama', 'History', 'War']
TMDB Genres: [18, 36, 10752]
Binarized Genres: [0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0]

Title: Psycho
String Genres: ['Drama', 'Horror', 'Thriller']
TMDB Genres: [18, 27, 53]
Binarized Genres: [0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

Title: The Dark Knight
String Genres: ['Drama', 'Action', 'Crime', 'Thriller']
TMDB Genres: [18, 28, 80, 53]
Binarized Genres: [0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0]

Title: Sing Street
String Genres: ['Comedy', 'Romance', 'Drama', 'Music']
TMDB Genres: [35, 10749, 18, 10402]
Binarized Genres: [0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0]

Figure 1. Binary Genres

TMDB Genre Key	Genre Value
12	Adventure
14	Fantasy
16	Animation
18	Drama
27	Horror
28	Action
35	Comedy
36	History
37	Western
53	Thriller
80	Crime
878	Science Fiction
9648	Mystery
10402	Music
10749	Romance
10751	Family
10752	War
10770	TV Movie

Figure 2. Genre Labels

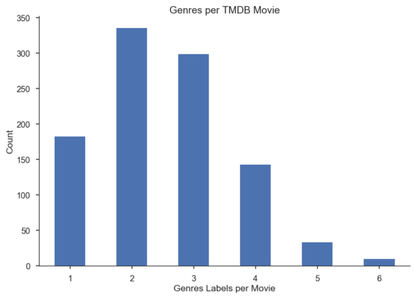

Figure 3. Genres per movie

Plot Transformations

Average plot word count: TMDB ≈ 50, IMDB ≈ 100

1. Bag-of-words TFIDF vectors

Applied term frequency inverse document frequency algorithm (Figure 4) to each plot:

- Dropped English stop words, ignored very rare and very common words.
- Vectors: 1x1171 TMDB, 1x2442 IMDB, & 1x3025 combined

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF Term x within document y

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

Figure 4. TFIDF algorithm

2. Word2vec matrix and mean vectors

- Utilized Google News word2vec model, trained on 100-billion word corpus, 3-million word model.
- Skipped words not in model (numbers, years, names)
- Each word as 300-dimension vector
- Mean vectors showed similar results to matrices

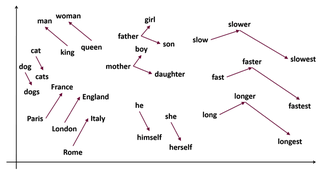

Figure 5. Two-dimensional word2vec example

3. Doc2vec vectors

- Similar to word2vec - each document is a "word."
- Relies on local document corpus to build model.
- Greatly reduces model training data (1000 movies vice 100 billion words in Google News model)

4. Explored Latent Dirichlet Allocation (LDA)

- Groups plots into clusters.
- Limited to single-genre classification (overfitting) and requires hand-labeling topics to genres.

Pre-modeling Data Analysis

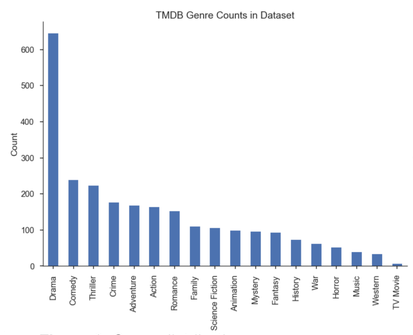

Figure 6. Genre distribution

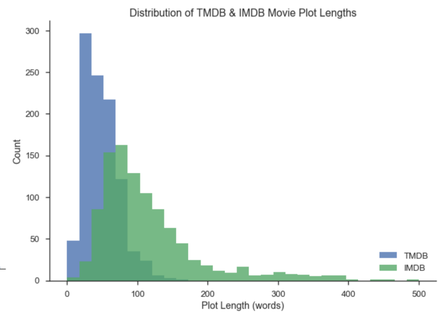

Figure 7. TMDB & IMDB plot lengths distribution

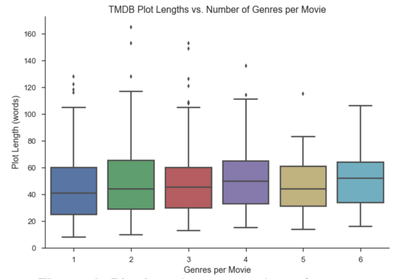

Figure 8. Plot lengths per number of genres

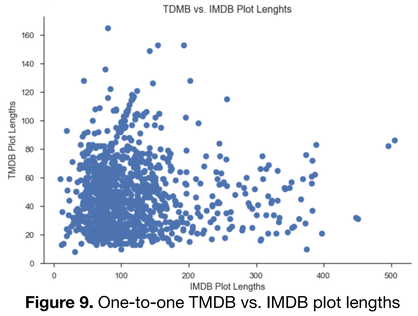

Figure 9. One-to-one TMDB vs. IMDB plot lengths

- Dataset skewed toward dramas (Figure 6)
- 2.5 TMDB genres per film
- IMDB has longer plots (Figure 7)
- One-to-one comparison has most IMDB plots double TMDB length (Figure 9).
- Plot lengths equal across films with multiple genre labels (Figure 8)

Hypothesis

"The longer IMDB or combined IMDB and TMDB plots with potentially richer w2v or d2v representations will lead to improved precision and recall metrics over sparser bag-of-words and shorter TMDB plot vector representations."

Modeling, Results & Conclusions

Features: 27 Models trained on 9 plot transformations (IMDB, TMDB, combined plots)

Multi-label Classification Models: Naïve-Bayes, Support Vector Machines (SVM), SVM with stochastic gradient descent (SGD)

Result Metrics: precision, recall, F1 (Figure 11)

	test_precision_score	test_recall_score	train_precision_score	train_recall_score
Naive-Bayes-combined_bow	0.725255	0.377953	0.982521	0.926914
Naive-Bayes-combined_doc_vec	0.166339	0.255906	0.163924	0.254321
Naive-Bayes-combined_w2v_mean	0.456782	0.267717	0.583491	0.274074
Naive-Bayes-imdb_bow	0.596821	0.340551	0.948176	0.780741
Naive-Bayes-imdb_doc_vec	0.166339	0.255906	0.163924	0.254321
Naive-Bayes-imdb_w2v_mean	0.178442	0.253937	0.605336	0.262716
Naive-Bayes-tmdb_bow	0.619958	0.301181	0.970122	0.79358
Naive-Bayes-tmdb_doc_vec	0.166339	0.255906	0.163924	0.254321
Naive-Bayes-tmdb_w2v_mean	0.316826	0.261811	0.494266	0.265185
SGD-combined_bow	0.166339	0.255906	0.163924	0.254321
SGD-combined_doc_vec	0.166339	0.255906	0.163924	0.254321
SGD-combined_w2v_mean	0.514741	0.409449	0.602544	0.42963
SGD-imdb_bow	0.166339	0.255906	0.163924	0.254321
SGD-imdb_doc_vec	0.166339	0.255906	0.163924	0.254321
SGD-imdb_w2v_mean	0.380598	0.444882	0.539381	0.497284
SGD-tmdb_bow	0.166339	0.255906	0.163924	0.254321
SGD-tmdb_doc_vec	0.166339	0.255906	0.163924	0.254321
SGD-tmdb_w2v_mean	0.470563	0.379921	0.619262	0.430617
SVC-combined_bow	0.678057	0.525591	0.95214	0.985679
SVC-combined_doc_vec	0.186744	0.525591	0.194323	0.532346
SVC-combined_w2v_mean	0.558305	0.688976	0.742788	0.893827
SVC-imdb_bow	0.607838	0.551181	0.87196	0.974321
SVC-imdb_doc_vec	0.153214	0.340551	0.162445	0.37679
SVC-imdb_w2v_mean	0.537334	0.685039	0.681017	0.864691
SVC-tmdb_bow	0.475574	0.496063	0.815313	0.946667
SVC-tmdb_doc_vec	0.186744	0.525591	0.194323	0.532346
SVC-tmdb_w2v_mean	0.540201	0.633858	0.748116	0.877531

Figure 10. Overall results from 27 models

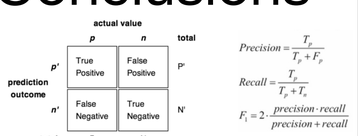

Figure 11. Result metrics

Results: SVC-combined_w2v_mean

	precision	recall
Adventure	0.58	0.73
Fantasy	0.38	0.68
Animation	0.89	1.00
Drama	0.90	0.86
Horror	0.93	1.00
Action	0.56	0.92
Comedy	0.59	0.91
History	0.90	1.00
Western	0.78	1.00
Thriller	0.61	0.86
Crime	0.62	0.81
Science Fiction	0.89	1.00
Mystery	0.73	1.00
Music	0.92	1.00
Romance	0.70	0.98
Family	0.88	1.00
War	0.95	1.00
TV Movie	1.00	1.00
avg / total	0.74	0.89

Results: Naive-Bayes-combined_bow

	precision	recall
Adventure	0.97	0.88
Fantasy	1.00	0.72
Animation	1.00	1.00
Drama	0.96	0.99
Horror	1.00	1.00
Action	0.98	0.88
Comedy	1.00	0.94
History	1.00	1.00
Western	1.00	1.00
Thriller	0.98	0.85
Crime	1.00	0.90
Science Fiction	0.99	0.93
Mystery	0.99	0.99
Music	1.00	0.82
Romance	1.00	0.85
Family	1.00	0.90
War	1.00	1.00
TV Movie	1.00	1.00
avg / total	0.98	0.93

	precision	recall
Adventure	0.43	0.74
Fantasy	0.27	0.65
Animation	0.52	0.76
Drama	0.85	0.77
Horror	0.60	0.55
Action	0.46	0.68
Comedy	0.49	0.76
History	0.20	0.16
Western	0.80	1.00
Thriller	0.42	0.77
Crime	0.56	0.73
Science Fiction	0.61	0.68
Mystery	0.28	0.37
Music	0.17	0.40
Romance	0.29	0.46
Family	0.52	0.76
War	0.70	0.84
TV Movie	0.00	0.00
avg / total	0.56	0.69

Results: Naive-Bayes-combined_bow

	precision	recall
Adventure	0.78	0.45
Fantasy	1.00	0.18
Animation	1.00	0.06
Drama	0.80	0.88
Horror	1.00	0.09
Action	0.86	0.30
Comedy	0.88	0.14
History	0.00	0.00
Western	0.00	0.00
Thriller	0.69	0.28
Crime	0.82	0.27
Science Fiction	0.80	0.32
Mystery	0.00	0.00
Music	0.00	0.00
Romance	0.40	0.07
Family	0.80	0.24
War	1.00	0.26
TV Movie	0.00	0.00
avg / total	0.73	0.38

Figure 11. Best models classification reports

	min_test_precision	max_test_precision	mean_test_precision	min_test_recall	max_test_recall	mean_test_recall
combined	0.166339	0.725255	0.402100	0.255906	0.688976	0.395888
imdb	0.153214	0.607838	0.328140	0.253937	0.685039	0.375984
tmdb	0.166339	0.619958	0.345431	0.255906	0.633858	0.374016

Figure 12. Aggregated results by plot type

	min_test_precision	max_test_precision	mean_test_precision	min_test_recall	max_test_recall	mean_test_recall
bow	0.166339	0.725255	0.466946	0.255906	0.551181	0.373360
doc_vec	0.153214	0.186744	0.169415	0.255906	0.525591	0.325241
w2v	0.178442	0.558305	0.439310	0.253937	0.688976	0.447288

Figure 13. Aggregated results by plot transformation

Best Model: SVC with combined word2vec plot vectors

- Best model has both high precision and accuracy.
- Many models overfit to drama genre.
- Doc2vec performed poorly in all models (limited training data).
- Combined plots did best confirming our richer description hypothesis.
- Future work should include modeling many movie attributes in databases and further explore LDA.

References: https://www.themoviedb.org/; https://www.imdb.com/; https://pypi.org/project/tmdbsimple/; https://pypi.org/project/IMDb/; http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MultiLabelBinarizer.html; http://www.tfidf.com/; http://filotechnologia.blogspot.com/2014/01/a-simple-java-class-for-tfidf-scoring.html; http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html; https://code.google.com/archive/p/word2vec/; http://www.samyzaf.com/ML/nlp/nlp.html; https://radimrehurek.com/gensim/models/doc2vec.html; https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation; https://matplotlib.org/; https://seaborn.pydata.org/; Spandan Madan, "An end to end implementation of a Machine Learning pipeline," https://spandan-madan.github.io/DeepLearningProject/; Mikolov et al "Efficient Estimation of Word Representations in Vector Space," https://arxiv.org/pdf/1301.3781.pdf; Minmin Chen, "Efficient Vector Representation for Documents through Corruption," https://arxiv.org/pdf/1707.02377.pdf; Chih-Wei, "A Practical Guide to Support Vector Machines, Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin," https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf; http://scikit-learn.org/stable/modules/multiclass.html; http://scikit- learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html; http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html; http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html; https://en.wikipedia.org/wiki/Support_vector_machine; https://stats.stackexchange.com/questions/31066/what-is-the-influence-of-c-in-svms-with-linear-kernel; https://en.wikipedia.org/wiki/Hinge_loss; http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html; https://www.safaribooksonline.com/library/view/python-data-analysis/9781785282287/ch10s03.html; https://alliance.seas.upenn.edu/~cis520/dynamic/2017/wiki/index.php?n=Lectures.PrecisionRecall;