PREDICTING MOVIE POPULARITIES USING THEIR GENOMES

A CS229 FINAL PROJECT

BILI XU
xbili@stanford.edu

IAN NGIAW
ianngiaw@stanford.edu

GERALD NG
geraldjs@stanford.edu

ABSTRACT

The 38 billion dollar movie industry has its successes and its flops. But which characteristic contributes to a movie's popularity? In this report, we investigate the correlation between movies' characteristics and their popularity using supervised learning algorithms.

DATASET

The genome tag score dataset was obtained from GroupLens, and is known as the MovieLens 20M dataset. A genome tag is a single characteristic exhibited by a movie.

Genome Tag Relevence score for Toy Story (1995)

Tags	Score
animal movie	0.54475
animation	0.987575
antartica	0.0375
apocalypse	0.1435
...	...

Each movie has a popularity score generated by TMDB's epic algorithm.

TMDB Popularity Score for Movies

Movie	Score
Toy Story (1995)	3.220556
Jumanji (1995)	2.252717
...	...
Finding Nemo (2003)	4.666026

We preprocessed the genome tag score data by centering it on the mean and normalizing the variance.

Centering on the mean ensures attributes are treated on the same scale, while normalizing the variance rescales the different attributes to make them more comparable.

We attempted to predict TMDB popularity using the genome tags as inputs.

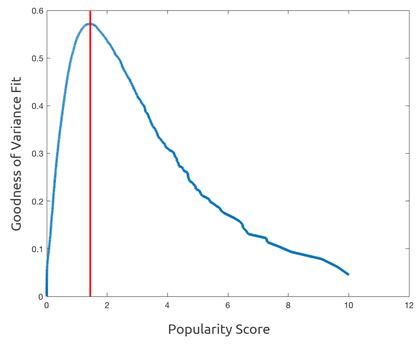
METHODS

We used textbook regression methods taught in class, mainly: Ridge Regression and Kernelized Ridge Regression. We also applied regression methods out of the course syllabus, such as Support Vector Regression, Forests Regression, and Gradient Boosting Trees.

To push the project further, we also implemented classification algorithms such as SVC, Decision Stump Boosting, Random Forests Classification, and Gradient Boosting Trees in order to give a binary answer of whether a movie is popular or not.

In order to do so, we implemented Jenks Natural Breaks Optimization to find the optimal threshold popularity score. After which we were able to convert our popularity scores into two classes: popular and unpopular.

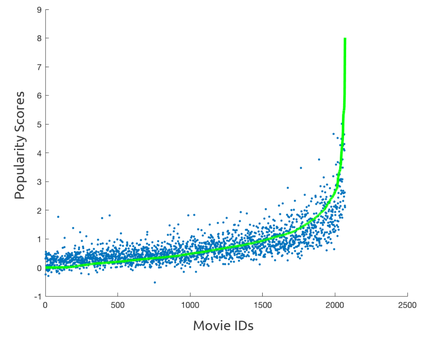
Result from Jenks Optimization



REGRESSION RESULTS

Algorithm	Kernel	Error
Ridge Regression	-	0.8328
RR w/ Regularization	-	0.8006
Kernelized Ridge Regression	R.B.F	0.2455
	Poly-2	0.2311
	Poly-3	0.2303
Support Vector Regression	R.B.F	0.2825
	Poly-2	0.2646
	Poly-3	0.2651
Random Forest Regression	-	0.3125
Gradient Boosting Regressio	-	0.2513

Kernelized Ridge Regression with Poly-3 kernel performed the best. The plot of the predicted popularity scores (blue dots) versus the actual popularity scores (green line) is provided below.



CLASSIFICATION RESULTS

Algorithm	F1 Score	Precision	Recall
Decision Stump Boosting	0.5463	0.6571	0.4674
Support Vector Classifier with R.B.F Kernel	0.5152	0.8086	0.3781
Random Forests	0.4081	0.7216	0.2846
Gradient Boosting Classificiation	0.5797	0.7143	0.4878

DISCUSSION

In regression, the best algorithm was Kernelized Ridge Regression with Poly-3 kernel. From our results, we can predict with some level of certainty the popularity of a movie based on their genome tags.

For classification, the best algorithm was Gradient Boosting Trees. The confusion matrix can be shown below:

		Predicted	
		Unpopular	Popular
Actual	Unpopular	1775	48
	Popular	126	120

It is observed that we can predict with relatively high precision (but low recall) if a movie is popular.

FUTURE WORK

Given more time, we would use our research to implement a movie recommendation system that makes use of genome tag scores, and other possible features in the dataset such as movie ratings, and movie casts.

REFERENCES

Jenks Natural Breaks Explained
https://www.ehdp.com/methods/jenks-natural-breaks-1.htm

Harper, F., & Konstan, J. (2015). The MovieLens Datasets: History and Context. *ACM Trans- actions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872*

Smola, A., & Vapnik, V. (1997). Support vector regression machines. *Advances in neural information processing systems, 9, 155-161. Chicago*

SciKit Learn Documentation
http://scikit-learn.org/stable/documentation.html