BOX OFFICE PREDICTION & IMDB RATING ANALYSIS

Bodong Zhang, Yixuan Wang

University of Utah
Salt Lake City, UT 84112

ABSTRACT

The report mainly summarizes the key concepts and implementation of our project on IMDB rating and box office prediction of various movies by using different features. In this project, we tried to estimate box office of movies and also analyze rating pattern of IMDB users by information on IMDB website. Thousands of movies data and public user data have been analyzed. Our results indicate that, with a combination of machine learning algorithms, our models have good performance on prediction.

Index Terms- Machine Learning, Clustering, Linear regression, IMDB rating, box office prediction

1. INTRODUCTION

Estimating ratings and box office of different movies is an interesting topic. Movie is very popular among almost everyone in this modern society. Ratings and box office play key roles in movie industry. Also there are abundant data on websites for us to learn. In this project, models are built by combining several machine learning algorithms and implemented on two data set from IMDB website. The first data set is movie data set, with nearly 100 features for each movie, there are about 600,000 movies in this data set. The second data set is user data set, including nearly 1000 IMDB users' rating behavior.

Our goal of this project is to find the relationship between different features such as genres, ratings and box office, etc. We want to find the most dominant features affecting the box office and the ratings of movies, and the rating behavior patterns of IMDB users.

All the programs and sample data set can be seen from our Github repository [1].

Related Works

One of the most famous film rating competition is Netflix Prize which basically tries to implement machine learning technique to find the best rating of movies from a training data set of more than 100,000,000 ratings that nearly 500,000 users gave to 17,770 movies. The 1 million dollar grand prize was awarded to a team implementing ensemble methods in machine learning including a generative stochastic artificial neural network model and gradient boosting. The result outperforms the Netflix's own algorithm by more than 10 percent [2].

Main Ideas

Inspired by ensemble methods in Netflix Prize and Adaboost introduced in class, sometimes any single machine learning algorithm may not has satisfied results. So in this project, methods are constructed based on several different algorithms. First, clustering method is used for selecting similar examples in data set. Then linear regression and its development play main roles in prediction. To extract valid features and have better performance, we explored Ridge regression and Lasso method and implemented them in this project. Cross-validation is also used to determine parameters in models.

The rest of this report is organized as follows. Section 2 describes general Machine Learning algorithms we explored related to this project. Section 3 mainly explains details about implementation. In section 4, we present some experimental results. Finally, we conclude the report and described future plan in section 5.

2. ALGORITHMS

In this part, we will mainly introduce algorithms that we used in the project. Then in the next section the detailed implementation would be explained. First, we learned clustering method, especially K-means clustering method which helps us to select suitable data from dataset. Next, linear regression is a popular method to predict output given input. Then, we further explored Ridge regression and Lasso for selecting features.

K-means Clustering

Clustering is a kind of unsupervised learning method for data analysis. Given a set of data, the algorithm tries to group them so that the data instances in the same group looks more similar to each other than in other groups based on features that are used in clustering. The reason why clustering is unsupervised learning is that there is no any given information where each of data instances should be placed.

K-means clustering is a clustering method to partition data set into k groups given k. First the number of groups that we want data set to partition into should be set, and we assume the number is k. Then we select k initial centers as the clustering center for each group. Then we can run the algorithm described below:

1 Given k, initialize the location of k centers as C_1, C_2, …, C_k

2 For each data d_i, we put it in j-th group if center C_j is the center that is closest to data d_i. So each data would be partitioned into one of groups.

3 For each cluster center C_j, we use the location of all the data examples in j-th group to calculate the new center of j-th group, then update C_j.

4 Go to step 2 to continue iteration.

The algorithm stops when the assignment of all data examples does not change. Then we have k clustering centers and all the data instances are put into one of k groups. When a new data comes, we can partition it based on the distance between the data and each center, and then we use the above algorithm to iterate until no further changes of assignment of data happen.

K-means clustering helps to find similar dataset for further process. The selection of k can be determined by some rules or prior knowledge and requirement.

Silhouette Analysis

To determine the best value of K for K-means, one of the most popular algorithm is silhouette analysis. It is used to study the separation distance between the different clusters, which means it measures how close each point in one cluster is to the points in other nearby clusters [3].

Linear Regression

Linear regression is a simple supervised learning method used to predict a quantitative output Y given quantitative inputs X. The hypothesis is that there is a linear relationship between input X and output Y. The mathematical description is below:

$$\hat{Y}_i=\beta_0+\beta_1 X_{i,1}+\beta_2 X_{i,2}+\ldots+\beta_d X_{i,d}$$

The regression coefficients β_0, β_1, …, β_d are unknown. The goal is to make sure the average difference between real output Y and estimated $\hat{Y}$ calculated by linear regression is as less as possible in the whole data set. Residual sum of squares(RSS) is used to measure the difference.

$$\text{RSS}=\sum_i(\text{Yi}-\hat{Y}\text{i})^2$$
$$=\sum_i(\text{Yi}-\beta_0-\beta_1X_{i,1}-\cdots-\beta_dX_{i,d})^2$$

$X_{i,j}$ means the input of j-th dimension in i-th example, Y_i means the true value of output in i-th example, $\hat{Y}$i means the estimated output calculated by linear regression model. We need to choose the best β_0, β_1, …, β_d such that RSS is minimized. By taking derivative, the estimated β can be calculated by:

$$\hat{\beta}=(\sum_i x_i x_i^T)^{-1}(\sum_i x_i y_i)$$

Where x_i=(1, $X_{i,1}$,…, $X_{i,d}$), $\hat{\beta}=(\hat{\beta}_0,\hat{\beta}_1, \ldots, \hat{\beta}_d)$

For the examples that can not be represented by linear model, we can first transform features by non-linear transformation and then implement the above algorithm.

Ridge Regression

In linear regression, if the number of features is large and there are not much training instances, the training results would be bad. Dimension reduction could be used to help eliminate irrelevant features by setting weight values to 0. In this section, Ridge Regression will be introduced, and another similar method would be introduced in next part.

The Ridge regression is similar to least squares method mentioned above. The main difference is that the goal is to minimize RSS plus penalty especially when coefficients in β are large.

$$\sum_i(\text{Yi}-\beta_0-\beta_1X_{i,1}-\cdots-\beta_dX_{i,d})^2+\lambda\sum_{j=0}^d\beta_j^2$$
$$=\text{RSS}+\lambda\sum_{j=0}^d\beta_j^2$$

λ is a parameter that balances the two terms. If λ is close to 0, then Ridge regression would act just like normal least squared method. If λ is very large, then all coefficients tend to shrink to 0. So it is very important to tune the λ, an efficient way is cross-validation.

Another thing that is worth paying attention is that each features in x should be standardized such that variance equals to 1 before implementing Ridge regression. In least squared method, if variance of j-th feature is large, we can shrink the absolute value of β_j without any side-effect. However, in Ridge regression, the second term of minimization formula would also be influenced by that.

The reason why Ridge regression outperforms least squared method in cases where size of instances is small is that Ridge regression use bias to trade-off variance. In least squared method, a little change in training set would greatly influence the training results.

Another form of Ridge regression is

$$\min_{\beta} \sum_i (\mathrm{Yi} - \beta_0 - \beta_1 X_{i,1} - \cdots - \beta_d X_{i,d})^2$$

$$\text{subject to } \sum_j \beta_j^2 \leq \mathrm{s}$$

s is tuning parameter which plays the same role as λ. If s is very large, then the constraint would have no effect when calculating minimum of sum of square. If s is close to 0, then all the coefficients in β would be close to 0.

Lasso

Lasso is quite similar to Ridge regression but change $\sum_j \beta_j^2$ to $\sum_j |\beta_j|$ in the constraint. We are trying to minimize the following formula by tuning β.

$$\sum_i (\mathrm{Yi} - \beta_0 - \beta_1 X_{i,1} - \cdots - \beta_d X_{i,d})^2 + \lambda \sum_j |\beta_j|$$
$$= \mathrm{RSS} + \lambda \sum_j |\beta_j|$$

Another form of Lasso is

$$\min_{\beta} \sum_i (\mathrm{Yi} - \beta_0 - \beta_1 X_{i,1} - \cdots - \beta_d X_{i,d})^2$$

$$\text{subject to } \sum_j |\beta_j| \leq \mathrm{s}$$

The drawback of Ridge regression is that although some coefficients in β can be small, they can not exactly reach zero. The “sum of square” implies that the formula try to make every coefficient not too large rather than being zero. However, in Lasso some coefficients in β can be exactly zero. The figure [4] below from James clearly shows how Ridge regression and Lasso works and how Lasso tends to make more coefficients be zero.

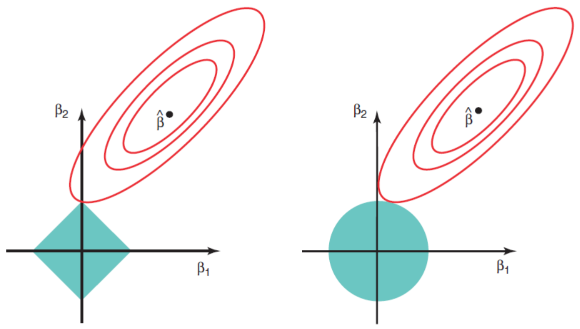

Fig.1. [4] Comparison of Ridge regression and Lasso

Assume we both use second form of Ridge and Lasso to implement. The left graph in figure 1 is Lasso, the right graph in figure 1 is Ridge regression. The green area means the constraints that $\sum_j |\beta_j| \leq$s and $\sum_j \beta_j^2 \leq$s. So the β has to be in the area to fulfill constraints. Point $\hat{\beta}$ is the estimation of β by least squared method that minimizes $\sum_i (\mathrm{Yi} - \beta_0 - \beta_1 X_{i,1} - \cdots - \beta_d X_{i,d})^2$. All points in the same red oval have the same value of $\sum_i (\mathrm{Yi} - \beta_0 - \beta_1 X_{i,1} - \cdots - \beta_d X_{i,d})^2$. The closer β is to point $\hat{\beta}$, the less value of $\sum_i (\mathrm{Yi} - \beta_0 - \beta_1 X_{i,1} - \cdots - \beta_d X_{i,d})^2$. So when oval expands to the edge of green area, the point at intersection would be the answer for Ridge and Lasso. Clearly the shape of green area in Lasso is like a cube while the shape of Ridge regression is a ball. So in Lasso, it is more likely that the intersection point is at the corner of cube, which exactly means that some coefficients would be zero. In the left graph, the coefficient of β_1 is zero, but in the right graph, the intersection point has no zero coefficient. So by using Lasso, there is more possibility that the solution has more zero coefficients.

Cross-validation

Cross-validation is widely used in machine learning. In homeworks of machine learning class, cross-validation is often used to optimize unknown model parameters to fit the training data as much as possible. It also helps to avoid overfitting problem. The main idea is, during training, the training set is partitioned into two subsets, the first subset (training set) which is usually larger is used to train the model, the second subset (validation set) is used to test the accuracy of the model. In most cases, the whole training set would be partitioned many times such that each instance plays the same role and cross-validation is also implemented many times according to different partition results. The validation of chosen parameter is based on the average accuracy. The reason why one round cross validation is not commonly used is that it may create bias if we just use part of training data to get the parameter that seems to be the best, also size of the validation set is much smaller than the whole training set, so the accuracy measured by one round may not be accurate.

To eliminate those drawbacks, cross-validation with multiple round is usually adopted, one of the methods is k-fold cross-validation. In k-fold cross-validation, the original training set is randomly partitioned into k equal size subsets, then the process repeats k times, with each of k subsets used once as validation set.

After determining best parameters by cross-validation, the whole training set would be used to train the model again.

3. DATA SET & IMPLEMENTATION

Data set:
We used the python library requests and beautifulsoup to scrape the data from IMDB website. IMDB actually provides lots of dataset from, but the problem is, not only the dataset has many instances that are not intact but also it has no page link attached to the movie. This caused a problem for us because we want to find the relationship between actors and movies or even users, they all have their own separate pages.

The IMDB has more than 6,000,000 movies and tv pages, most of them has no ratings or lack important information. Those movie pages won't help us to train. So we filtered the data and only kept movies whose number of votes are more than 100, and exclude tv series or video games. There're lots of web scraping rules we have to follow in order to scrape the data successfully.

Criteria that how we preliminarily choose data is described as follows.
1. For movie dataset M, only films with 100 votes or above are considered, the total number of movies met this criterion is 60,000+. There are nearly 100 features for each movie.
2. For user dataset U, users who rated more than 10 movies are considered. We scraped nearly 1000 users' public data. There are more than 30 features for each user.

The goal is trying to find the relationship between all the features and the predictor for movie ratings and movie gross.

Movie Dataset M:
In movie dataset M, some features only includes single value like below.

year	duration	metaScore	rating users	ratingAve
reviews	critics	budget	gross	

Rating users is number of users who rated the movie, ratingAve is the average score of movie based on users and the range is from 0 to 10, reviews is number of reviews, critics is also number of critics, budget is usually measured by dollars, gross is the box office we want to predict by using information from other features.

Some features include multiple values, preprocessing steps include expanding genrelist and labeling it for each movie.

genres	genres of the movie, one movie may be classified as action, romance and history, etc
reclist	Recommended movies from imdb for a specific movie
stars	Major actors/actress for a movie

There are also demographic breakdowns of ratings, indicating vote numbers and rating average from a specific type of voters. Some of them are shown below:

age1829	Including age between 18 and 29 voters' number and average rating
male_age1829	Including age between 18 and 29 male voters' number and average rating
female_age1829	Including age between 18 and 29 female voters' number and average rating
imdbstaff	Including imdb staff voters' number and average rating
us	Including US voters' number and average rating

User Dataset U:
In user dataset U, each instance stores information for one user. This dataset is used to analyze rating pattern rather than box office. The information basically include users' rating distribution by year & genre. One of the feature is the user's average rating and rating number on drama genre movie. In this project we are going to predict the average rating and number of ratings on drama genre movies of a user based on other features.

Procedures & Methods:
Before implementing all the methods, we need to clean the data. The problem is that most movies may have more than one genre they belong to, and some features are not numbers. We preprocessed the dataset in such a way that each genre expanded as a feature and all the strings feature get replaced by a floating point number. For example:

1. In the stars feature, we have all the leading actor/actress in the movie. We replaced the actor/actress name for the average movie gross they gained over the years with different weight on year. The logic is the box office of a movie in 90's has smaller impact on the

future movie box office than some movies only two years ago. The weight we chose for movies from 1990 to 1999 is 0.5, for movies from 2000 to 2009 is 1, for movies after 2010 is 2. We tried different weights in the process and those numbers generated the best predictor.

2. In the language section, we replaced the English as 1 and other language as 0

3. When predicting the box office, we only select movies after 1990 which include all the necessary information because movies before that usually don't have the box office information. The total number of movies here is 4699, a relatively small set comparing to the original 60,000.

For movie data set, first we implemented the K-means clustering methods. The reason for clustering before linear regression is that there're some significant difference between different 'groups' of movies, which we saw in the intermediate report while doing linear regression without clustering.

For selecting the best cluster number, we tried silhouette analysis on K-means with different cluster numbers. The number we tried is from 2 to 15, and 5 gave us the best average silhouette score, which is 0.65.

After the clustering procedure, we implemented several linear regression methods on those 5 clusters such as Ridge and Lasso to build this model for predicting box office. We used the genre, mpaa rating, rating user type distribution and several other features to predict the movie rating. We used all the features except rating distributions to predict the box office. Of course the data set is the cleaned data which includes only floating point numbers. We normalized the data set before implementing linear regression. As we talked about before, Ridge and Lasso has their own pros and cons when dealing with the data set.

For user dataset, similar to the procedure for movie data set, we find the best cluster for users, which is 4, and then implemented the linear regression methods mentioned above to predict the average rating and rating movie number for drama genre movie, based on information for all the other genre movies those user rated.

4. EXPERIMENTAL RESULTS

Movie data set:

After the 5 clusters K-means, there're 2776 movies in first cluster, 78 movies in second cluster, 402 movies in third cluster, 199 movies in fourth cluster and 1244 movies in fifth cluster.

The following is the weight for each feature for predicting box office on cluster 1. In the cluster, we use 80% of it as training data. If the difference between real box office and predicted box office is below 10%, we say the prediction is correct, otherwise it is incorrect.

feature	coefficient	feature	coefficient
R	-2398002.46	director	67522.55
NC-17	-88393.75	stars	19702.53
Comedy	78926.79	language	3181923.14
Drama	-1793967.35	budget	0.1384944
Family	670868.64	datePub	-59653.35
Horror	1556587.91	Other features	0

By selecting λ from 10^{-1} to 10^{5} in Ridge regression and Lasso, the MSE varies from 7.30×10^{-2} to 7.30×10^{4} , the best λ equals to 7829, 10000 iterations were implemented.

After training, we use the predictor to test on test case, we get 82.05% correctness in the test data (movies in cluster1)

User data set:

As mentioned before, K-means with 4 clusters has the highest silhouette score on user data set. The 4 clusters we get are: 618 users in first cluster, 8 users in second cluster, 175 users in third cluster, 31 users in fouth cluster.

Predicting drama voting number on cluster 1. The following table shows the weight that is learned that has the best performance. Some of the coefficients are zero because the dimension reduction property in Lasso. From the results, it is clear that biography has the most positive influence on number of ratings on drama. Some other features has negative influence, for example, if the user is more fascinated about news, the possible number of voting on drama movies would be lower.

feature	coefficient	feature	coefficient
Action	-0.1075	Fantasy	0
Adult	0	Film-Noir	0
Adventure	-0.0273	Game-Show	0
Animation	0	History	0
Biography	2.1727	Horror	0
Comedy	0	Music	0
Crime	0.3640	Musical	-0.3441
Documentary	0.0354	Mystery	0.8306
Family	-0.05042	News	-0.6752
Western	0	Sport	-0.2549
Reality-TV	0.01095	Talk-Show	0
Romance	0.9492	Thriller	0
Sci-Fi	0	War	1.1863
Short	0.1065		

By selecting λ from 10^{-4} to 10^{2}, the best λ equals to 0.0937, with 10000 iterations implemented.

5. CONCLUSION & FUTURE PLAN

Compare to the Netflix prize problem, what we lacked here is the user rating information for specific movie. We want to continue our project by getting the user rating information firstly and relate this user data set to movie data set, then try to find the best predictor for movie ratings. The other problem for the movie rating machine learning model is that the sensitivity of distance metric and the curse of dimensionality. We talked about these major problems in class, to solve these issues, one method that can be implemented is Topological Data Analysis (TDA) [5]. TDA has the potential to find the most consistent data structure (persistent homology) in the high dimensional data set and can be immune to the sensitivity of different distance metrics we choose. We did some implementations before and will continue to implement this method on our website (Figure 2) [6]. Another method we will try to do is the ensemble method from the Netflix prize contestants. We learned the advantages of ensemble methods in class and the competition using different ensemble methods is really fascinating and intriguing.

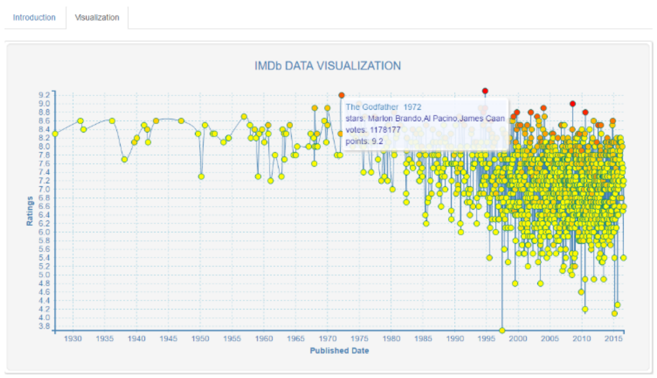

Fig.2. [1] IMDB Data Visualization

6. ACKNOWLEDGEMENT

Special thanks to Professor Vivek and all the TAs, we had a wonderful time learning fascinating machine learning theory and methods.

7. REFERENCES

[1] github.com/RespectMyAuthoritah/ML-Project

[2] Koren, Yehuda. "The bellkor solution to the netflix grand prize." Netflix prize documentation 81 (2009): 1-10.

[3] Rousseeuw, Peter J. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis." Journal of computational and applied mathematics 20 (1987): 53-65.

[4] James, Gareth, et al. An introduction to statistical learning. Vol. 6. New York: springer, 2013.

[5] Singh, Gurjeet, Facundo Mémoli, and Gunnar E. Carlsson. "Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition." SPBG. 2007.

[6] yixuanwang.info/imdb_ml/