Movie Recommender System

Aayush Grover
CSCI 5502(distance section)
Graduate Student
Student id : 105830034
University of Colorado, Boulder
aagr3102@colorado.edu

Alimulla Shaik
CSCI 5502
Graduate Student
Student id : 105835026
University of Colorado, Boulder
shal5122@colorado.edu

Camilla Lambrocco
CSCI 4502
Under Graduate Student
University of Colorado, Boulder
cala8661@colorado.edu

ABSTRACT

This paper presents an overview of of movie recommender system and describes 4 different techniques of data mining(analysis) for recommending movies based upon not only just the historical ratings of the user, but also upon many other aspects like we can recommend the best era of years for Sci-Fi movie genre and similarly for every possible genre present, we can also recommend movies as per the profession of a user, age of a user, sex of a user, and can also recommend movies based on the combination of many such attributes. In order to achieve our goal we've analysed the MovieLens 1M rating data set by using the following techniques : Data Pre-Processing, Exploratory Data Analysis,K-Means Clustering, Movie Recommendation Engine based upon Alternating Least Squares Algorithm, Decision Tree, Random Forest, Association Rule Mining and Logistic Regression. The Results and Analysis section of this report indicates all the results of the analysis tests that we've performed.

1. INTRODUCTION

The movie industry is a market of great economic interest. Most movie service providers like IMDB only gives brief information about overall rating of the movie but does not personalize user's recommendations. What has been done before considering the current movie recommender systems is that the current systems like MovieLens, netFlix, rottenTomatoes etc recommends the movies to users by just asking user to rate certain movies and based the historical preferences of users, they recommend the movie titles to users.The problem with these Movie Recommender Systems is that they work on a specific technique just simply recommends the movie by recommending the movie title based upon the historical ratings of a user, and do not consider the other parameters related to user, and that's the reason why this problem is very interesting and thus, we chose to build a recommender system in different ways using different techniques, such that this recommender system recommends movies based upon the historical preferences of users as well as the other parameters like the profession of a user, age of a user, sex of a user, prference of a user to a particular genre, etc.

To accomplish our goal of building such a movie recommender system, we aimed to develop a recommendation engine to provide statistics on ratings and trends such that the public can rely on a valuable movie ranking chart. Our recommendation engine has taken care of individual user's movie preferences and thereby generate custom movie recommendations.

We have considered and analyzed the data available on genre, year released, ratings, ocuupation of user, user ID etc. of the numerous people that throughout the years expressed their opinion on the selected movies, and eventually we accomplished building such a system by doing : (i)Data preprocessing on movie lens data set in order to obtain an efficient dataset. (ii)Exploratory data analisys(where we found many interesting patterns that are explained in the Design and Implementation section. (iii)Performed K-Means clustering in order to recommend the best era of years for every genre. (iv)Implemented a movie recommender engine using the integration of Hadoop and Apache Spark by leveraging the MlLib library of Spark. This movie recommender engine is based upon Alternating Least Squares algorithm. (v)Analysed the Movie Lens Dataset using Decision tree in order to find out the classification rules that can contribute towards good movies of every genre. (vi)Performed logistic regression on the Movie Lens Dataset in order to find out which parameter related to user as well as movie affects the rating of a movie the most.

All the above tasks have been coded in R and Scala. We have used Movie ratings data set which has 1 million records. With this huge data set we were unable to get the results as expected for k-means clustering, decision tree and logistic regression so we consulted professor and after consent from her we reduced our data set to 230K records to use for the same.But our recommender engine uses 1 million data set.

2. RELATED WORK

There are several movie recommendation systems available online that have been built before like netFlix. Most of these recommendation systems work on two algorithms that incude RBM(Restricted Boltzman) and a form of Matrix Factorization algorithm.Restricted Boltzman Machines are, simply put, fancy neural networks. The form of Matrix Factorization in use is the so-called SVD++. This is basically an asymmetric form of SVD that can make use of implicit information (just as RBMs do also). In production they are

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

combined using a linear blend.These algorithms primarily uses user's ratings, viewing history, and taste preferences to determine the recommendations for the user. But, these algorithms do not consider the parameters like age of user, sex of user, occupation of user etc. to recommend the movies. That's why we have used the several techniques mentioned in the Introduction to overcome the shortcomings of such movie recommender systems, and also these systems do not predict what rating a user can give to a movie that he has not watched earlier, while our system does this as well.

3. LITERATURE SURVEY

Recommendation system is one of the growing area and extensive research is going on. Many researchers contributed in this area and published different papers based on this system.

Barry Smith and Michael [1] developed a recommender system for online course enrollment. They provided a platform to recommend the courses to the users based on their interest. This system focuses on online courses and helps users especially students to get they what they are looking for or may be interested in.

Kleanthi Lakiotaki [2] worked on system based on multiple criteria analysis. Their system measured accuracy and compared to a Multiple Rating Collaborative Filtering approach.

Pooja Vashisth [3] proposed recommendations predicted by an internet based recommender system. It takes into account user's preferences to generate recommendations. Its architecture is based on the agents having Belief Desire Intention. They have conducted experimental study to prove that personalized interest based recommendations improve quality.

Linas Baltrunas [4] worked on some approaches to exploit context in Recommender Systems. The main focus given in the paper is to investigate new approaches that can bring a real added value to users. It provides a general architecture of context-aware Recommender Systems and analyzes separate components of this model.

Chien-Pang Kao [5] proposed A Trust-based Recommender System for Peer Production Services. Peer production model is heavily depends on information creating and sharing. More online users are relying on this type of services such as news, social media, blogs, articles, bookmarks, and various user-generated contents around World Wide Web. but, the quality of peers' contributions are not well managed. The consequence is information overloading without a practical means to assess the quality of peer production services. Based on the trust of social networks authors described a recommender system and the quality of peer production services can be assessed through the trust computing. Two fuzzy logic applications are utilized to support the decision of service choice. The experimental results proved that this recommender system can significantly enhance the quality of peer production services and can overcome the information overload problems furthermore.

Rong Hu [6] analyse different Acceptance Issues of Personality based Recommender Systems. Author evaluated an existing PBR using the technology acceptance model and also compare it with a baseline rating-based recommender. Provided guidelines on how to design personality-based recommender systems as well.

Different data sets, different types of analysis and different techniques were used in this area to develop various recommendation systems. But still researchers are working on various algorithms and incorporating other techniques to contribute to this area.

4. PROPOSED WORK; DESIGN AND IMPLEMENTATION

We are using the Movie-Lens movie dataset. While analyzing a dataset, 50-60% of the time of a data analyst is consumed in doing Data Preprocessing and Exploratory Data Analysis. Same was the case with me. Even though our dataset looks structured, it required a lot of preprocessing. As far as Exploratory Data Analysis is concerned, we can know about the modeling techniques to be applied, only when we explore our dataset thoroughly and find out different existing patterns in the dataset, due to which i performed Exploratory Data Analysis in great depth in order to find out all the patterns that exist between the attributes, so that based upon those patterns we can figure out what modeling techniques we can apply.

Our subtasks are as follows:

4.1 Data Pre-Processing

Initially we had 5 different datasets, so after loading these datasets, i had to 1st split the these datasets as per different delimeters and then organize them properly, providing the column names for each of the dataset. Then splitting the genres, forming a vector consisting of all the unique genres, then assigning this genre vector as column names of the movie dataset, and then providing logical values of '0' or '1' to each of the genre present for the movie. Then, converting the timestamps of the user rating, movie released, and $movie_video_released to dates$.

Some of the timestamps were giving absurd values of year of movie released as 2021, thus, in order to fix these outliers i simply checked the years corresponding to the movie from IMDB and replaced the outlier values with the correct ones. Then looking out for the NA values of genres and removing them. Then merging these different datasets to a unified dataset, and then removing the duplicate records from these datasets, and then looking again for the missing values and removing them.These are just the major tasks done during Data Preprocessing. There were many other tasks as well that are indicated in code.

4.2 Exploratory Data Analysis

Exploratory Data Analysis has been a major part of this project. As a part of exploratory data analysis, I performed several tasks in order to find out interesting patterns between attributes. These patterns include like : what the ages for the users are like in our data, with which I found out that most of the users are in the age group 20-40. Next I analysed what the release dates are like for the movies in our data, and I found out that most of the movies in our sample were released between 1990-2000. Next I investigated users as per their profession, In this I analysed total number of each profession that contributed to the dataset, and how these professions tend to rank movies. I found out that most of our users are students. There are very few doctors and homemakers, so we probably cannot say anything about these groups with much confidence. Students average age is very low as compared to the retired. There are more

males in our sample, some professions like engineering are completely male dominated. The professions do not rank movies evenly. Some appear more picky; for example executives seem to sometime rank movies very low and healthcare workers seem to have a very low average rating. Then I investigated the movies with respect to users in the dataset. As per movies, I found out that majority of the movies are multigenre. Movies that are documentaries have no genre associated with them.

Then I investigated the trends in movies with respect to rating and other factors in the dataset. Does genre affect the rating of a movie? Does genre matter to the average male or female? For this I found out that Noir and Animation movies are rated the highest. Women rated musical movies higher as compared to men, and thus it can be concluded that women like musical movies more as compared to men. Horror movies do not get a rating >3 from most of the women. There are very few fantasy films and they tend to be rated low. From this it could be concluded that no one wants to take risks upon making these type of movies because of this. Then I investigated relationship between occupation and movie genre preference. For this I generated a heatmap, and from this heatmap I found out some interesting trends as:

1.Most of the Lawyers have rated noir and romance movies high, i.e. most of the lawyers like noir and romance films.

2. Executives dislike most of the movie genres except Noir films.

3. The retired and elderly people do not like pure crime movies. This is the total Exploratory Data Analysis I did, the visualizations(a total of 13 graphs) can be referred in order to see all these patterns.

All these interesting patterns can be observed in the 13 graphs that I have obtained after Exploratory Data Analysis. These graphs are present inside Exploratory Data Analysis Graphs output folder.

4.3 K-Means CLUSTERING

Every data analyst knows what K-Means Clustering is, but most of them do not know how to find out the value of K, i.e. the no. of clusters. In order to find out the value of K, I have used the Elbow curve in R. Elbow curve has been provided by R with a sole motive to find out the no. of clusters for K-Means Clustering. Elbow curve plots the no. of clusters against the sum of squares of the data points, as per their variance. I performed K-Means clustering for each genre, and thus each time I found out the value of 'K' for performing clustering for each genre. I performed K-Means clustering with the Y-axis depicting the year in which the movie released, and the X-axis depicting the rating of the movie, and through this upon obtaining the cluster for each of the genre we can recommend user to look for which year movies for which genre. For example, for animation genre I obtained 4 clusters and the cluster with year 1980-2000 has the highest centroid rating. Thus, in order to recommend an era to user for a decent animation movie, we can recommend him to lookout for the movies in the era 1980-2000.

All the elbow curves for performing clustering for each genre, and the animation graphs of clustering for each genre can be found in the Graphs provided separately.

4.4 Movie Recommendation Engine using Alternating Least Square Algorithm

Alternating Least Square Algorithm is one of the best algorithms to implement a recommendation engine. I have implemented the recommendation engine using Alternating Least Squares algorithm leveraging the MLLib library of Spark. MlLib is a machine learning component in spark which provides us various algorithms like Alternating Least Squares, Guassian Mixture, Power Iteration etc. I extracted the rating information from the dataset. Then i created a rating object out of it using the rating case array, where in i have passed the parameters userId, movieId, and the rating that this guy gave to the movie. Then using the grain method i trained the model using the ALS library. This train method expects the rating object that i obtained earlier, to be passed as one of the parameters, along with the other parameters like no. of iterations, depth, and precision. Thus, i called the train method on ALS library, and trained the model with 700000 records, and obtained the desired model for the given dataset. Then the model that i obtained after training, using this model i have called the prediction method, where in i am asking the model to predict the rating for the user 789 for the moveId 123, i.e. if user 789 have watched the movie 189, then predict the rating that the user would have given to the movie. The ALS model provides us another in built method called as recommend Products, to this method i am passing the values userId and K, i.e. i am asking the model to recommend top K movies for a particular user. The result can be seen in the output folder. I have implemented this recommendation engine based on ALS algorithm using the following pipeline:

Figure 1: Movie Recommender Engine pipeline

1. I stored the data in HDFS(Hadoop Distributed File System), which acted as input to my ALS Scala implementation using MLLib.One of the problem with different type of sources is that raw data is not well structured and we need something which can store data from different sources at a single place. In this case Hadoop is the best fit which solves the problem.

2. Once, after having all the data in place, I ran the spark ALS job to do in memory computation on the data. I ran this Spark job with my cluster manager as YARN, so that the recommendation engine can also leverage the resources of YARN.

NOTE : It is possible to build a recommendation engine without using Spark. We can build a recommendation engine by only using Mahout in Hadoop, but since Hadoop reads and writes to disk, not in memory, which takes extra time. so, a recommendation engine built using Hadoop will not be a real time. While the one that i've built using Spark is a real time and this is what our use-case requires because when a user comes to our web application and asks for the recommended movies based upon his preferences, then we

should be able to provide the recommendations in real time.

4.5 Logistic Regression

Logistic regression determine the relationship between the categorical dependent variable and multiple independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution. Logistic regression predicts the probability of particular outcomes so the predicted values are probabilities (restricted to [0,1] through the logistic distribution function). Before applying logistic regression on any model, we need to ensure that the model does not have any missing values. If it has any, then replace those values with 'na'. After taking care of missing values in our data set, I have applied logistic regression on our model.

I have used glm function in R to achieve logistic regression on our 230K data set. Here, we are determining logistic analysis of our model using rating, genre, gender and occupation variables. Output of logistic analysis gives Coefficients with estimate, standard error, z- value and p-value of z-test. Estimate indicates impact on dependent variable. Means that particular variable is positively or negatively influencing the dependent variable. Standard errors can be used to estimate confidence intervals. The estimate coefficient divided by its standard error gives z-value of the particular variable. If the z-value is too big in magnitude (i.e., either too positive or too negative), it indicates that the corresponding true estimate coefficient is not 0 and the corresponding variable matters. The p-value for each term tests the null hypothesis that the coefficient is equal to zero which means no effect. A low p-value (less than common alpha level of 0.05) indicates that we can ignore the null hypothesis. To be precise, a predictor with a low p-value means it is likely to be a meaningful addition to our model because changes in the predictor's value are related to changes in the response variable. In the output of the logistic regression analysis, the variables that add meaningful value to our model are marked with '*' next to p-value. So we can omit the remaining variables which are not marked. Estimate parameter gives an idea how each variable have an impact on dependent variable. We can omit variables which does not have any impact on dependent variable (meaning which are not having large positive or negative values).

I have used anova function to analyze Deviance Table in our model. This analysis of deviance table outputs, deviance, residual degree of freedom, residual deviance and chi-square value. The number of observations minus the number of necessary relations among these observations gives us residual degree of freedom. Deviance measures fit of the model. The difference between the null deviance and the residual deviance determines how a model is doing against the null model. More the gap between these two, the better.

All these logistic analysis and deviance table analysis can be observed in the graphs that are submitted with source code separately in results folder and please refer to them.

4.6 Decision Tree

We decided that the best way to derive rating predictions, based on our dataset, was by using Decision Tree because it is interpretable, easy to use, scalable and robust. We used, as training data, a sample of $\simeq$ 200,000 pre-processed records to calculate the information gain of each attribute and, thus, found the splitting criteria of our decision tree at each node. The decision tree algorithm is fond on a divide and conquer recursive approach based on predefined categorical attributes $A_i s$. Attribute splitting needed to undergo certain criterion rules to maintain consistency and credibility of the decision tree. As a result, the implemented algorithm, performs several checks: if all output values are the same, then we return the leaf node which predicts the unique output; if the input values are balanced in a leaf node, then we return the leaf predicting the majority of the outputs on the same level; if none of the previous conditions is met then we find the attribute A_i with the highest information gain for which, in case the A_i has m values, we return the internal (non-leaf) nodes with m children and we build every child i by recursively calling the tree data structure. The decision tree was implemented using the "party" library of the statistical programming language R. This package provides the programmer with useful functions that can, considerably, fastener the implementation process. "Ctree" had a decisive role in the predictive scheme of our recommendation engine. Ctree needed to be fed with a cleaned dataset, its attributes, and a rule that determined the class upon which our model focused on: "Rating". The several attributes used to implement the rule were "Age", "Genre", "Occupation", and "Sex". The tree produced by the functions needed to be visualized in order to better interpret the decision pattern associated with the attribute chosen.

5. EVALUATION

The data set used is the Movie-Lens 1M ratings, but for some of the techniques like K-Means and Decision trees we were not able to obtain trees or clusters for 1M ratings. Thus, with the consent of Professor, for these 2 techniques we used a random sample of 230,000 records.

In this project we achieved the following results:

1. prediction of the best era of years for every genre movies.
2. recommendation of a list of movies to a particular user, and prediction of what rating a user can give to a particular movie(based upon his past preferences).
3. prediction, by means of logistic regression, of which independent variables are co related or have an impact on user rating and to determine the probability of rating using multiple independent variables like genre, gender and occupation which have impact on rating.
4. generation of decisions trees to have a predictive scheme on which we can base the recommendation of the movies.

In order to evaluate our model, for the above results we have used the following techniques:

1. For K-Means Clustering, the evaluation corresponds to the evaluation of the effectiveness of clusters, where we have the efficiency or cluster strength of each of our clustering that we implemented for every genre. For finding out the cluster strength, we have used chi-square test while performing the K-Means Clustering. Most of the clusters had a cluster strength in the range of 92% to 98%. The optimal no. of clusters that I obtained from the elbow curve are based upon calculating the sum of squares of points and their variance. The optimal no. of clusters that I obtained range from 4 to

6 for various genres. For detailed results and analysis of K-Means clustering, please refer to the *Results and Analysis section*.

2. The recommendation engine, which was based on the Alternating Least Squares Algorithm, provides two outputs. The first one is when it predicts the rating that a user can give to a particular movie, which is based upon the past ratings given by user. The second output that this model provides is that it provides a list of movies that can be recommended to a particular user. For detailed results and analysis using the Recommendation engine, please refer to *Results and Analysis section*.

3. Logistic regression is achieved using *glm* function, in R, on our 230K data set using rating categorical dependent variable and genre, gender and occupation independent variables. This function resulted in Coefficients with expected estimate, std. Error, z-value and p-value parameters. As mentioned, estimate highlighted which variables have impact on rating and p-value determined which variables are meaningful addition to our model. All genres are having low p-value and are meaningful addition to our model. This kind of meaningful variables are highlighted with '*' mark next to p-value in the logistic regression for better view. We can determine confidence intervals using standard errors. Confidence intervals are calculated using [estimate + 2x(std.error), estimate - 2x(std.error)]. Z value determined using estimate coefficient divided by its standard error. Z value indicates which variables are matter in our model. Here, Drama, war genre have large z values so these variables matter to us. This analysis helped us to determine the probability of rating for the given genre, gender or occupation variables.

   Anova function is used to analyze Deviance Table in our model. After analyzing the table we can observe that difference between null and residual deviance is wider in our model which means model is doing great against null model and a good fit. And we can also observe the drop in deviance when adding each variable one at a time in our model. Detailed report of logistic regression and deviance table with graphs are attached with source code separately in results folder and please refer them.

4. By checking age, sex and employment of the rater, and genre of the movie we were able to predict the rating of the costumer who meet specific values of the attributes just mentioned. To create the model in the first place, we applied on the cleaned data set specific rules. We chose our attributes to be, as previously mentioned, *age*, *occupation*, *sex* and *genre*. Along with the creation of the tree, thus nodes and leaves, we produced statistical values to evaluate the error associated with the splitting criterion. P-value, summation of the residual (SSE) and variance were calculated. Once the model was done, we tested it by first selecting as sample of *users* from the data set to create the model and then faking a sample of new *users*. Every *user* had a specific occupation, age and sex. By feeding the model with such values we were able to predict a rating for a specific genre of movies. We, then, mapped every rating obtained (for the specific genre) to the genre fed into the model, thus producing a table of mapped values. This table was then processed to extrapolate the rating of the genres associated with a specific movie. Finally, by averaging these extrapolated values, we predicted the probable rating that the new user would give to a specific movie. As an example, suppose we have the *user* Jenny, who is a 22 years old female engineer; the value of the attributes associated with Jenny are fed into the predictive model along with every genre of movies, once at the time. A table of all the genres is produced along with Jenny's predicted rating for each genre. If Jenny then asks to the *Recommendation engine* if she will like the movie Toy Story, the engine will then extrapolate the genres associated with Toy Story (animation, fantasy etc ...), look them up in the table of ratings created for Jenny when she registered, and finally average out the ratings associated with the specific genres to eventually produce the final predicted rating for the movie Toy Story.

6. RESULTS AND ANALYSIS

1. *Exploratory Data Analysis*
   I obtained several interesting patterns by doing Exploratory Data Analysis. All the patterns can be seen in the 13 graphs in Exploratory Data Analysis pdf. These graphs include violin graphs, ggplots, heat-map etc.

   (a) *K-Means Clustering*
   For K-Means Clustering, i found out the clustering strength for each of the 12 clustering processes performed for each of the genre. This clustering strength is calculated by the formula (betweenss/totalss), where both of these parameters have been calculated by chi-square method. Here are the results for no. of clusters found by elbow method as well as clustering strength of all 12 clustering processes for all 12 genres:

   (i) Romance
   $no.of clusters = 5$
   $betweenss = 3803451;$
   $totalss = 4008073;$
   $Clusterstrength = (3803451/4008073) = 94.9\%$
   (ii) Comedy
   $no.of clusters = 4$
   $betweenss = 4294752;$
   $totalss = 4599907;$
   $Clusterstrength = (4294752/4599907) = 93\%$
   (iii) Children's
   $no.of clusters = 4$
   $betweenss = 2790191;$
   $totalss = 2876577;$
   $Cluster_strength = (2790191/2876577) = 96.9\%$
   (iv) Action
   $no.of clusters = 5$
   $betweenss = 1701319;$

$totalss = 2039626$;
$Cluster_strength = (1701319/2039626) = 91.1\%$
(v) War
$no.of clusters = 5$
$betweenss = 2408010$;
$totalss = 2485439$;
$Cluster_strength = (2408010/2485439) = 96.9\%$

Similarly, the no. of clusters and cluster strength for all the clustering done for every genre is found and the range of the cluster strengths lies between 91% to 98% which is very efficient. All this can be verified from the K-Means Clustering output pdf.

The era of years predicted for every genre through K-Means Clustering came out to be:

(i) Romance - The best era for Romance movies as per our clustering mechanism was 1964 to 1982.
(ii) Comedy - The best era for Comedy movies as per our clustering mechanism, was 1915 to 1935(THE CHARLIE CHAPLIN ERA!!)
(iii) Childern's - The best era for Children's movies as per our clustering mechanism, was 1935 to 1955.
(iv) War - The best era for Action movies as per K-Means clustering, was 1975 to 1982
(v) Action - The best era for War movies as per K-Means clustering, was 1969 to 1982
(vi) Thriller - The best era for Thriller movies as per K-Means clustering, was 1992 to 2000.
(vii)Sci-Fi - The best era for War movies as per K-Means clustering, was 1974 to 1986(Star Wars Era)
(viii)Musical - The best era for Musical; movies as per K-Means clustering, was 1935 to 1955.
(ix)Animation - The best era for Animation movies as per K-Means clustering, was 1985 to 1995(the time when modern day animation movies started releasing).
(x)Horror - The best era for Horror movies as per K-Means clustering, was 1958 to 1970(this was the era that revolutionized the Horror movies with movies like Halloween, Alien, Dawn of the Dead etc.)
(xi)Adventure - The best era for Adventure movies as per K-Means clustering, was 1940 to 1960.
(xii)Romance - The best era for Romance movies as per K-Means clustering, was 1966 to 1982.
All these results of K-Means clustering can be verified from the K-Means-Graph pdf attached in the results folder.

(b) *Movie Recommender Engine based upon Alternating Least Square Algorithm.*
The movie recommender engine is trained with 700,000 records and it is implemented using the MlLib library of Apache Spark, and it is trained by 700,000 records and it predicts the rating that a user can give to a movie that he hasn't rated earlier and also recommends movies to a userId.
ex. to userId 789 it recommends the following movies: The GodFather(1972), Trainspotting(1996), Dead Man Walking(1995), StarWars(1977), Swingers(1996), Leaving Las Vegas(1995), Bound(1996), Fargo(1996), The Last Supper(1995), Private Parts(1997).
This output can be verified from the snapshots present in Alternating Least Square Output folder.

(c) *Logistic Regression*
The detailed report generated in R with meaningful results is shown in the table 1 below.
Deviance Residuals:

Min	1Q	1Q	3Q	Max
-2.8846	0.2762	0.3118	0.3548	0.8171

Coefficients:

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for binomial family taken to be 1)
Null deviance: 93846 on 213547 degrees of freedom Residual deviance: 91062 on 213508 degrees of freedom AIC: 91142
Number of Fisher Scoring iterations: 6

I have applied logistic regression to predict user rating and probability of rating for a given independent variables like genre, gender and occupation. Below mentioned genre, gender and occupation variables have impact on user rating and are meaningful addition to our model (variables marked '*' right next to p-value).
Deviance measures how fit is the model. The null deviance shows how well the dependent (response) variable is predicted by a model that includes only the intercept but residual deviance shows with inclusion of independent variables. As you can see in the above report, Residual deviance is smaller in value than null deviance which indicates it explains data pretty well. Deviance table analysis is shown below.

Analysis of Deviance Table
link: logit
Response: rating
Terms added sequentially (first to last)

Var	Df	Dev	Res.Df	Res.Dev	P(>Chi)
NULL			213547	93846	
genre	18	697.63	213529	93149	<2.2e-16 ***
gender	1	175.39	213528	92973	<2.2e-16 ***
occup'n	20	1910.93	213508	91062	<2.2e-16 ***

—
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

From the above report we can observe that difference between null and residual deviance is wider in our model which means model is doing great against null model and a good fit. And we can

Variable Pr(>|z|)	Estimate	Std. Error	z value
(Intercept) < 2e-16 ***	2.93299	0.04843	60.559
genreChildren's 1.39e-07 ***	-0.26956	0.05118	-5.267
genreComedy 5.59e-13 ***	-0.24729	0.03430	-7.210
genreDrama < 2e-16 ***	0.36011	0.03598	10.008
genreFilm-Noir 3.61e-09 ***	0.90817	0.15390	5.901
genreHorror < 2e-16 ***	-0.46936	0.05384	-8.718
genreMystery 0.000262 ***	0.25390	0.06956	3.650
genreRomance 6.11e-09 ***	0.24799	0.04266	5.814
genreWar 1.78e-14 ***	0.45731	0.05966	7.665
genreWestern 0.000147 ***	0.47234	0.12441	3.797
genreCrime 0.001150 **	0.18509	0.05693	3.251
genreFantasy 0.002273 **	-0.31431	0.10298	-3.052
genreThriller 0.031879 *	0.08449	0.03937	2.146
genderM 1.43e-05 ***	0.10069	0.02320	4.339
occupationartist 1.14e-08 ***	-0.39655	0.06946	-5.709
occupationentertainment < 2e-16 ***	-0.59495	0.06901	-8.622
occupationexecutive < 2e-16 ***	-1.07780	0.05424	-19.872
occupationhealthcare < 2e-16 ***	-1.53811	0.05320	-28.910
occupationhomemaker 3.00e-06 ***	-0.66575	0.14253	-4.671
occupationprogrammer 0.000184 ***	-0.19533	0.05224	-3.739
occupationretired 4.26e-05 ***	0.47124	0.11514	4.093
occupationsalesman 4.33e-09 ***	-0.56891	0.09690	-5.871
occupationstudent 1.48e-09 ***	-0.26275	0.04346	-6.046
occupationwriter < 2e-16 ***	-0.65989	0.05125	-12.877
occupationdoctor 0.004889 **	0.55044	0.19559	2.814
occupationother 0.003493 **	-0.14332	0.04907	-2.921
occupationscientist 0.024233 *	0.20295	0.09007	2.253

Table 1: Logistic Regression Analysis report

also observe the drop in deviance when adding each variable one at a time in our model.

These results can be verified in the detailed report of logistic regression and deviance table with graphs present in results folder and please refer them.

(d) *Association Rule Mining*
I have applied association rule mining on our movie data set but did not get any interesting patterns which can be useful for this project to suggest new movie recommendations to the user. I got 138 rules which are shown in scatter plot in results folder. Please refer it to verify the association rule mining results.

(e) *Decision tree*
By performing selection on the attributes considered we found slightly different rating predictions. Few values would change but overall we observed a consistent behavior of the prediction schema. Although the ratings did not differ, we noticed a considerably different pattern on the splitting of the decision tree branches. The number of nodes and the average per node defined different trends. This makes sense since as we change the attributes upon which we make our prediction we are changing the selection patter. It was comforting to notice no significant variance on the predictions, demonstrating, therefore, reliability on the method used. We applied a range cut on some attributes and we performed the same analysis we did with the original attributes. It was interesting to notice that the trees differed consistently. To further test our model we run the decision algorithm on users used to construct the prediction schema. Unfortunately we observed discrepancy with respect the true rating. This might be due to the limited amount of data points or due to the little variance presented by the ratings in the original data set. When we used the model constructed with cuts, on *Ratings* and *Age*, we noticed a bigger discrepancy than when we run it with no cuts applied. This results to be consistent with the previous observation; if the little variance of ratings, in the original training set, produced a poor predictive schema, then by reducing the variance even more, that is, by constraining *Age* and *Ratings* values into ranges, we will increase the error on the prediction. Moreover, when learning about the generation of the predictive scheme through the function *Ctree*, we observed several patterns as we increased the number of data points analyzed: the more data points the worst the prediction. This observation is not shown in the *Plots and Outputs* attachment since it was the result of a trial and error approach. As final observation from the results produced, the percentages obtained in the final nodes reflected the entire population examined out of the splitting parameters. The fact that we had a "most probable" does not mean that we will necessarily hit that prediction. As a result, we are observing a mean range of the genres with respect the

population chosen.

(f) *Random Forest*
Unfortunately the function run to produce Random Forest did not give good result. we experienced an error on the predictive schema of over 65%. We decided to not pursue this path and rather look for other functions and/or a more various data set with spreader of ratings (see section *Future Work*.)

7. SUMMARY OF PEER REVIEW SESSION

Our team had a chance to introduce and interact with other team who are developing music recommendations system.
During session, we discussed about algorithms, techniques or association rules we would be using for better results of the system.
As our both systems involves about recommendations, there will not be any right or wrong answer rather it is just a opinion.
During our discussion, we got a question from other team that what makes our recommendation system unique?
In our system, we have used algorithms and techniques to get the results or data based on user preferences and there by will apply association rules to find more interesting rules considering different options of the movies like genre. We would be using machine learning Alternating Least Square Algorithm program leveraging the MlLib of Apache Spark to enable more meaningful and produces comparative results. We feel that this will make our system unique and efficient.
We talked about how recommender system area is growing and how extensive research is going on. At the end, we received positive feedback from all the members of the other team. Both teams are working towards recommendations system using different algorithms and techniques which differentiate us from each other. It was a good experience to learn about different techniques in recommendations systems area which we will explore in the meantime to get familiar with.

8. CONCLUSION

In this paper, we have proposed a movie recommender system. Firstly, we've performed Data Pre-processing and Exploratory Data Analysis in order to see what techniques we can apply to our data set. And we came up with K-Means Clustering, Alternating Least Square Algorithm, Decision Trees, and Logistic Regression. We have categorized the movies on the basis of their genres and the year in which they were released, using K - means clustering in order to predict the best era for every genre.Then i have implemented the Movie Recommender engine using Alternating Squares Algorithm leveraging the MlLib library of Apache Spark, this model predicts the rating that a user can give to a movie, as well as it also recommends the top movies for a particular user.And then we've used decision trees to find out the interesting classification rules. We also applied logistic regression to predict which independent variables are co related or have an impact on user rating. Using this we can determine how well our model is doing against null model. After this our system is capable of suggesting new movie recommendations to the user.

9. FUTURE WORK

Given the scope and time for this project, we didn't get enough time to build a web app, where user can go create his account, rate some movies and based upon that our system would recommend a few movies to the user. As a future work, we would like to implement this web-app using node.js and react.js as our front-end application. I would also like to do some research in machine learning in order to find better combinations of parameters no. of iterations, depth, and precision for Alternating Least Square Algorithm, and would also like to look for other algorithms leveraging which we can build a movie recommender model and then compare the performances and results of both the models. Just like using K-Means clustering we've formed clusters based upon the rating of the movie and the year in which the movie was released, what we can do further in this is that we can apply the advance clustering techniques like Fuzzy Clustering, Canopy Clustering, and LDA clustering and form clusters based upon some other attributes and look for interesting patterns in those.
We can determine the probability of rating for the given genre, gender, occupation or any independent variables. To do so first we need to get the logistic analysis using *glm* function as mentioned above and then compute logistic regression of dependent variable x using independent variables x1,x2,x3...xk based on formula logit(p) = log(p/(1-p))= y0 + y1*x1 + ... + yk*xk. Here y0, y1, . . . , yk are estimates parameter values (we got from the glm function above). Then we can determine the probability using the formula p= exp(y0 + y1*x1 + ... + yk*xk)/(1+exp(y0 + y1*x1 + ... + yk*xk)). We can even determine odds from the probability and log of odds.
An alternative to the logistic analysis is a predictive tree. As previously mentioned, the decision tree developed is somehow limited. We saw that, as we increase the data points, the rating achieved flattens out resulting in a not reliable prediction. To overcome such a behavior we could choose a data set that presents more variety of ratings. This might allow the decision tree predictions on new users to be more precise and accurate. Concerning random forest, to develop better predictions and lower the relative error on the leaves, we could try to use functions of other R packages. Moreover, we could test if by increasing the dataset and by lowering the number of attributes considered, we can lower the error on the predicted ratings. Reducing the number of attributes and increasing the data points might create more consistency amongst the different trees developed in the forest.

10. REFERENCES

[1] Michael P. O'Mahony, Barry Smyth, A recommender system for online course enrolment: an initial study,

Proceedings of the 2007 ACM conference on Recommender systems, October 19-20, 2007, Minneapolis, MN, USA.
[2] Kleanthi Lakiotaki," UTA-Rec: A Recommender System based on Multiple Criteria Analysis", RecSys'08, October 23–25, 2008, Lausanne, Switzerland. ACM 978-1- 60558-093-7/08/10 (pp 219-225).
[3] Pooja Vashisth," Interest-Based Personalized Recommender System", 2011 World Congress on Information and Communication Technologies 978-1-4673-0125-1 2011 IEEE (pp 245-250).
[4] Linas Baltrunas, Exploiting contextual information in recommender systems, Proceedings of the 2008 ACM conference on Recommender systems, October 23-25, 2008, Lausanne, Switzerland.
[5] Yung-Ming Li , Chien-Pang Kao, TREPPS: A Trust-based Recommender System for Peer Production Services, Expert Systems with Applications: An International Journal, v.36 n.2, p.3263-3277, March, 2009
[6] Rong Hu," Acceptance Issues of Personality-based Recommender Systems", RecSys'09, October 23–25, 2009, New York, New York, USA. ACM 978-1-60558-435- 5/09/10 (pp 221-224)

11. APPENDIX

The following tasks have been performed and implemented by AAYUSH GROVER:

(i) Data PreProcessing

(ii) Exploratory Data Analysis

(iii) K Means Clustering

(iv) Movie Recommender engine based upon Alternating Least Square Algorithm using Apache Spark and Hadoop integration leveraging Spark's MLLib component.

The following tasks have been performed and implemented by CAMILLA LAMBROCCO:

(i) Decision Tree

(ii) Random Forest

The following tasks have been performed and implemented by SHAIK ALIMULLA ALI:

(i) Logistic Regression

(ii) Association Rule Mining

Honor Code Pledge:
"On my honor, as a University of Colorado Boulder student, I have neither given nor received unauthorized assistance."