Hybrid Music Recommendation system

A Project Report Submitted

in Partial Fulfilment of the Requirements

for the Degree of

Bachelor of Technology

in

COMPUTER SCIENCE AND ENGINEERING

by

MANNEM SRINIVAS

(Roll No. 2016BCS0021)

Indian Institute of Information Technology Kottayam

to

DEPARTMENT OF COMPUTER SCIENCE

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

KOTTAYAM-686635, INDIA

November 2019

DECLARATION

I, **Mannem Srinivas** (**Roll No: 2016BCS0021**), hereby declare that, this report entitled **"Hybrid Music Recommendation System"** submitted to Indian Institute of Information Technology Kottayam towards partial requirement of **Bachelor of Technology** in **Computer science Engineering**is an original work carried out by me under the supervision of **Dr.J.V.Bibal Benifa** and has not formed the basis for the award of any degree or diploma, in this or any other institution or university. I have sincerely tried to uphold the academic ethics and honesty. Whenever an external information or statement or result is used then, that have been duly acknowledged and cited.

Kottayam-686635 **Mannem Srinivas**

November 2019

CERTIFICATE

This is to certify that the work contained in this project report entitled **"Hybrid Music Recommendation System"** submitted by **Mannem Srinivas** (**Roll No: 2016BCS0021**) to Indian Institute of Information Technology Kottayam towards partial requirement of **Bachelor of Technology** in **Indian Institute of Information Technology, Kottayam** has been carried out by him under my supervision and that it has not been submitted elsewhere for the award of any degree.

Kottayam-686635 Dr.J.V.Bibal Benifa

November 2019 Project Supervisor

ABSTRACT

The objective of this project is to develop a music recommendation system. The system will determine the musical preferences of the users based on the analysis of their interaction during use. This way the system is able to estimate what artist or group would match user preferences to the user at a given time. It has been taken into account the fact that we do not always want to hear the same artists or genres, we do have favorite bands, but sometimes we appreciates a surprise, a new discovery. Recommendation is an important field strongly related to web business which has been intensely researched in the past years, since electronic commerce web sites started their activity. Among the reviewed approaches, many solutions were found for data analysis, data colleting, or data-objects representation. According these, models like content-based,collaborative or context-based give differenced solutions to select key information to face recommendations. In parallel to the recommender models development also evolves the mathematical world related to the most pure heuristic bases of recommender systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A Recommender System or a Recommendation System is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. These systems collect user information and choices, and use this information to improve their suggestions in future. There are various approaches that can be used to build such systems.Considerable research and projects have been done on this topic, and scientists have developed various algorithms for this purpose. This project focuses on implementing multiple algorithms and comparing them on various metrics, to determine which approach provides the best recommendation.Recommendation Systems include software tools and techniques that finds items to be recommended and displays them to a user. These suggestions relate to various decision-making processes. Music Recommender systems can be seen as a surrogate of real-world radio stations and music magazines. These real-world organizations main purpose is to promote certain artists, sometimes because the radio directors or magazine editors find noteworthy the quality of their musical works, sometimes just because of economic interest. Some people listen to these stations and read magazines in order to make decisions about what music to acquire: either by traditional means or through some share-alike network. Music recommenders have the chance of making accessible to users not only the market-defined “good music”, but also new emerging groups, minor rare music and independent labels productions.

1.1 Approaches to Recommendation Systems

The most common mathematical models used in current recommender systems have been reviewed helping the author of this project to build up a solid idea about what recommendation is and how can it be achieved. These mathematical approaches to recommenders:

Logic recommender systems

Logic recommender systems try to find an exact match among the recommendation options compared the user profile. The data representation is build using the attributes that define objects. Attribute types as used as they are with no further abstraction.

Vector space-based systems

Vector space-based systems due to its numeric data modeling, estimate which objects best suit the user profile statements. Data object surrogates represent attributes in vector form being each cell a concrete attribute.

Probabilistic Systems

Probabilistic systems estimate a concrete objects importance using a probability function. This function estimates the probability the object has in order to meet the preferences stated in the user profile.

1.2 Recommendation Phases

The recommendation process can be divided into 3 phases: Information Collection Phase,Learning Phase, and Recommendation phase.

1.2.1 Information Collection Phase

Collecting relevant information about the user to generate a user profile is the first step.Relevant information includes the user's behavior and the content of the items the user accesses.Recommender systems rely on different types of input such as, explicit feedback, implicit feedback, or hybrid feedback.

- Implicit feedback
  - For implicit feedback, the system automatically infers the user preference by monitoring their activities, such as history of purchase, time spent on web-pages, or types of videos watched on YouTube. Although this method does not require effort from the user, it is less reliable than explicit feedback. Because such information can be susceptible to click baits, it may be erroneous.
- Explicit feedback

  - This feedback is received from the user. It is in the form of ratings for the items the user has consumed. The efficiency of the system relies on the quality of rating provided by the user. This requires efforts from the user, and users are not always ready to provide these ratings. As a result, this is the shortcoming of the method.

- Hybrid Feedback
  - This feedback combines the implicit and explicit feedbacks, by taking the user behavior implicitly, while allowing users the option of providing explicit feedback voluntarily

1.2.2 Learning Phase

The learning phase uses various algorithms to filter, while using the features of information gathered during the first phase.

1.2.3 Prediction phase

The prediction phase predicts and recommends the type of items the user may prefer.

1.3 Performance Metrics

This project uses metrics to evaluate the algorithms that are being implemented. These metrics are explained below.

1.3.1 Mean Absolute Error

The most straight-forward metric is mean absolute error, or MAE. Suppose we have n ratings in the test set to evaluate. For each rating the predicted value is y, and the actual rating the user provided is x. The error for that rating prediction is the difference between these two ratings.The sum of these errors across all n-ratings in our test set is divided by n to find the average, or mean absolute error.

1.3.2 Root Mean Square Error

The root mean square error metric is more popular than MAE. This is because the RMSE metric penalizes more when the predicted rating is significantly different than the actual rating,and penalize less when the ratings are relatively similar. The difference between the calculations for these metrics is that, instead of summing the absolute values of each rating prediction error, the RMSE metric sums the squares of the rating prediction errors. Taking the square ensures the result is a positive number. This inflates the penalty for larger errors. The final step is to take the square root, which results in a number within the range of predictions.

Chapter 2

Literature survey and background

The latest generation of music listeners currently rely on music suggestions provided by music streaming services.Gone are the days, when people used to manually search for .mp3/.mp4/.wav music files and save them on their music playing electronic devices. Therefore, the music streaming services need a capable recommendation systems which can suggest music based on the listener's interests. Most of the streaming services rely on CF algorithms to suggest music. CF is just a single parameter in determining the taste of the listener. To overcome this, top streaming services use a combination of algorithms to form what is called as a hybrid recommender system.

Use of music recommendation system is fully grown because of large number of online music websites. The most challenging gap we found is that there is an utmost need to consider more contextual information like weekdays, session of day, time, and frequency of listening songs. we propose a hybrid music recommendation system which is a combination of two approaches. In the first approach we propose to use cosine similarity measure for ranking of music. The second approach considers the graph-based approach. In the graph-based approach we propose using particle swarm optimization with differential evolution to get optimized ranking of music. We recommend top-n songs by combination of these two approaches. Standard Last.fm dataset is considered for experimental purpose. Data pre-processing operation is performed on dataset to remove the noisy and inconsistent data. Comparison of our proposed model with the state-of-the-art model shows the effectiveness in the form of recall rate. Traditional collaborative filtering algorithms are applied to the field of music recommendation. However, collaborative filtering does not handle data sparse problems very well when new items are introduced. To solve this problem, some people use the logistic regression method as a classifier to predict the user's music preferences to recommend songs. Logistic regression is a linear model that does not handle complex non-linear data features. In this paper, we propose a hybrid LX recommendation algorithm by integrating logistic regression and eXtreme Gradient Boosting(xgboost).

Chapter 3

Recommendation systems

Recommendation engines filter out the products that a particular customer would be interested in or would buy based on his or her previous buying history. The more data available about a customer the more accurate the recommendations.The following are the types of recommendation system:

1. Popularity based filtering
2. Content based filtering
3. Collaborative based filtering
4. Hybrid recommender systems

3.1 Popularity based filtering

Easiest way to build a recommendation system is popularity based, simply over all the products that are popular, So how to identify popular products, which could be identified by which are all the products that are bought most, Example, In shopping store we can suggest popular dresses by purchase count.

3.2 Content based filtering

This filtering is based on the description or some data provided for that product. The system finds the similarity between products based on its context or description. The user's previous history is taken into account to find similar products the user may like. For example, if a user likes movies such as 'Mission Impossible' then we can recommend him the movies of 'Tom Cruise' or movies with the genre 'Action'.In this filtering, two types of data are used. First, the likes of the user, the user's interest, user's personal information such as age or, sometimes the user's history too. This data is represented by the user vector. Second, information related to the product's known as an item vector. The item vector contains the features of all items based on which similarity between them can be calculated. The recommendations are calculated using cosine similarity. Values calculated in the cosine similarity matrix are sorted in descending order and the items at the top for that user are recommended.If 'A' is the user vector and 'B' is an item vector then cosine similarity is given by

3.3 Collaborative filtering

Collaborative filtering models which are based on assumption that people like things similar to other things they like, and things that are liked by other

$$cos(\theta) = \frac{A \cdot B}{\|A\| \, \|B\|} = \frac{\sum_i A_i B_i}{\sqrt{\sum_i A_i^2}\sqrt{\sum_i B_i^2}}$$

Figure 3.1: Cosine similarity

people with similar taste.For example, if the user 'A' likes 'Coldplay', 'The Linkin Park' and 'Britney Spears' while the user 'B' likes 'Coldplay', 'The Linkin Park' and 'Taylor Swift' then they have similar interests. So, there is a huge probability that the user 'A' would like 'Taylor Swift' and the user 'B' would like 'Britney Spears'. This is the way collaborative filtering is done.

In these type of recommendation systems are recommending based on nearest neighbors, nearest neighbor approach used to find out either similar users or similar products, It can be looked at two ways,

1. User based filtering
2. Item based filtering
3. Restircted boltzmann machines

3.3.1 User based filtering

Find the users who have similar taste of products as the current user , similarity is based on purchasing behavior of the user, so based on the neighbor purchasing behavior we can recommend items to the current user.

3.3.2 Item based filtering

Recommend Items that are similar to the item user bought,similarity is based on co-occurrences of purchases Item A and B were purchased by both users X and Y then both are similar.

3.3.3 Restricted boltzmann machines

Restricted Boltzmann machines can be used to build a recommender system for items ratings. The RBM recommender system can learn the probability distribution of ratings of items for users given their previous ratings and the ratings of users to which they were most similar to. Then RBM recommender system uses the learned probability distribution to predict ratings on never-before-seen items.

The learning rule is much more closely approximating the gradient of another objective function called the Contrastive Divergence

3.4 Hybrid Recommender systems

Hybrid Recommendation systems are combining collaborative and content-based recommendation can be more effective. Hybrid approaches can be implemented by making content-based and collaborative-based predictions separately and then combining them.

Chapter 4

Proposed Architecture

The proposed Architecture is design a Framework which takes the dataset create a prepossessed dataset required to train the recommendation models. This Architecture consists of following modules

- DataLoader
- DataGenerator
- ModelBuilder
- ModelFactory
- HybridModel
- Performance Metrics

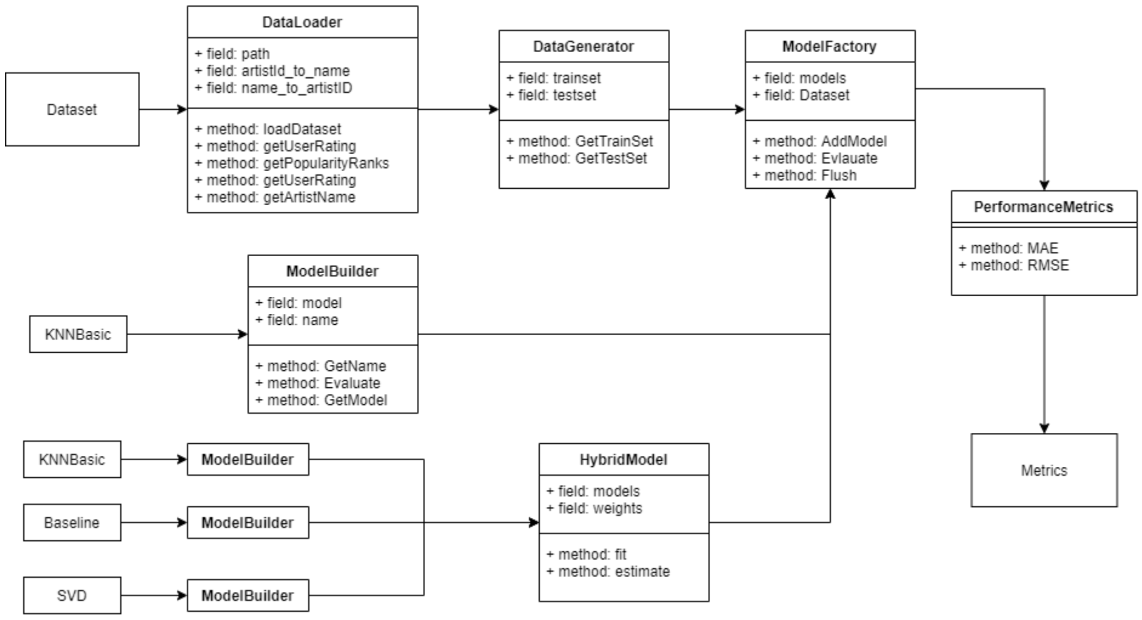


Figure 4.1: Proposed architecture

4.1 DataLoader

This module is used to load the raw dataset and returns the necessary details of the dataset. It consists of following functionalities:

- loadData : Returns the user-item-rating datset loaded
- getUserRating : Returns the particular user ratings given user as input
- getPopularityRanks : Returns the popular items based on the counted ratings
- getArtistName : Returns Artist name given Artist Id
- getArtistId : Returns Artist Id given artist name

4.2 DataGenerator

This module is used to create the training and testing datasets required for the machine learning algorithms to get trained on train dataset and test the performance on test dataset. It consists of following functionalities:

- getTrainDataset: Returns the 75 percent of dataset items randomly choosen by default for the training the Algorithms
- getTestDataset : Returns the 25 percent of dataset items remained after choosing the training dataset

4.3 DataGenerator

This module is used to create the training and testing datasets required for the machine learning algorithms to get trained on train dataset and test the performance on test dataset.This module takes the dataset from the DataLoader Module. It consists of following functionalities:

- getTrainDataset: Returns the 75 percent of dataset items randomly choosen by default for the training the Algorithms
- getTestDataset : Returns the 25 percent of dataset items remained after choosing the training dataset

4.4 ModelFactory

This module is used to store a set of ML models to train the dataset and returns the performance of each Model using metrics of Performance Metrics module .This module has the dataset returned from the DataGenerator module.It takes each ML model returned from the ModelBuilder Module. It consists of following functionalities:

- AddModel: This is used to append the ML model of ModelBuilder class to the set of ML models in ModelFactory
- Evaluate : This makes the each ML model stored in ModelFactory get trained on the dataset and returns the performace metrics by testing model with testing dataset
- FlushAllModels: This deletes all the ML models stored.

4.5 ModelBuilder

This module is used to create the a Machine Learning Algorithm of surprise library. It stores the details of ML Models. It consists of following functionalities:

- getModel: Returns the algorithm loaded into the modelbuilder object
- getName : Returns the name of the algorithm loaded.

4.6 PerformanceMetrics

This module takes the predictions estimated by ML models and returns the accuracy metrics. It consists of following functionalities:

- MAE: Returns the MAE score of the predictions

- RMSE: Returns RMSE score of the predictions.

4.7 HybridModel

This module is the subclass of AlgoBase class of the surprise library.It is used to combine the estimations of multiple ML Models based their preference weights.It consists of following functionalities:

- fit: fits the training dataset
- estimate: estimates the weighted-average of estimations of the ML models loaded into this module

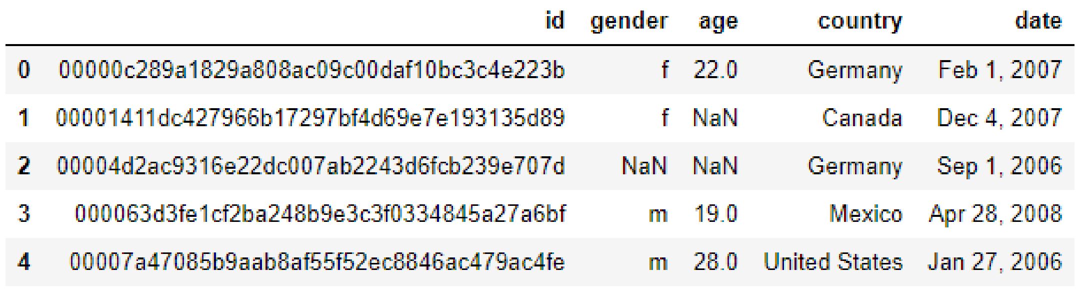
Chapter 5

User Segmentation

Customer segmentation groups similar customers together, based on purchasing behavior, demographic, preference and other information. It helps sales teams and marketing teams get a better understanding of existing customers, and identifying/targeting potential customers. After that, using personalized product recommendation approach to boost sales is the final goal.Segmentation isn't just the key to generating more product recommendations—it's the ideal method for generating better recommendations. This gives you the upper hand when it comes to driving sales, creating brand loyalty and creating better marketing emails.

5.1 User demographic dataset

This part of project is to find segment the users in the dataset based on their demographic features. This consists of following operations.The dataset consists of the demographic features of users which is shown in the below figure.It



	id	gender	age	country	date
0	00000c289a1829a808ac09c00daf10bc3c4e223b	f	22.0	Germany	Feb 1, 2007
1	00001411dc427966b17297bf4d69e7e193135d89	f	NaN	Canada	Dec 4, 2007
2	00004d2ac9316e22dc007ab2243d6fcb239e707d	NaN	NaN	Germany	Sep 1, 2006
3	000063d3fe1cf2ba248b9e3c3f0334845a27a6bf	m	19.0	Mexico	Apr 28, 2008
4	00007a47085b9aab8af55f52ec8846ac479ac4fe	m	28.0	United States	Jan 27, 2006

Figure 5.1: Finding the optimal clusters using WCSS

consists of user id, gender,country and date. The clustering techniques cannot be applied directly on the dataset. The dataset need to preprocess to do that. The dataset has gender,age and year categorical variables. These variables are label encoded.Now the dataset is ready to apply clustering techniques.

5.2 Applying clustering Techniques

K-means clustering is a clustering method that subdivides a single cluster or a collection of data points into K different clusters or groups.The algorithm analyzes the data to find organically similar data points and assigns each point to a cluster that consists of points with similar characteristics. Each cluster can then be used to label the data into different classes based on the characteristics of the data.K-Means clustering works by constantly trying to find a centroid with closely held data points. This means that each cluster will have a centroid and the data points in each cluster will be closer to its centroid compared to the other centroids.

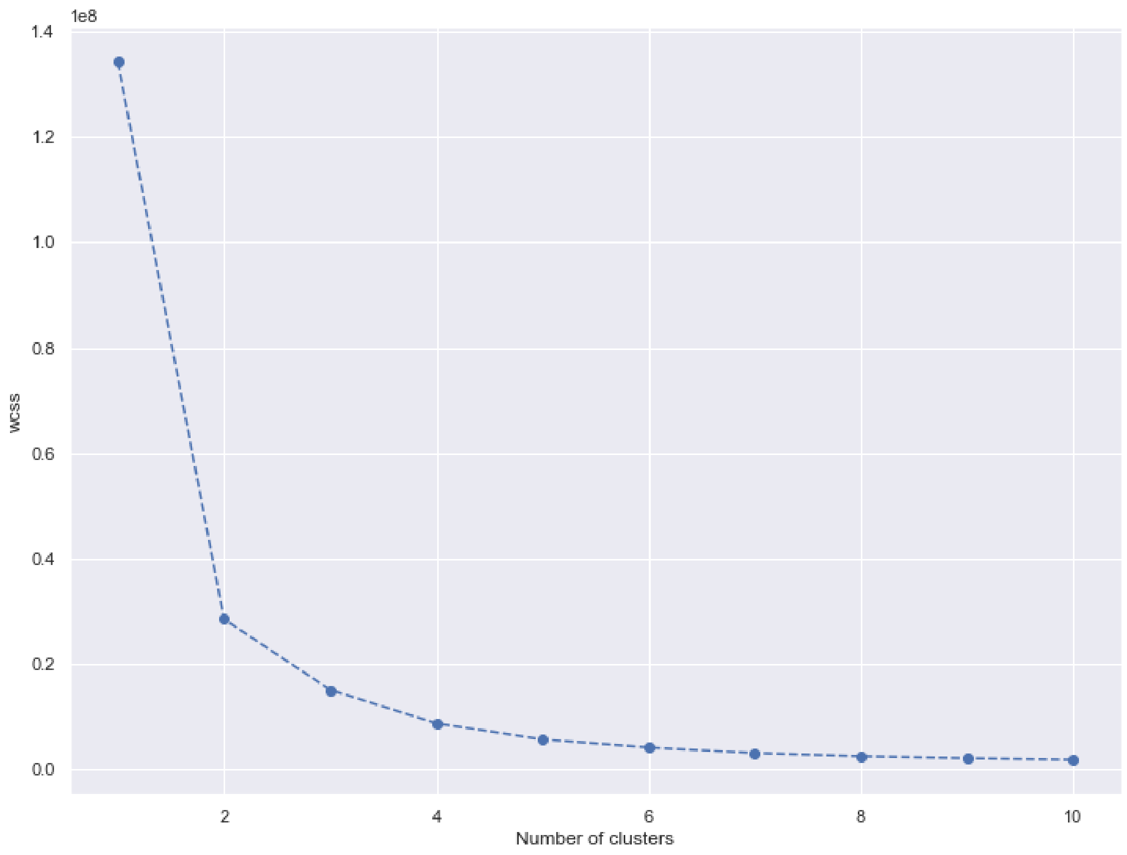


Figure 5.2: Finding the optimal clusters using WCSS

5.2.1 Finding Optimal clusters

The number of clusters that we choose for a given dataset cannot be random. Each cluster is formed by calculating and comparing the distances of data points within a cluster to its centroid. An ideal way to figure out the right number of clusters would be to calculate the Within-Cluster-Sum-of-Squares (WCSS). It was found that users can be segmented into 3 clusters.

5.3 Applying PCA along with K-means

It is a common practice to apply PCA (principal component analysis) before a clustering algorithm (such as k-means). It is believed that it improves the clustering results in practice (noise reduction). PCA is applied on the dataset and cummulative variance of the components is plotted to find out the no.of.components to retain the expected variance. The total variance is the sum of variances of all individual principal components.The fraction of variance explained by a principal component is the ratio between the variance of that principal component and the total variance.

From the graph one can retain 10 components with explained variance of more than 98 percent. K-means clustering is applied on the components and again wcss scores are plotted and found that the optimal no.of clusters are matched with number without PCA i.e 3

5.4 Visualization of segments

The clusters are plotted with any of two components to view how they are segmented. The graph clearly shows the how segments are formed apart from each other in different colors.

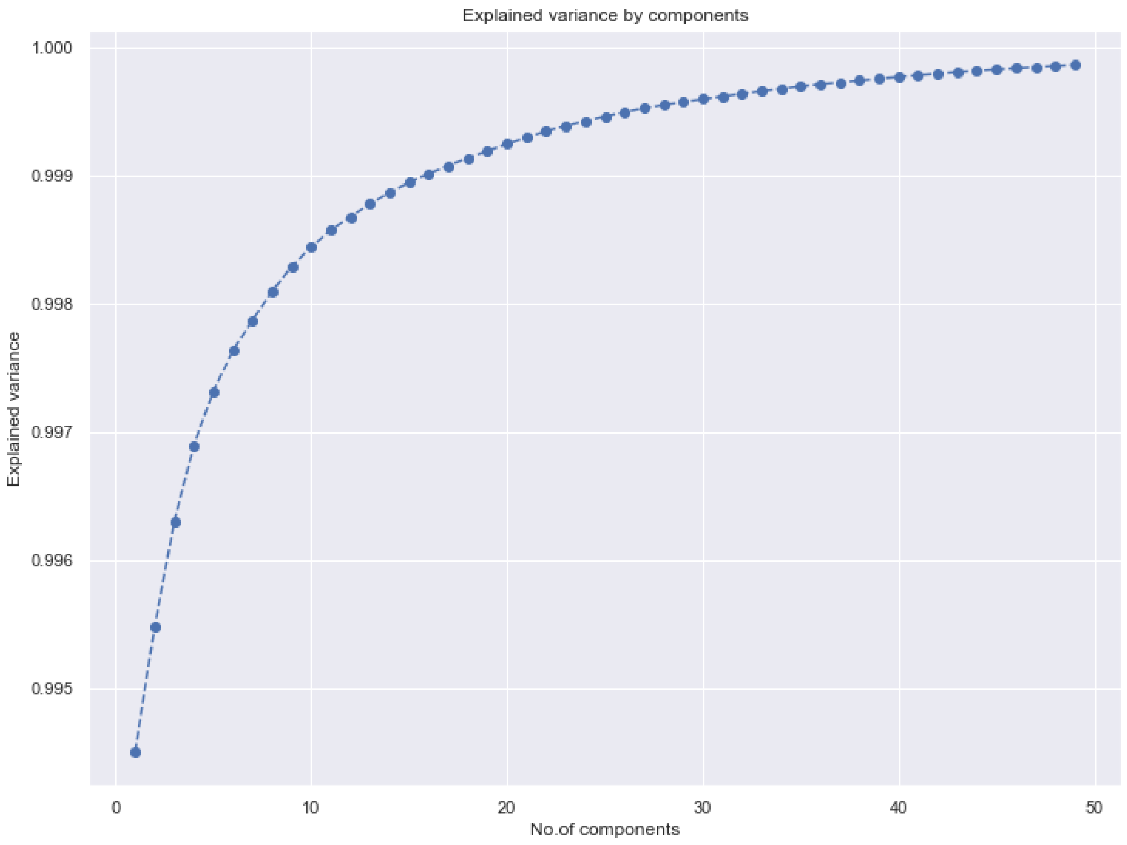


Figure 5.3: Cumulative Explained Variance by components

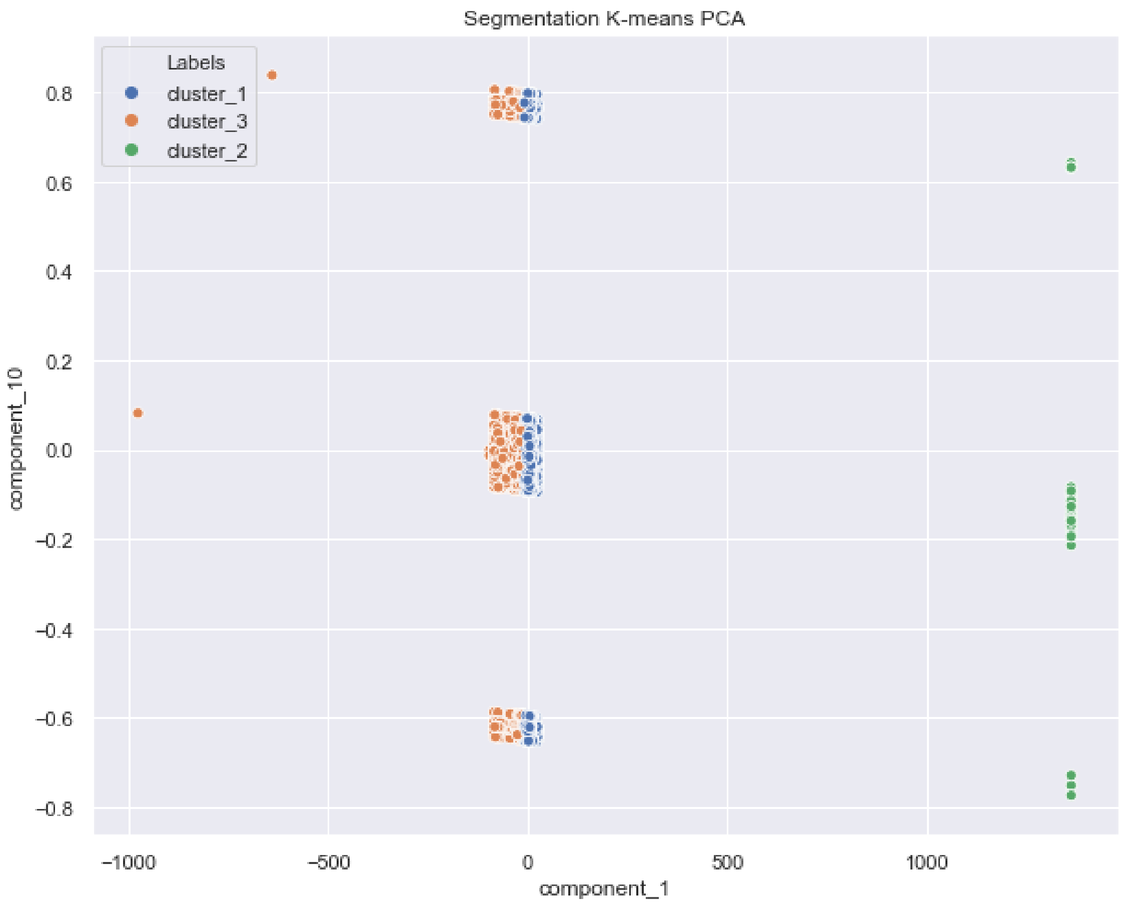


Figure 5.4: Visualization of user clusters

Chapter 6

Experimental results

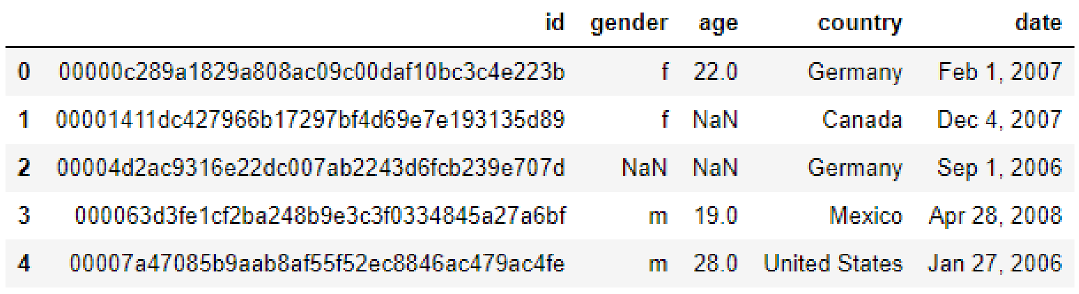
In this chapter we present the Dataset used, Libraries and tools used for the project along with experimental results in graphical representation.

6.1 Datasets used

For this project two datasets are used. one which has user demogrpahic data and another one has the listening count of songs associated with artist ID, artist name by the users defined with user ID. The dataset is subset of lastfm dataset with 9000 unique users.

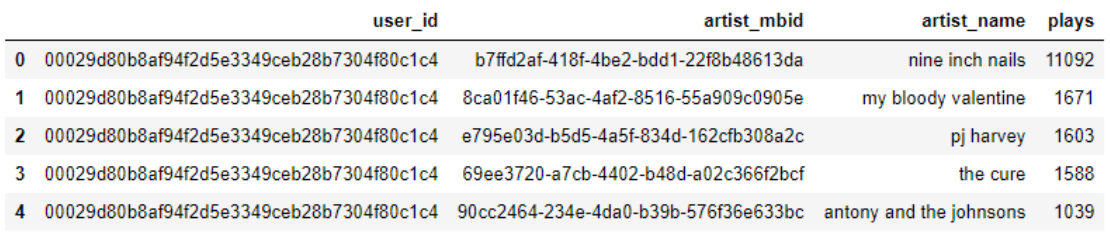
6.2 Tools used

- Numpy
  - NumPy is a scientific linear algebra library that deals efficiently with matrices and vectors. This library is used in Keras and the



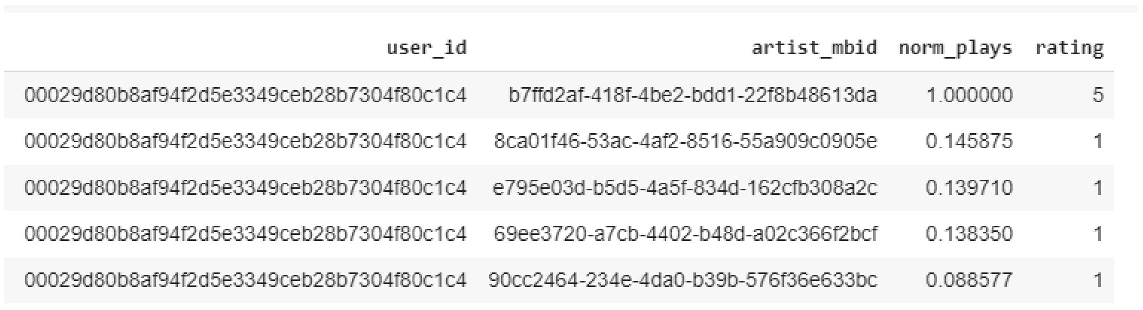
	id	gender	age	country	date
0	00000c289a1829a808ac09c00daf10bc3c4e223b	f	22.0	Germany	Feb 1, 2007
1	00001411dc427966b17297bf4d69e7e193135d89	f	NaN	Canada	Dec 4, 2007
2	00004d2ac9316e22dc007ab2243d6fcb239e707d	NaN	NaN	Germany	Sep 1, 2006
3	000063d3fe1cf2ba248b9e3c3f0334845a27a6bf	m	19.0	Mexico	Apr 28, 2008
4	00007a47085b9aab8af55f52ec8846ac479ac4fe	m	28.0	United States	Jan 27, 2006

Figure 6.1: user demographic data



	user_id	artist_mbid	artist_name	plays
0	00029d80b8af94f2d5e3349ceb28b7304f80c1c4	b7ffd2af-418f-4be2-bdd1-22f8b48613da	nine inch nails	11092
1	00029d80b8af94f2d5e3349ceb28b7304f80c1c4	8ca01f46-53ac-4af2-8516-55a909c0905e	my bloody valentine	1671
2	00029d80b8af94f2d5e3349ceb28b7304f80c1c4	e795e03d-b5d5-4a5f-834d-162cfb308a2c	pj harvey	1603
3	00029d80b8af94f2d5e3349ceb28b7304f80c1c4	69ee3720-a7cb-4402-b48d-a02c366f2bcf	the cure	1588
4	00029d80b8af94f2d5e3349ceb28b7304f80c1c4	90cc2464-234e-4da0-b39b-576f36e633bc	antony and the johnsons	1039

Figure 6.2: Raw dataset



user_id	artist_mbid	norm_plays	rating
00029d80b8af94f2d5e3349ceb28b7304f80c1c4	b7ffd2af-418f-4be2-bdd1-22f8b48613da	1.000000	5
00029d80b8af94f2d5e3349ceb28b7304f80c1c4	8ca01f46-53ac-4af2-8516-55a909c0905e	0.145875	1
00029d80b8af94f2d5e3349ceb28b7304f80c1c4	e795e03d-b5d5-4a5f-834d-162cfb308a2c	0.139710	1
00029d80b8af94f2d5e3349ceb28b7304f80c1c4	69ee3720-a7cb-4402-b48d-a02c366f2bcf	0.138350	1
00029d80b8af94f2d5e3349ceb28b7304f80c1c4	90cc2464-234e-4da0-b39b-576f36e633bc	0.088577	1

Figure 6.3: Dataset processed for Training

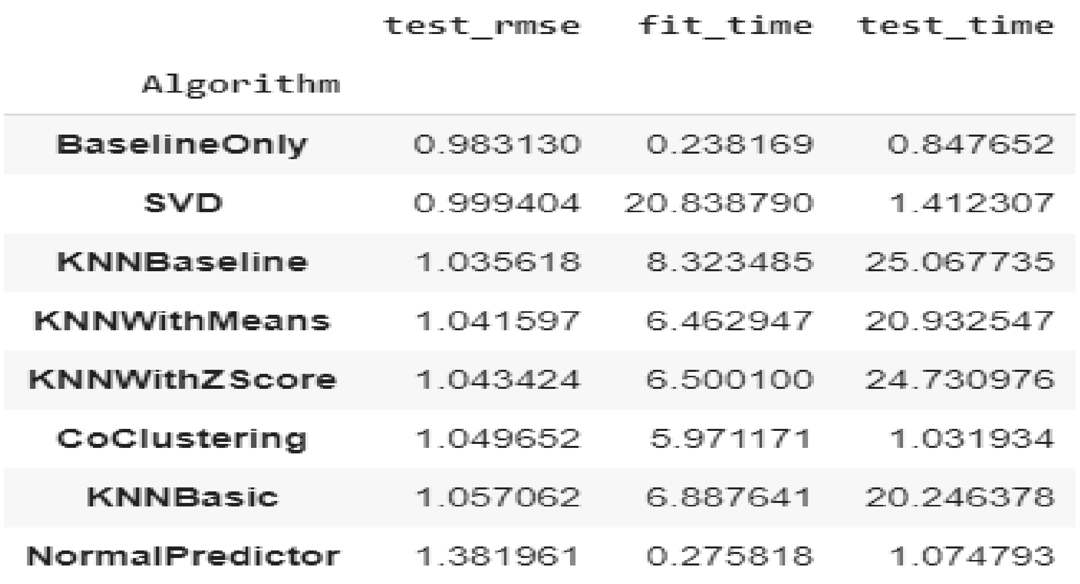
  training and testing data have to be NumPy arrays before feeding them to any network. Moreover, we use NumPy in the encoding and decoding part to deal with matrices (e.g. adding a vector to multiple rows in a matrix).

- Jupyter Notebook
  - A web application for data visualization and statistical modelling. We use Jupyter Notebook in experimentation to be able to execute blocks of code in any sequence we choose. This helps to rapidly prototype experiments and NN models.
- surprise library
  - Surprise is a Python scikit building and analyzing recommender systems that deal with explicit rating data.
- Scikit-learn
  - Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms
- Matplotlib
  - Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython.

6.3 ML algorithms and results

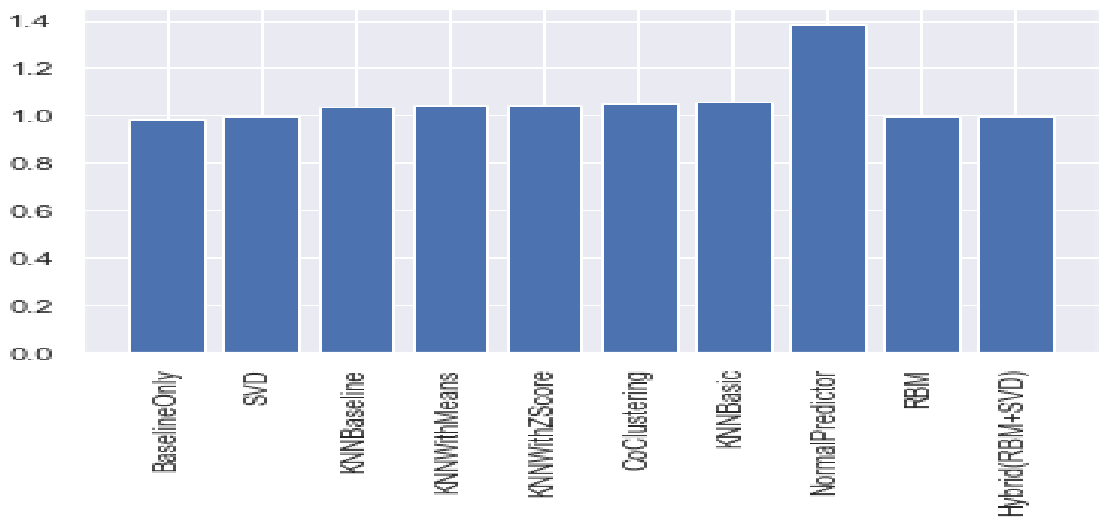
The following algorithms has been trained on the dataset and metrics has been shown in tabular form

1. **NormalPredictor** : Algorithm predicting a random rating based on the distribution of the training set, which is assumed to be normal.
2. **BaselineOnly**: Algorithm predicting the baseline estimate for given user and item.
3. **KNNBasic**: A basic collaborative filtering algorithm.
4. **KNNWithMeans**:A basic collaborative filtering algorithm, taking into account the mean ratings of each user.
5. **KNNWithZScore**: A basic collaborative filtering algorithm taking into account Zscores of rating.
6. **KNNBaseline**: A basic collaborative filtering algorithm taking into account a baseline rating.
7. **SVD**: The famous SVD algorithm, as popularized by Simon Funk during the Netflix Prize.When baselines are not used, this is equivalent to Probabilistic Matrix Factorization.
8. **CoClustering**: A collaborative filtering algorithm based on co-clustering.
9. **RBM**: A deep learning based collaborative filtering algorithm.
10. **Hybrid Model**: A Hybrid model of RBM and SVD.



	test_rmse	fit_time	test_time
Algorithm			
BaselineOnly	0.983130	0.238169	0.847652
SVD	0.999404	20.838790	1.412307
KNNBaseline	1.035618	8.323485	25.067735
KNNWithMeans	1.041597	6.462947	20.932547
KNNWithZScore	1.043424	6.500100	24.730976
CoClustering	1.049652	5.971171	1.031934
KNNBasic	1.057062	6.887641	20.246378
NormalPredictor	1.381961	0.275818	1.074793

Figure 6.4: Results



cm

Figure 6.5: RMSE of algorithms

Chapter 7

Conclusion

On training the different ML models on dataset. It was found that BaselineOnly model got the less RMSE i.e 0.983130 following SVD a collaborative filtering with RMSE of 0.99. These models have less RMSE compared to NormalPredictor which recommends Items to users randomly.

RBM model which a deep learning model for recommendation systems has been trained on the dataset with 10 epochs resulted in RMSE error of 0.998.Hybrid Model of RBM and SVM with equal weights of 0.5 have been trained on the dataset which resulted in the RMSE error of 0.997 The RMSE's of well trained algorithms are shown in table 7.1. Collaborative filtering does not andle data sparse problems very well when new items are introduced. To solve this problem, In future we can use the logistic regression method as a classifier to predict the user's music preferences to recommend songs.

Model	BaselineOnly	SVD	RBM	Hybrid(SVD+RBM)
RMSE	0.98	0.99	0.998	0.997

Table 7.1: RMSE of algorithms

Logistic regression is a linear model that does not handle complex non-linear data features.So one can create a hybrid LX recommendation algorithm by integrating logistic regression and eXtreme Gradient boosting(xgboost).

Bibliography

[1] Liu, N.H., Lai, S.W., Chen, C.Y. and Hsieh, S.J., 2009. Adaptive music recommendation based on user behavior in time slot. International Journal of Computer Science and Network Security, 9(2), pp.219-227.

[2] Liang, D., Zhan, M. and Ellis, D.P., 2015, October. Content-Aware Collaborative Music Recommendation Using Pre-trained Neural Networks. In ISMIR (pp. 295-301).

[3] Chen, H.C. and Chen, A.L., 2001, October. A music recommendation system based on music data grouping and user interests. In Proceedings of the tenth international conference on Information and knowledge management (pp. 231-238).

[4] Park, H.S., Yoo, J.O. and Cho, S.B., 2006, September. A context-aware music recommendation system using fuzzy bayesian networks with utility theory. In International conference on Fuzzy systems and knowledge discovery (pp. 970-979). Springer, Berlin, Heidelberg.

[5] Kuo, F.F. and Shan, M.K., 2002, December. A personalized music filtering system based on melody style classification. In 2002 IEEE International

Conference on Data Mining, 2002. Proceedings. (pp. 649-652). IEEE.

[6] Kaji, K., Hirata, K. and Nagao, K., 2005, November. A music recommendation system based on annotations about listeners' preferences and situations. In First International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution (AXMEDIS'05) (pp. 4-pp). IEEE.

[7] Song, Y., Dixon, S. and Pearce, M., 2012, June. A survey of music recommendation systems and future perspectives. In 9th International Symposium on Computer Music Modeling and Retrieval (Vol. 4, pp. 395-410).

[8] Dias, R. and Fonseca, M.J., 2013, November. Improving music recommendation in session-based collaborative filtering by using temporal context. In 2013 IEEE 25th international conference on tools with artificial intelligence (pp. 783-788). IEEE.

[9] Thorat, P.B., Goudar, R.M. and Barve, S., 2015. Survey on collaborative filtering, content-based filtering and hybrid recommendation system. International Journal of Computer Applications, 110(4), pp.31-36.

[10] Domingues, M.A., Gouyon, F., Jorge, A.M., Leal, J.P., Vinagre, J., Lemos, L. and Sordo, M., 2013. Combining usage and content in an online recommendation system for music in the long tail. International Journal of Multimedia Information Retrieval, 2(1), pp.3-13.