SABIR AKHTAR

SHIVAM SARIN

SIDDHANT CHAUHAN

last.fm

Recommendation Tools

PROJECT REPORT

`

Preprocessing of Data:

- The pictureURL and URL columns were removed from the Artists data. We keep only the name and ID of the users.
- We ensured that in both the databases, the artists and users should be the same.

Users	1891
Artists	12127
Tags	9718

- We merge the Artists names with the dataset to make the data more intuitive.
- We do a boxplot and observe that the weights in userArtists has many outliers.
- We need to make the data discrete. So, we decide to take deciles in percentile measure of 1 to 10, 1 being the least listened and 10 being the most listened.
- After that, we change the column names to make it more intuitive. Each decile has now equal number of songs listened. So, there is no class imbalance.
- We merge the tags data with the artist information to create a matrix for Content-Based recommendation system. Artist names are also merged to make the data more intuitive.
- Next, we find the frequency of how many times the user has tagged an artist in a particular year and month.
- We grouped the tags based on frequency of tags Top 50, Top 50 to 100, Mid 100 to 500, Mid 500 to 1000, Bottom 1000 to 2500 and Bottom which is named as tag_type

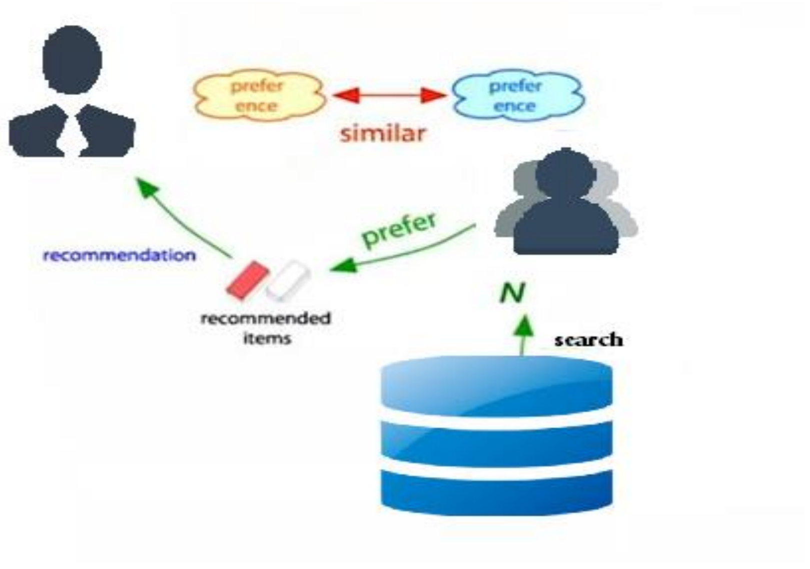
Tag Type	**Upper Bound**	**Lower Bound**
Top 50	7459	700
Top 50-100	682	315
Mid 100-50	312	39
Mid 500-1000	38	17
Bottom 1000-2500	16	4
Bottom	3	1

`

- Finally, we try to find that for each artist, the number of times he has been categorized in a tag_type.

For this project, we use **User based Collaborative Filtering** and **Item Based Collaborative Filtering**.

User based Collaborative Filtering



The basic idea behind collaborative filtering is very simple.

- We suppose that we have a user X to whom we want to make recommendations.

- So, at first, we try to find a group of other users whose likes and dislikes are similar to user X. For e.g. While recommending movies, this group of users you know, like the same movies that X likes *and* dislike the same movies that X dislikes. We call this set of users the neighborhood of user X.

- Next, we find other movies that are liked by a lot of users in the set N and recommend those items to the user X.

`

- So that's the basic idea behind collaborative filtering the key trick is to find the set of users that are similar to user X the neighborhood of user X and to do that we need to define a notion of similarity between users.

Instead of using the cosine measure here, we decided to take the Pearson correlation approach:

Mathematically, Pearson's connection coefficient is the covariance of the two variables isolated by the product of their standard deviations.

$$sim_{(x,y)} = \frac{\sum_{i \in I_{xy}} (r_{x,i} - \overline{r_x})\,(r_{y,i} - \overline{r_y})}{\sqrt{\sum_{i \in I_{xy}} (r_{x,i} - \overline{r_x})^2 \,\sum_{i \in I_{xy}} (r_{y,i} - \overline{r_y})^2}}$$

Where x, y: users

$r_{x,y}$: rating of user for item

I_{xy}: set of items, rated both by x and y

i = item in the set I_{xy}

$r_{x,i}$ = rating of user x for item i

$r_{y,i}$ = rating of user y for item i

$\overline{r_x}$ = average rating of user x

$\overline{r_y}$ = average rating of user y

The possible similarity values lie between 0 and 1.

After getting the similarity matrix, we use it to find 10 nearest neighbors. Rating of user x for item I or the prediction value is denoted by,

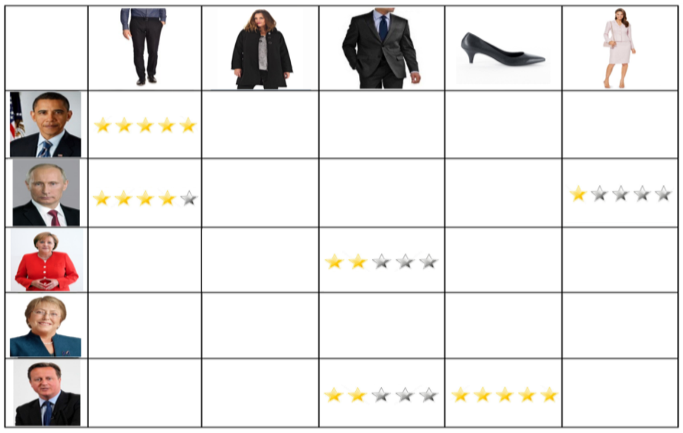
$$pred_{x,i} = \overline{r_x} + \frac{\sum_{n \in N} sim(x,n) * (r_{n,i} - \overline{r_i})}{\sum_{n \in N} sim(x,n)}$$

`

We chose 10 nearest neighbors as a parameter after we performed a cross-validation using a different range of values like 5,10,15 and 20. Resulting to which, 10 nearest neighbors was giving a better performance.

Item based Collaborative Filtering

Instead of looking for other people like the user and recommending stuff they liked, here we look at the things the user liked and recommend stuff that's similar to those things. We call this Item based Collaborative Filtering.



Similarity between items are obtained by looking at who are the users, who have rated how, how the users have rated for the item there are two items which most which users have rated in a similar way then those items are similar, this is an alternative to content-based similarity of items. We can use this matrix, we can compare the column vectors to find out the pair wise item similarity and we can use it for the recommendation in this way for a user we look at those items the user has recommended user has liked and find out items which are similar to these items.

The formula used to calculate the Pearson-Correlation similarity between different items is given as follows:

$$sim_{(i,j)} = \frac{\sum_{u \in U_{ij}} (r_{u,i} - \bar{r}_i)\,(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{i \in U_{ij}} (r_{u,i} - \bar{r}_i)^2 \sum_{u \in U_{ij}} (r_{u,i} - \bar{r}_j)^2}}$$

where,

i and j are two ITEMS

U_{ij} = the set of users rated both items i and j

`

u = item in the set U

$\mathrm{R}_{\mathrm{u,i}}$ = rating of user u for item i

$R_{\mathrm{u},j}$ = rating of user u for item j

$\overline{R_{\imath}}$ = average rating of item i

$\overline{R_{J}}$ = average rating of item j

After getting the similarity matrix, we use it to find 10 nearest neighbors. Rating of user x for item i or the prediction value is denoted by,

$$pred_{x,i} = \frac{\sum_{n \in N} sim(\mathrm{i}, n) * (r_{\mathrm{x},n})}{\sum_{n \in N} sim(\mathrm{i}, n)}$$

We chose 10 nearest neighbors as a parameter after we performed a cross-validation using a different range of values like 5,10,15 and 20. Resulting to which, 10 nearest neighbors was giving a better performance.

Clustering Based Recommendation System

Cluster-based recommendation is best idea of as a variation on user-based recommendation. Here, the items are prescribed to clusters/groups of likely clients. This involves a step, in which all users are initially divided into clusters. Each cluster is provided with a recommendation, with the motive that the recommendations are of maximum interest to the biggest number of users. The advantage of this methodology is that everything is pre-processed and hence, the run time is quick.

Information regarding ratings and interactions can be portrayed as a matrix or set of matrices, keeping the dimensions as users and items. Considering that the two matrices are almost same, we subtract the second from the first by changing existing ratings with 1 and missing ratings by 0. The resultant matrix is a truth table where:

- 1 represents whether there is interaction between user and product

- 0 represents no interaction between user and product

`

We use K-Nearest neighbor to calculate the set of individuals for recommendations dependent on the rating or item.

Content-Based Recommender Systems

Content-Based Recommender Systems are conceived from utilizing the content of everything for recommendations and attempting to take care of the issues faced by Collaborative Filtering. The main idea of this type of recommendation system is to identify similarity in the characteristics of each artist like, tags, country of artist, etc.

We have created a matrix for each artist which gives us the information related to user-tags for an artist. As explained in the pre-processing steps the different features used are as follows:

1) Year of tags per Artist aggregated being categorized in VeryOld, Old, Recent and New.

2) Months of tags per Artist aggregated being categorized in January – December.

3) Tags Frequency has been categorized in Top 50, Top 50-100, Mid 100-500, Mid 500-1000, Bottom 1000-2500 and Bottom tags.

The Cosine similarity of this matrix is calculated using the below mentioned formula to identify the similarity between characteristics of items:

$$sim_{(i,j)} = \frac{\sum_{c \in C} i_c * j_c}{\sqrt{\sum_{c \in C} i_c} * \sqrt{\sum_{c \in C} j_c}}$$

Where:

i and j are two ITEMS

C = the set of characteristics (columns)

c = an element of the set of characteristics C

After getting the similarity matrix, we use it to find 10 nearest neighbors. Rating of user x for item i or the prediction value is denoted by,

`

$$pred_{x,i} = \frac{\sum_{n \in N} sim(\mathrm{i}, n) * (r_{\mathrm{x},n})}{\sum_{n \in N} sim(\mathrm{i}, n)}$$

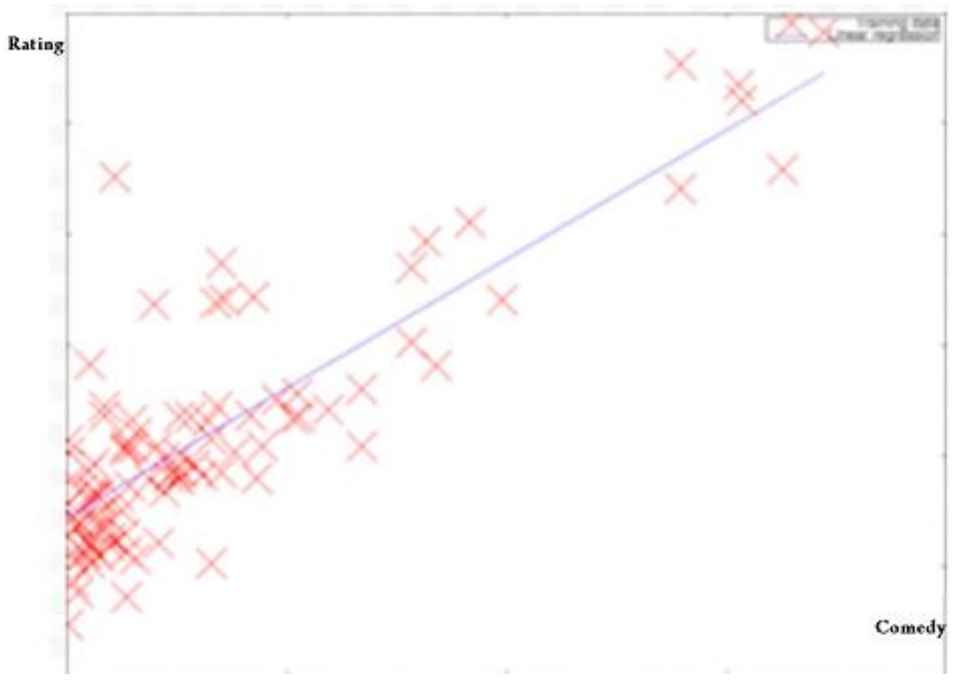
Hybrid - Average Mean of Predictions

In Hybrid recommendation technique we use predictions of different modelling techniques to pick up information from collaborative and content-based techniques to overcome the shortcomings of each technique.

In Average mean of predictions, we take the simple average of predictions of different algorithms for each respective cell <u,i> where u is the user and i is the item/artist. The final average is considered as the output of hybrid recommendation engine and evaluated against the actual test scores.

Hybrid - Linear Regression

Linear regression is a technique to identify the linear relationship between a scalar response and one or more explanatory variables. In the case of just one explanatory variable, the process is called simple linear regression and for more than one explanatory variable, the process is called multiple linear regression. For example, there is a linear relationship between the level of comedy and the movie rating from the user who saw the movie as shown in the diagram below:



`

As a part of Hybrid recommendation systems Linear regression is used in a similar way. We used multiple model predictions of Collaborative Filtering, Content-Based and Cluster-Based as Independent variable to train the Linear Regression model against Dependent variable Normalized Song Listening score (1-10). Using the Beta's obtained by model we can predict the score for the testing the data using the prediction scores of Collaborative Filtering, Content-Based and Cluster-Based and finally evaluate the hybrid predictions against the actual score.

Root Mean Square Error (RMSE) is the standard deviation of prediction errors. These prediction errors are an estimate of the distance of the data points from the regression line. RMSE gives us an idea how much the data is concentrated around the line of best fit.

Mean of Absolute Errors (MAE): It's the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight

$$\boxed{\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} | x_i - x |}$$

Where:

- n = the number of errors,
- Σ = summation symbol (which means "add them all up"),
- |xi – x| = the absolute errors.

The formula may look a little daunting, but the steps are easy:

1. Find all your absolute errors, xi – x.
2. Add them all up.
3. Divide by the number of errors. For example, if you had 10 measurements, divide by 10.

Precision: It is defined as the number of true positives divided by the number of true positives plus the number of false positives. False positives are situations where the model incorrectly labels as positive that are actually negative.

`

$$\text{Precision} = \frac{TP}{TP + FP}$$ where TP= True Positive, FP=False Positive

Recall: expresses the ability to find all relevant instances in a dataset, precision expresses the proportion of the data points our model says was relevant actually were relevant.

$$\text{Recall} = \frac{TP}{TP + FN}$$

Accuracy: It is how close a measured value is to the actual (true) value.

$$accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FlaseNegative}$$

F1 Score: The F1 score is the harmonic mean of precision and recall taking both metrics into account in the following equation:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

Results achieved:

	CB	UBCF	IBCF	CBCF	Hybrid (Mean)	Hybrid (Linear Regression)
RMSE	0.3722222	0.313518	0.0999373	0.071403	0.186	0.166
MAE	5.371259	4.563523	1.302681	0.682515	2.46	0.801
Recall	0.0001864	0.841263	0.090305	0.930338	0.163	0.9
Precision	1	0.886761	0.9270123	0.929589	0.996	0.93
Accuracy	0.3828922	0.833736	0.4269635	0.913537	0.5	0.894
F1	0.0003728	0.863413	0.1645777	0.929963	0.28	0.91

`

Advantages and Disadvantages of recommendation techniques used:

	Advantages	Disadvantages
User-Based Collaborative Filtering	1) As the number of users is less it is fast to process. 2) Achieving good results 3) Stable results for different training and testing data	1) Cold Start problem for new users 2) Scalability issue might occur if the number of users increase
Item-Based Collaborative Filtering	1) The items tend to be of a more permanent nature than people	1) As the number of artists is too big it takes long to run the process. 2) Cold-start Problem for new items
Cluster-Based Collaborative Filtering	1) As the number of users is less it is fast to process. 2) Achieving good results 3) Stable results for different training and testing data 4) Better and faster than regular user-based technique as centers are calculated for each cluster.	1) Cold Start problem for new users 2) Scalability issue might occur if the number of users increase
Content-Based	1) Comparison between items possible 2) New items can be associated based on other items characteristics	1) Heavily dependent on information related to the product. 2) Overspecialization 3) Difficult to collect accurate product information
Hybrid	1) It overcomes the shortcoming of content and collaborative. 2) Overcomes cold-start problem 3) Hybridization designs monolithic design combining different features parallel features, weighting/voting, pipelined, invocation of different systems.	1) It might pull down the accuracy of an excellent performing algorithm if combined with a moderately performing model.

Conclusion:

Based on the advantages and evaluations achieve we come to a conclusion that Hybrid-Linear Regression and Cluster-Based collaborative Filtering techniques are performing better for the Last.FM dataset. Henceforth, we chose Cluster-Based recommendation system as our best performing algorithm because it gave better results in most of the evaluation metrics.