**AMSTERDAM AIRBNB**

**RECOMMENDATION SYSTEM**

**MRITUNJYA KUMAR**

Table of Contents

[**Abstract** 3](#_Toc22250105)

[**INTRODUCTION** 3](#_Toc22250106)

[**Recommender Systems and Intended Use** 4](#_Toc22250107)

[**Materials and Methods** 5](#_Toc22250108)

[**Different Types of Similarity Methods: -** 9](#_Toc22250109)

[**METHODOLOGY** 12](#_Toc22250110)

[**COLLABORATIVE FILTERING** 12](#_Toc22250111)

[**Item-Item Collaborative Filtering** 14](#_Toc22250112)

[**CONTENT BASED FILTERING** 17](#_Toc22250113)

[**MODEL BASED RECOMMENDATION SYSTEM** 20](#_Toc22250114)

[**HYBRID BASED RECOMMENDATION SYSTEM** 23](#_Toc22250115)

[**CHALLENGES** 24](#_Toc22250116)

# **Abstract**

We have developed a Data driven solution for the **AMSTERDAM** **AIRBNB** based on the various techniques in the field of Recommender Systems. The main methods which we used for it are mainly:

* Collaborative based Recommendation
* Content Based Recommendation
* Model Based Recommendation
* Hybrid based Recommendation

All the above algorithms are aimed to predict the hotel recommendation for the users on the basis of their likings from their previous experience of the stay with AIRBNB as well as their Hotels similarities on the basis of their features.

The collaborative filtering uses the approach of the similarity between the users and recommending the hotels on the basis of their likings, as well as the similar type of ratings between the hotels.

The content-based approach, uses the text mining approach of their features like amenities and on the basis of that, we predict and recommend the hotels to the users.

In Model-based algorithm tries to compress huge database into a model and performs recommendation task by applying reference mechanism into this model. Model-based CF can response user’s request instantly.

In the hybrid approach, we combined Collaborative and Model Based technique using SVD (Singular Value Decomposition) and ALS (Alternating Least Square), and provide the recommendations to the users.

# **INTRODUCTION**

Recommendation systems are basically used to suggest the items based on the user’s past history and preferences. These systems are pretty useful in the various fields like movies, hotels, news, ecommerce, music, etc. An application of the Recommendation systems to suggest the new hotels and stays to the users on the user’s similar preferences thus promise him to get the best experience and to accelerate his search and the increase relevance.

There are multiple recommendation systems that uses either the personal preferences of the user (Content based approach) or to find out the resemblance between the news user’s preference and the previous user’s preference (Collaborative approach).

We all know there is a stiff competition among the various Hotels recommendation systems and to include the finest level of granularity among the various features and to recommend the best possible solutions according to the likings and dis likings.

The entire data consists of more than **20,678 records** which include the various hotels and their locations, the amenities that the hotels provide. As we know, when the data comes to the fine level, we need to include all the features to create the recommendation system to the users so that their requirements needs to be addressed. Please find the dataset below for your reference.



In the Recommendation engine, we present the four recommendation engines mainly Collaborative based, Content based, Model Based Technique and Hybrid based recommendation systems.

We tuned and tested the algorithms for the various users according to their preferences. First we calculated the predictions using the various similarity methods further including mainly Cosine, Euclidean, Pearson and Jaccard and look for the Mean absolute error for the different methods, and give the suggestions on that basis.

# **Recommender Systems and Intended Use**

Recommender systems technologies are not new and they dated back in the late nineties and are intended to make personalised suggestions based on the previous recorded data for the user’s preferences. These days recommender systems are accepted and widespread technology by various industries. Recommendation systems have the capability to cope up with the stated challenges with the data to be processed and are additionally able to permit insight into the decision-making process by making the outcomes interpretable for AMSTERDAM AIRBNB Management.

The basic classification of Recommendation algorithms differentiates between the collaborative based, Content based, Model Based and hybrid approaches. All the approaches convert the estimation of the user’s choice for Hotels (Items) into recommendations using explicit and implicit ratings as preferences but here we are considering explicit ratings. While the content-based approach attaches preferences to item attributes like Hotels features, the collaborative filtering method and model based using SVD and ALS considers the ratings of the other users to make personalised recommendations.

The underlying algorithm for rating prediction and recommendation computation are based on similarity metrices which have the capability to process sparse and inhomogeneous data vectors.

# **Materials and Methods**

**Data: Amsterdam Airbnb Dataset**

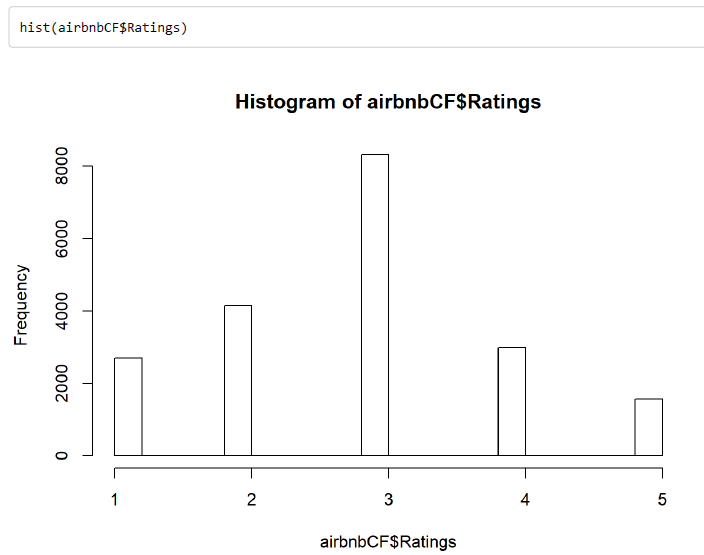
This is the first and the most crucial step for building the recommendation engine. The datasets which we have used for the recommendation systems are mainly:

* Amsterdam Airbnb Dataset with 20 fields

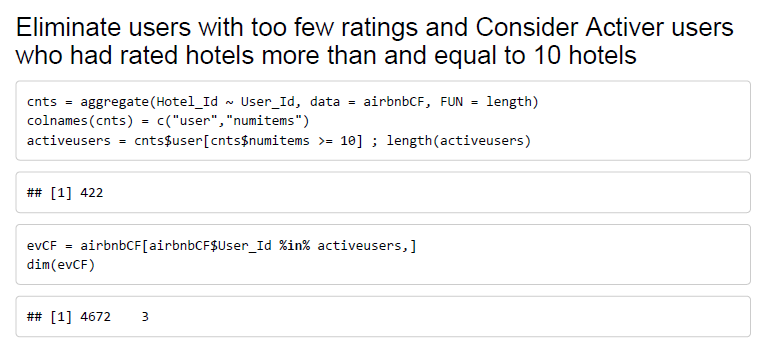
The amount of data tells us how good the recommendation system can we get. After collecting the data, we need to sort out and filter the data to extract the relevant information required to make the recommendation systems. Below are the steps which we have done before proceeding with actual recommendation algorithms.

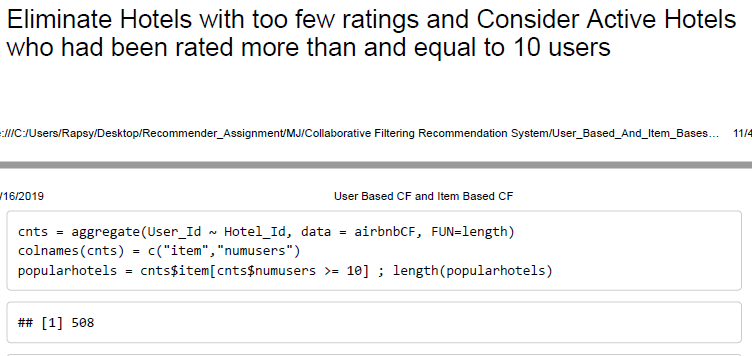
1. Cleaning up and mining of Dataset
2. Finding the Active Users and Active Hotels
3. Splitting the dataset into train and test
4. Building the model on Train Dataset
5. Validating the model using Test Dataset
6. Checking the MAE and Accuracy of the model
7. Then tune the model on various parameters.

* **Cleaning up and mining of Dataset**
* Data type transformation like (Integer to character, character to factor and so on)
* Missing value Imputation
* For mining, Stop word removal
* White space removal
* Stemming
* Lemmatization
* Punctuation removal
* Number removal
* User-defined stop word removal etc

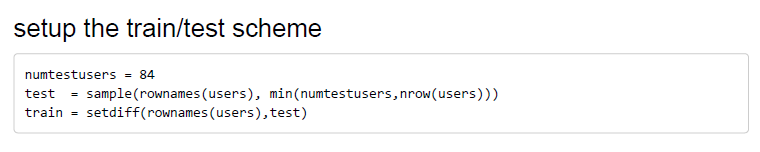


* **Finding the Active Users and Active Hotels**
* Consider Active User who had rated more than 10 hotels
* Consider Active Hotel Who got more than 10 ratings





* **Splitting the dataset into train and test**
* Split the dataset into Train and Test like 80% in train and 20% in Test dataset
* This splitting should be randomly with equal distribution of ratings value then fix these datasets

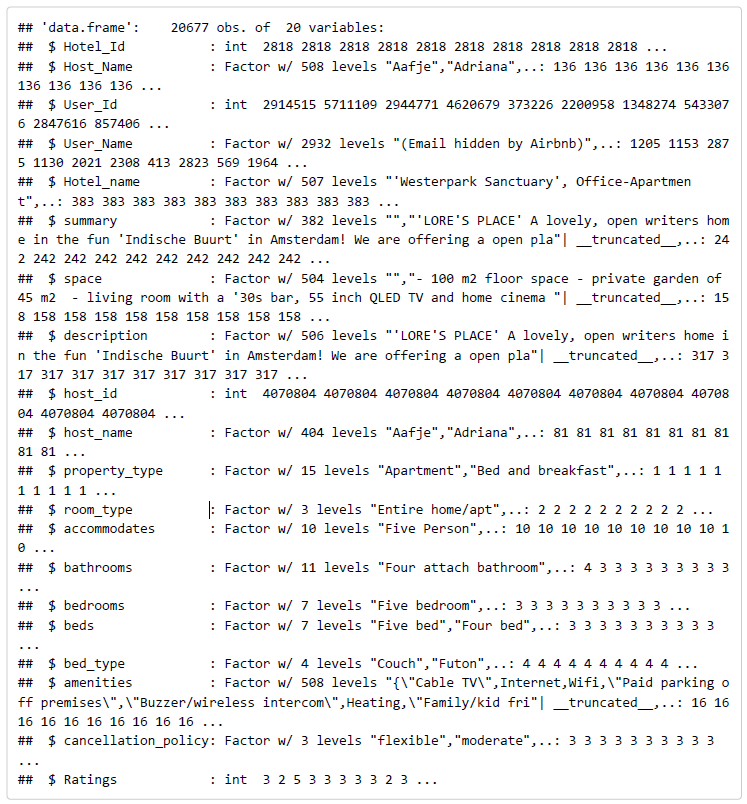


* **Building the model on Train Dataset**
* User-Based and Item-Based model using different Similarity Methods
* Content-Based using different similarity methods
* Model-Based using SVD and ALS
* Hybrid using Collaborative and Model-Based
* **Validating the model using Test Dataset**
* Predict the Ratings using build model on test dataset
* **Checking the MAE and Accuracy of the model**
* Calculate the mean absolute error and accuracy using test dataset observation
* **Then tune the model on various parameters.**
* Do some transformation on the Rating of the users like normalization, Z-score, etc
* Find the best similarity method which is giving less MAE

Different type of Similarity methods which we are using for building models: -

1. Cosine Similarity Method
2. Pearson Similarity Method
3. Euclidean Similarity Method
4. Jaccard Similarity Method

The above methods have been used in all the models and has been explained separately below in details with formulas.

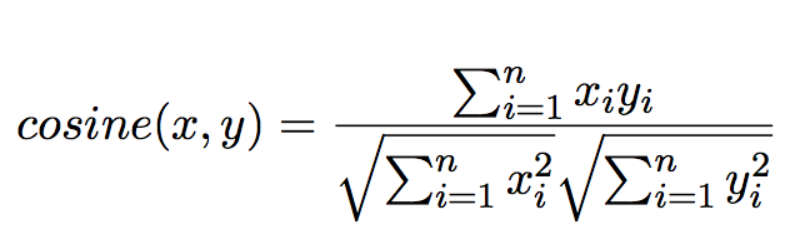
**Below are the Fields and its structure**

**Airbnb Fields Name and its top 4 records**

# **Different Types of Similarity Methods: -**

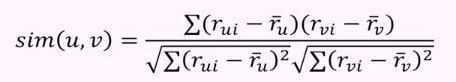
1. **Cosine Similarity Method: -**

This method is also most commonly used method in collaborative filtering in recommender systems. Cosine similarity finds how two vectors are related to each other using measuring cosine angle between these vectors. The major drawback with cosine similarity is that it considers null preferences as negative preference.

To compute similarity between the user and item, we simply take the cosine similarity between the user vector and the item vector. This gives us user-item similarity.

1. **Pearson Similarity Method: -**

It tells us how much the two items are correlated. Higher the value of correlation, more will be the similarity between the . Pearson’s correlation can be calculated by using the formula:

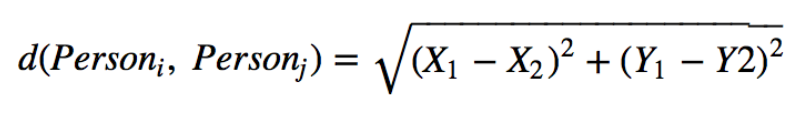


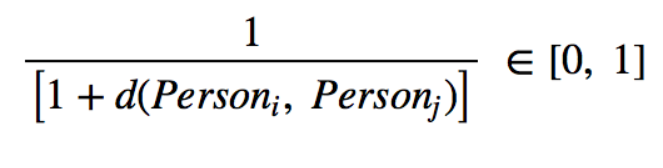
1. **Euclidean Similarity Method: -**

**Euclidean Similarity** between two people based on their common tastes. Those tastes are retrieved from our main data structure stored in our *data* variable.

**Euclidean Distance** – This one basically tells us the how much the similar items lie in the close proximity to each other if plotted in the N dimensional space. Here we calculate the distance between items and based on that distance, we recommend the items to the user.

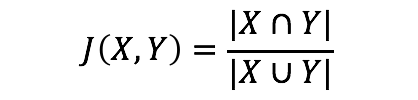
The formula used to calculate the Euclidean distance is given by:





1. **Jaccard Similarity Method: -**

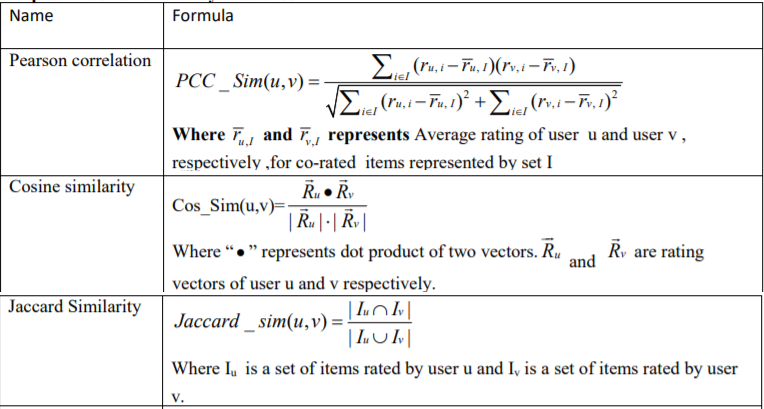
Jaccard similarity takes number of preferences common between two users into account. This does not consider the absolute ratings rather it considers number of items rated. Two users will be more similar, when two users have more common rated items. Jaccard produces limited number of values which makes the task of user distinction difficult. Also known as intersection over union, the formula is as follows:



This is used foritem-item similarity.We compare item vectors with each other and return the items that are most similar.

Jaccard similarity is useful only when the vectors contain binary values. If they have rankings or ratings that can take on multiple values, Jaccard similarity is not applicable.

**All Similarities Equations: -**



# **METHODOLOGY**

The algorithms described in the following aim at recommending the potentially effective recommendation system to the users are below:

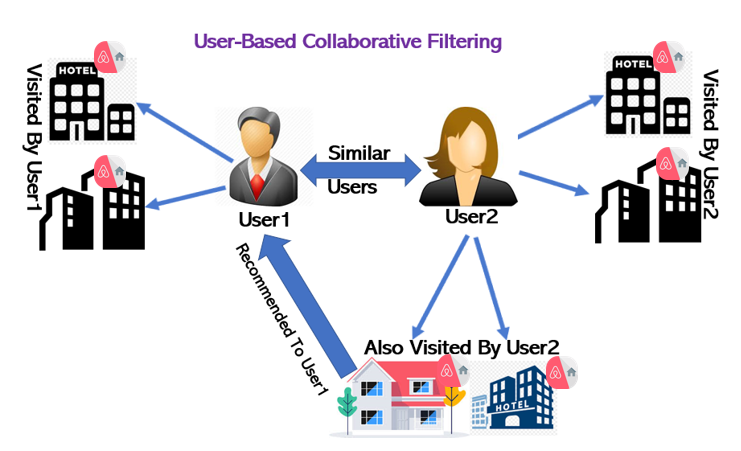
# **COLLABORATIVE FILTERING**

Collaborative filtering uses *“User Behaviour” as* well as Item similarityfor recommending items. There are different types of collaborative filtering techniques:

Collaborative Filtering is a technique which is widely used in recommendation systems and is rapidly advancing research area. The two most commonly used methods are memory-based and model-based. In this post, we will only focus on (User-Based Collaborative Filtering) UB-CF which is a memory-based method

* **User- User collaborative filtering**
* **Item-Item Collaborative filtering**

**User- User Collaborative Filtering**



This algorithm first finds the similarity score between users. Based on the similarity score, it then picks out the most similar users and recommended products which these similar users have liked or bought previously.  this algorithm finds the similarity between each user based on the ratings they have previously given to different movies. The prediction of an item for a user u is calculated by computing the weighted sum of the user ratings given by other users to an item i.

The prediction Pu,i is given by:



* Pu,i is the prediction of an item
* Rv,i is the rating given by a user v to a movie i
* Su,v is the similarity between users

Now, we have the ratings for users in profile vector and based on that we have to predict the ratings for other users. We follow the below steps to do so:

* For predictions we need the similarity between the user u and v. We can make use of Pearson correlation.
* First, we find the items rated by both the users and based on the ratings, correlation between the users is calculated.
* The predictions can be calculated using the similarity values. This algorithm, first of all calculates the similarity between each user and then based on each similarity calculates the predictions. Users having higher correlation will tend to be similar.
* Based on these prediction values, recommendations are made.

**RESULTS**

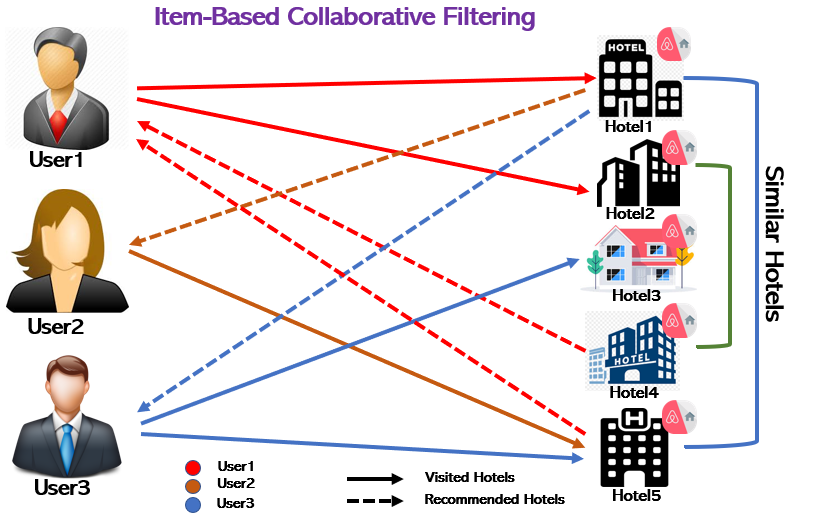
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User Based Collaborative Filtering** | | | | |
|  | **Cosine** | **Euclidean** | **Pearson** | **Jaccard** |
| **Tests** | 375 | 588 | 18 | 588 |
| **MAE** | 1.16354 | 1.119629 | 1.113358 | 1.11389 |
| **accuracy** | 99.5 | 99.8 | 100 | 99.8 |

**Recommendation**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **User Based Recommendation for a User** | | | | | | | |
| **User\_ID** | **User\_Name** | **Similarity Type** | **Top1\_Hotel** | **Top2\_Hotel** | **Top3\_Hotel** | **Top4\_Hotel** | **Top5\_Hotel** |
| 57920 | Ginny | **Cosine** | 284319 | 735973 | 516482 | 361342 | 189754 |
|  |  | **Euclidean** | 284319 | 632434 | 189754 | 621108 | 311694 |
|  |  | **Jaccard** | 284319 | 632434 | 189754 | 621108 | 311694 |
|  |  | **Pearson** | 570753 | 791690 | 226121 | 236180 | 101618 |

**   **

# **Item-Item Collaborative Filtering**



This algorithm is used to compute the similarity between each pair of items. This filtering method works similar like user-user collaborative filtering with just a small change that instead of taking the weighted sum of ratings of “user-neighbours”, we take the weighted sum of ratings of “item-neighbours”. The prediction is given by:



Then we will find the similarity between items.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item Based Collaborative Filtering** | | | | |
|  | **Cosine** | **Euclidean** | **Pearson** | **Jaccard** |
| **Tests** | 570 | 587 | 77 | 587 |
| **MAE** | 1.052709 | 1.089706 | 1.098366 | 1.085475 |

**Results: -**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Item Based Recommendation for a User** | | | | | | | |
| **User\_ID** | **User\_Name** | **Similarity Type** | **Top1\_Hotel** | **Top2\_Hotel** | **Top3\_Hotel** | **Top4\_Hotel** | **Top5\_Hotel** |
| 57920 | Ginny | **Cosine** | 171631 | 182839 | 543930 | 675673 | 682461 |
|  |  | **Euclidean** | 64769 | 171631 | 182839 | 221922 | 528594 |
|  |  | **Jaccard** | 64769 | 171631 | 182839 | 221922 | 528594 |
|  |  | **Pearson** | 294468 | 304189 | 320689 | 327285 | 793845 |

**Recommendations: -**

All recommended Excels with different similarity methods: -

**START**

Load the relevant libraries

Load the Airbnb dataset and check the structure and summary of data.

. Clean the dataset i.e. imputation of missing values, labelling, datatype transformation, feature transformation etc.

. Create a new dataset with appropriate columns for collaborative filtering.

Find active users and active hotels to sort out the data sparsity, then remove the duplicate records.

Split the Airbnb dataset into train and test. ie. 80% train and 20% test data

Convert the data from long format to wide format, then do the normalisation of data and convert into matrix for collaborative processing.

END

Predict top 5 list of the hotels similar to the hotels list rated by the user on the basis of similarity score.

Find list of rated hotels for the specific user from the dataset.

Predict the similar users using the rated hotels by a specific user on the basis of similarity score.

There is different similarity score to boost your prediction. Cosine, Jaccard, Euclidean and Pearson

Do the testing part. Find the accuracy using test dataset. Now predict the users of test datasets and find the accuracy and MEA

ITEM BASED

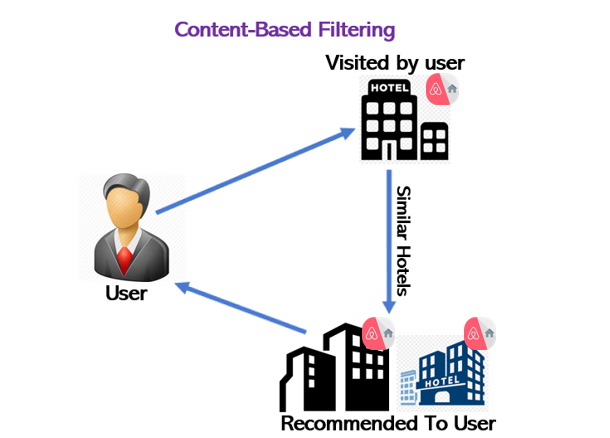
USER BASED

Find the similarity score between item -item using similarity score

Split the Airbnb dataset into train and test. ie. 80% train and 20% test data

# **CONTENT BASED FILTERING**

This algorithm recommends hotels which are similar to the one’s user who has like in the past.



This type of filter does not involve other users if not ourselves. Based on what we like, the algorithm will simply pick items with similar content to recommend us.

In this case there will be less diversity in the recommendations, but this will work either the user rates things or not. If we compare this to the example above, maybe user in the diagram potentially likes Hotels with all facilities but he/she will never know, unless he/she decides to give it a try autonomously, because this filter will only keep recommending dystopian Hotels or similar. Of course, there are many categories we can calculate the similarity on: in the case of Hotels, we can decide to build our own recommender system based on Hotel features like Amenities such as: -

* Internet and Wifi and Buzzer/wireless intercom
* Paid parking off premises
* Heating and Washer
* Smoke detector and Carbon monoxide detector
* First aid kit, Safety card and Fire extinguisher
* Shampoo
* Lock on bedroom door
* 24-hour check-in
* Hangers, Hair dryer, Iron and Laptop friendly workspace
* Private entrance
* Hot water
* Bed linens, Extra pillows and blankets
* Single level home

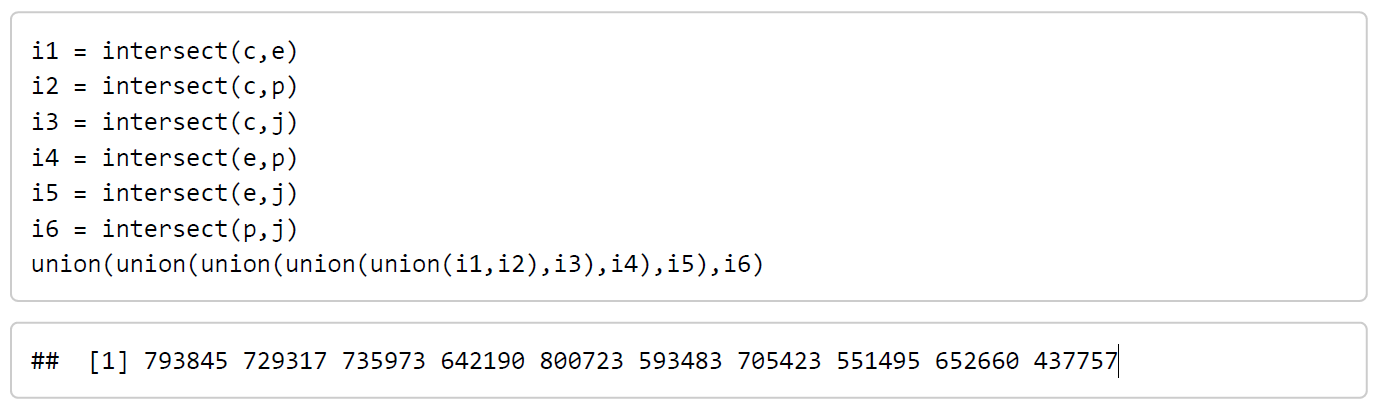
Please have a look into the dataset of Airbnb for more details (attached in the above page)

**Results: -**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Hotel\_Id** | **Hotel\_Name** | **Method** | **Top1\_Similar\_Hotel** | **Top2\_Similar\_Hotel** | **Top3\_Similar\_Hotel** |
| **2818** | **Quiet Garden View Room & Super Fast WiFi** | **Correlation** | 794322 | 705423 | 652660 |
| **Cosine** | 793845 | 729317 | 735973 |
| **Euclidean** | 593483 | 705423 | 481664 |
| **Jaccard** | 793845 | 437757 | 800723 |

|  |  |
| --- | --- |
| **Top4\_Similar\_Hotel** | **Top5\_Similar\_Hotel** |
| 593483 | 551495 |
| 800723 | 642190 |
| 793845 | 729317 |
| 356424 | 729317 |

This recommendation is the union and Intersection between a pair of similarity to boost the recommendation: -



**START**

Load the relevant libraries

Load the Airbnb dataset and check the structure and summary of data.

Create a new Airbnb\_cintent dataset with all the hotel features (Hotel Id, Hotel Name, Summary, space, description, room type, property type, amenities, etc

Create a long text columns by merging all the Hotel Features

Create corpus using text columns and perform text mining on the corpus like (Lower case conversion, Tokenization, Number removal, Stop-word removal

(Along with user defined stop-words), Punctuation removals, Stemming, Lemmatization, Whitespace removal)

Normalized the corpus and create DTM

Then Create Weighted Term frequency-Inverse Document frequency(DTM\_TI) using DTM

Convert DTM\_TI into matrix (MAT\_TI)

Predict top 10 similar hotels using MAT\_TI and similarity matrix

12.1 Output1 - using Cosine Similarity

12.2 Output2 - using Pearson Similarity

12.3 Output3 - using jaccard Similarity

12.4 Output4 - using Euclidean Similarity

Find the common hotels from below:-

13.1 Intersection1 = (Output1 and Output2)

13.2 Intersection2 = (Output1 and Output3)

13.3 Intersection3 = (Output1 and Output4)

13.4 Intersection4 = (Output2 and Output3)

13.5 Intersection5 = (Output2 and Output4)

13.6 Intersection6 = (Output3 and Output4)

Now finally take the union of all the Intersections outputs (Intersection1, Intersection2, Intersection3, Intersection4, Intersection5, Intersection6)

END

**Efficiency and Accuracy Test for Content-Based Recommendation: -**

Content Based Recommended hotels can be verified by taking the feedback from the recommended users on weekly basis to create the confusion matrix and tune the model if required.

Content-Based Recommended hotels can also be verified manually by checking the test dataset user features along with recommended hotel feature and then we should create the confusion matrix and decide whether we need to tune the model more or not.

# **MODEL BASED RECOMMENDATION SYSTEM**

Model based recommendation is used mainly when rating matrix becomes so huge that computational resources are consumed a lot and the system performance goes down.

Through this approach dimensionality reduction methods are used as a complementary technique to improve the accuracy and robustness.

Methods like SVD, ALS known as Latent factor models compresses the user-item matrix into a low dimensional representation in terms of latent factors. Its main advantage is, instead of dealing with high dimensional matrix which have lot of missing values, we will deal with a much smaller matrix in lower dimensional space.

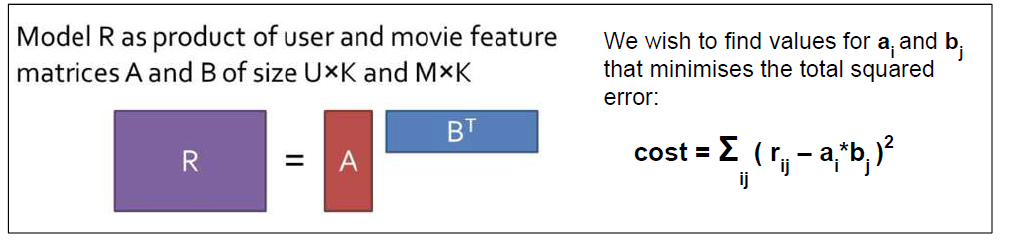
A reduced presentation is used for either user-based or item-based algorithms.

The main advantages of Model based are:

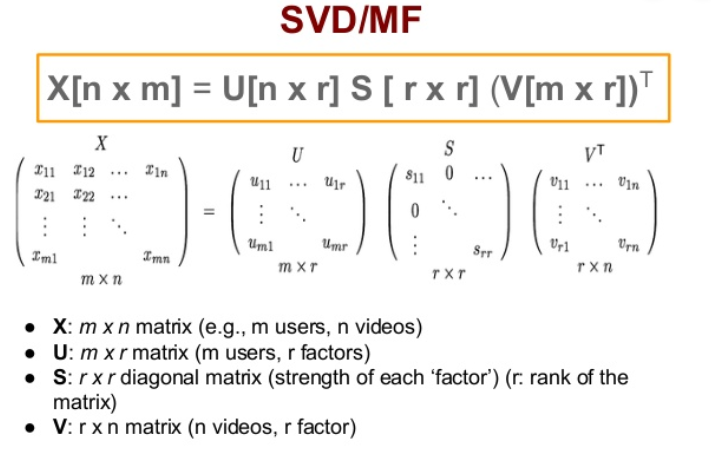
**Scalability** – Model that we get from Model based algorithm are much smaller than actual dataset, and the resulting small model enough to be used efficiently.

**Prediction Time** – The prediction time from Model based is much smaller in comparison to other methods as it builds the model on the smaller dataset instead of the whole dataset.

**Avoidance of Overfitting** - We can avoid the overfitting of the model if the dataset on which we are building our model is representative of the whole data.

**ALS (ALTERNATING LEAST SQUARES)**

**SVD (SINGULAR VALUE DECOMPOSITION)**



|  |  |  |
| --- | --- | --- |
|  | **ALS** | **SVD** |
| **MAE** | 2.662412 | 2.57295 |

**Results: -**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **User-Id** | **User-Name** | **Method** | **Top1\_Hotel** | **Top2\_Hotel** | **Top3\_Hotel** | **Top4\_Hotel** | **Top5\_Hotel** |
| **57920** | **Ginny** | **ALS** | 445345 | 240191 | 546221 | 55709 | 276275 |
| **SVD** | 445345 | 240191 | 546221 | 55709 | 205759 |

**Recommendation for User: -**

All recommended Hotels for each user attached below with SVD and ALS

** **

**START**

compute the MAE for the predictions made for the test events

Find the Confusion Matrix by setting threshold rating as 2.5 or 3

compute a predicted rating for each (user,item) in test events, using the formula: prediction = sum( userfeatures (from u matrix) \* singularvalues (d matrix) \* hotel features (from v matrix) )

Load the relevant libraries

Load the Airbnb dataset and check the structure and summary of data.

Convert User\_iD and Hotel\_ID Columns as a factor to do ALS and SVD Modelling

Add user and item names to U and V so we can index them by user name and item name

Use soft Impute method to build the model by using type as "ALS" or "SVD" (ALS will be by default) to create factorised matrices and make predictions for all of the empty (user,item) pairs (the test pairs + those missing in orginal dataset)

Split the dataset into Train (80%) and Test (20%) and Factorize into U \* D \* V using 30 latent features

Perform data cleaning on the dataset mainly Data type transformation, Duplicate removals, Treatment of missing values, Feature Engineering, etc

After that create a dataset using only active users and active hotels active users who have rated more than 10 hotels and active hotels which are rated at least 10 times.

Create a new Airbnb\_ALS\_SVD dataset with required columns like User\_ID, Hotel\_ID and Ratings

**END**

# **HYBRID BASED RECOMMENDATION SYSTEM**

Model Based Techniques Using SVD

Hybrid Recommendation system is the combination of two different models such as Model-1 and Model-2. In our case, we are using the combination of Item-Based collaborative filtering and Model-Based Technique using SVD as it have less MAE.

Model-Based Techniques Using SVD

Item-Based Collaborative Filtering

Hybrid System (Avg on Score or result like ratings)

**Result of Hybrid System i.e. Recommendation of Hotels for each user: -**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User\_Id** | **User\_Name** | **Hybrid\_Top1\_Hotel** | **Hybrid\_Top2\_Hotel** | **Hybrid\_Top3\_Hotel** |
| 57920 | Ginny | 445345 | 240191 | 546221 |

|  |  |  |
| --- | --- | --- |
| **Hybrid\_Top4\_Hotel** | **Hybrid\_Top5\_Hotel** | **Hybrid\_Top6\_Hotel** |
| 107195 | 171631 | 182839 |

All recommended Hotels for each user attached below with IBCF and MB using SVD



# **CHALLENGES**

Working on any project where you have found the data and the proper domain is challenge in itself, especially when you want to make something unique and different (not the one which are easily available on Internet like movielens, jester and all). Below are the few challenges we have faced:

**PROPER DATA**

We have googled a lot, look into the various topics and then getting the proper dataset which have all the features required to build a recommendation system and on which we can apply all the techniques like Content based, Collaborative and hybrid is a daunting task.

Sometimes, we get the data which have very few records, and sometime we get very huge dataset which becomes difficult to access in system.

Finding the statement problem and proper data is one of the main challenges we have faced in the initial stage.

**CASTING**

After getting the data, we have to do a lot of clean and scrubbing of the data. While casting the data is also another hurdle that we have faced in the project. While doing the casting in R, the dataframe becomes huge and it goes out of memory. Then we looked at data, as there are lot of duplicate records are there, which makes the task tedious.

We have to look into the insights of data, after lot of work, we are finally able to cast the data as we need to eliminate where we get duplicate records. After that the hotels which have been reviewed very less, and the users who have reviewed very less hotels.

**SPARSITY**

After casting, when we look at the sparsity of the dataframe, the sparsity is vey high. It is difficult to distinguish the similar interests between customers because of sparsity due to the insufficient rating of the hotels and the customers. Sparsity has issues as it becomes difficult to create matrix. Dimensionality reduction technique is used to deal with the sparsity problem by generating the dense user-item interaction matrix that considers the relevant users and items for recommendation systems.

**STOPWORDS**

While creating the content-based approach, we need to look at the stop words. Initially, while doing the text analytics on the reviews for content based approach, there comes a lot of words which repeat many times, like ***can, will, best, use, find, miss, public, plan, close, just, like*** which has least significant to the word cloud, as they don’t define the features of the hotels.

We need to delete the least important stopwords, to come with the most meaningful and useful stopwords for the content-based approach.

**REVIEWS**

As we have lot of total of more than 20,678 records, and there are around 508 Active hotels and 422 unique active users (Definition of Active user and Active hotels are mentioned above under Materials and method section) which are unique, so when we try to combine(or group) the reviews of the hotel, our machine gets hanged, as the reviews written by users are lengthy and sometimes combining 100 different reviews of same hotels creates a very tedious task. So, we tried different methods to get to combine, but every time machine is not able to combine reviews for all the hotels. So we have excluded the review dataset(Review was in the different dataset where we need to join the both dataset using hotel\_id and User\_id)