Recommendation tools

Vinay RAJAGOPALAN

Aleksandr SHKURIN

Ahmad Omar Nakib

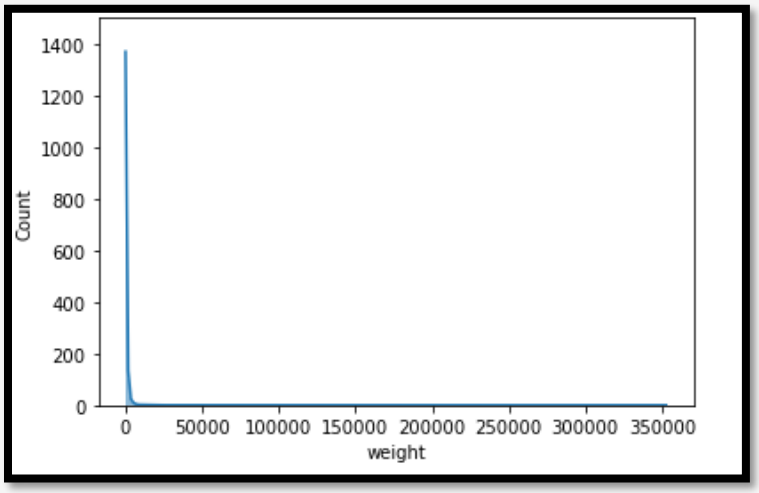
1. Intro & Data:

The “Artist” table contains artist id, artist name, artist URL, and artist picture URL

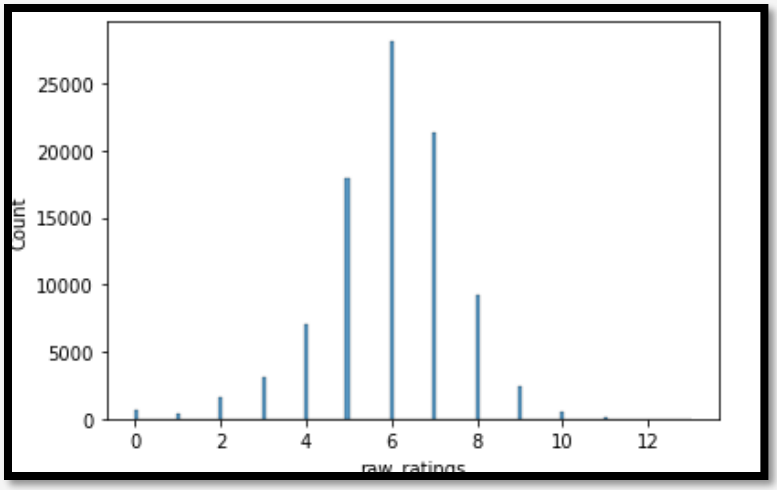
User_artist Table contains user ID, artist ID, and the weight.

Starting off with the User_artist table we decided to check the distribution by weight across the table and we used log transformation in order to scale it. We also brought the range down from 350000 to 12 so that we can mimic the usual rating systems in addition to reducing the skewness.

n.b: unscaled distribution



n.b: scaled distribution



Deeper look into the data we can see these summary statistics of the raw ratings:

```
count    92834.000000
mean         5.965595
std          1.571694
min          0.000000
25%          5.000000
50%          6.000000
75%          7.000000
max         13.000000
Name: raw_ratings, dtype: float64
```

We then proceeded by removing the 0 ratings as they might disrupt our results and we proceed to check the summary statistics to see the differences.

```
count    92198.000000
mean         6.006746
std          1.496687
min          1.000000
25%          5.000000
50%          6.000000
75%          7.000000
max         13.000000
Name: raw_ratings, dtype: float64
```

We finally take a look at the counts to assess the distribution:

```
6.0     28177
7.0     21355
5.0     17976
8.0      9172
4.0      7052
3.0      3162
9.0      2489
2.0      1593
10.0      587
1.0       419
11.0      165
12.0       41
13.0       10
Name: raw_ratings, dtype: int64
```

Tags table contains tag ID and Tag values fields only.

user_taggedartists table contains userID, artistID, tagID, day, month and year. In the user_taggedartists table we concatenated the date using the month day and year fields and merged into one field then we proceeded to drop the day the month and the year.

Tmp table contains merged results of user_taggedartists and tags. In this case Tmp contains userID, artistID, tagID, date and tagValue.

We created an aggregarted result of the tmp table where we merged all tags associated by artist into a list. We then made sure of the data by dropping any duplicated values in the list in each row of the dataframe called aggregate_1.

```
aggregate_1.head()
```

artistID	tagValue
1	[better than lady gaga, gothic, j-rock, japanese, jrock, visual kei, weeabo]
2	[ambient, dark, darkwave, electronic, german, gothic, gothic rock, industrial, seen live, true goth emo, vocal]
3	[black metal, norsk arysk metal, norwegian black metal, saxophones, true norwegian black metal, very kvlt]
4	[bazarov, gothic, gothic japanese, gothic metal, j-rock, japanese, metal, rock, visual kei]
5	[covers, darkwave, deathrock, gothic, gothic rock]

Aggregate_2 is a dataframe containing the frequency at which each user has tagged an artist. The fields initially contained artist ID, userID and the count (number of times user tagged the artist). We then proceeded our preprocessing by unstacking the group by (pivoting the table).

Aggregate_3 is a table containing the recency in which each user has tagged their respective artist. In order to do that we looked at the max value the date in which the user tagged the artist and by doing so subtract todays date from this max value thus obtaining the recency in days. We then proceeded our preprocessing by unstacking the group by (pivoting the table).

2. Recommendation System: Collaborative Filtering

In order to apply collaborative filtering, we first subset the data from user_artist table we take userID, artistID, raw_ratings columns only. We then continue by splitting our new subset into a training set and test set at a random state of 42 and a test size of 30% and we reset our index.

Subset size: 92198

Train split size: 64538

Test split size: 27660

We also removed ratings of 11 and 12 and kept it concise from 1 to 10 as a scale.

Moving into the models:

1- Item Based:

```
# item-based
options = {'name':'cosine', 'user_based':False}
ib = KNNBasic(k=15, min_k=5, sim_options=options, random_state=42)
ib.fit(df_train)
ib_pred = ib.test(df_test)
```

Item based recommendation system looks at each user rating by item and attempts on predicting the ratings based on cosine value and using k nearest neighbor in our case we took a value of K = 15 and a minimum neighbors of 5 at a random state of 42.

2- User Based:

```
# user-based
options = {'name':'cosine', 'user_based':True}
ub = KNNBasic(k=20, min_k=5, sim_options=options, random_state=42)
ub.fit(df_train)
ub_pred = ub.test(df_test)
```

User based recommendation system looks at each user attempts to find each artist by user similarities to predicting the on cosine value and using k nearest neighbor in our case we took a value of K = 20 and a minimum neighbors of 5 at a random state of 42.

3- Co Clustering:

```
# Co Clustering
clust = CoClustering(n_cltr_u=10, n_cltr_i=10, n_epochs=50, random_state=42)
clust.fit(df_train)
predclu = clust.test(df_test)
```

Co-clusterting recommends artist to a user in a specific cluster only using the rating statistics of the other users of that cluster. In our case we used number of clusters = 10 for both users and items and number of epochs = 50.

4- SVD:

```
# svd
mf = SVD(n_factors=20, biased=False, random_state=42)
mf.fit(df_train)
mf_pred = mf.test(df_test)
```

SVD is a probabilistic matrix factorization algorithm which predicts rating for the users using matrix multiplication technique in our case we used a number of factors 20 and without bias.

5- ALS:

```
# als
options = {"method": "als", "n_epochs": 20}
als = BaselineOnly(bsl_options=options)
als.fit(df_train)
als_pred = als.test(df_test)
```

ALS factorizes the user to item matrix A into the user-to-feature matrix U and the item-to-feature matrix M it tries to find optimal factor weights to minimize the least squares between predicted and actual ratings.

Results of collaborative filtering models:

	IB	UB	CLU	SVD	ALS
RMSE	1.155835	1.496085	1.114257	1.421003	0.940659
MAE	0.829390	1.128415	0.820945	1.004436	0.713441
Recall	0.955123	0.994916	0.925705	0.832660	0.975791
Precision	0.922934	0.867718	0.939723	0.942417	0.950985
F1	0.938753	0.926974	0.932662	0.884145	0.963228
NDCG@5	0.843581	0.856282	0.852969	0.859563	0.855251

Evaluation metric:

RMSE : ROOT MEAN SQUARE ERROR

MAE : MEAN ABSOLUTE ERROR

RECALL : The recall is intuitively the ability of the classifier to find all the positive samples

PRECISION : The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

F1 : F1-score combines the precision and recall of a classifier into a single metric

NDCG@5 : Sum the true scores ranked in the order induced by the predicted scores

RMSE, MAE: Since all of these ratings are error rates, we need to identify the lowest possible combination of rmse and mae in order to deduce the best algorithm tested.

Recall and precision: Recall is the absolute positive rate so we should look at maximizing it and precision should also be maximized.

Looking at our results we decided to go with RMSE in order to choose our collaborative filtering models and by checking the table we chose ALS & co-clustering.

3. Recommendation System: Content Based

Content-based filtering is one popular technique of recommendation or recommender systems. The content or attributes of the things you like are referred to as "content." Here, the system uses your features and likes to recommend you with things that you might like. It uses the information provided by you over the internet and the ones they can gather and then they curate recommendations according to that.

In our case we have created the content-based recommendation system based on the existing tags per artist id (aggregate_1) table, frequency at which each user has tagged the artist (aggregate_2) table and the recency by which each artist was tagged (aggregate_3) table.

Based on this we evaluated our models against the metrics defined above and we noticed that recency-based content-based model performed the best followed by frequency-based content-based system and lastly existing tags for the user. The RMSE as shown below were low for all content-based model as compared to most collaborative filtering model, but ALS still had the best RMSE.

	CB_REC	CB_FREQ	CB_TDIDF	IB	UB	CLU	SVD	ALS
RMSE	0.995778	1.002499	1.061201	1.155835	1.496085	1.114257	1.421003	0.940659
MAE	0.724015	0.728852	0.767399	0.829390	1.128415	0.820945	1.004436	0.713441
Recall	0.949873	0.950623	0.953873	0.955123	0.994916	0.925705	0.832660	0.975791
Precision	0.951657	0.950267	0.940703	0.922934	0.867718	0.939723	0.942417	0.950985
F1	0.950764	0.950445	0.947242	0.938753	0.926974	0.932662	0.884145	0.963228
NDCG@5	0.864225	0.866111	0.864106	0.843581	0.856282	0.852969	0.859563	0.855251

In addition, we also performed Principal Component Analysis (PCA) to reduce the dimension of the recency-based content-based system to see if we could improve the performance of the model. There were close to 1828 features, and we tried to pick the 40 components using PCA to see if that improves our performance.

Once PCA was applied and we had the 40 components we repeated the procedure of content-based recommendation and noticed that we did improve on our RMSE when compared to the recency based approach. We able to improve our RMSE for recency based content based model from 0.99577 to 0.969446.

4. Recommendation System: Hybrid

In Hybrid recommender systems combine two or more recommendation strategies in different ways to benefit from their complementary advantages.

To follow this idea, we decided to combine our best models from collaborative and content based which in this case would be the ALS and Recency Based Content Based approach as discussed above.

In order to accomplish this, we merged the predictions from ALS and Recency Based Content Based in a data frame. We then split the data into test and train and then applied linear regression and random forest models to the train and then evaluated the performance on the test dataset to obtain the evaluation of the model. We saw that our models were able to reduce the RMSE considerably using the Hybrid model and thus we the best model with lowest RMSE was the hybrid random forest model. The evaluation of all the three hybrid models along with the collaborative and content-based recommendation is demonstrated below. The lowest RMSE can be seen for HYBRID random forest of 0.8626 closely followed by HYBRID linear regression and content-based hybrid.

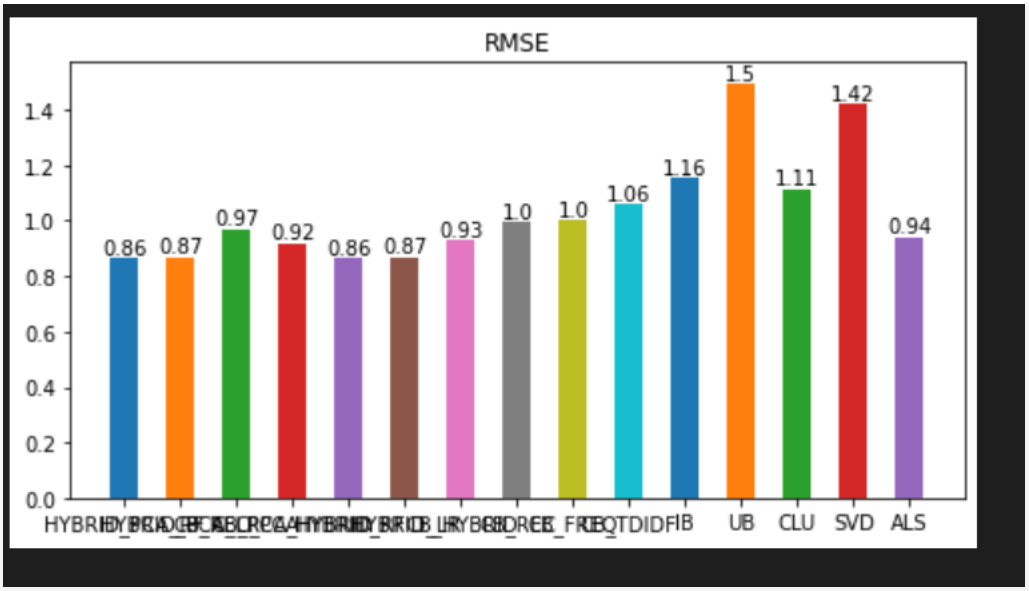
	HYBRID_RF	HYBRID_LR	CB_HYBRID	CB_REC	CB_FREQ	CB_TDIDF	IB	UB	CLU	SVD
RMSE	0.862606	0.868621	0.928663	0.995778	1.002499	1.061201	1.155835	1.496085	1.114257	1.421003
MAE	0.652072	0.651895	0.695740	0.724015	0.728852	0.767399	0.829390	1.128415	0.820945	1.004436
Recall	0.941623	0.934747	0.966082	0.949873	0.950623	0.953873	0.955123	0.994916	0.925705	0.832660
Precision	0.974556	0.976537	0.951492	0.951657	0.950267	0.940703	0.922934	0.867718	0.939723	0.942417
F1	0.957806	0.955185	0.958731	0.950764	0.950445	0.947242	0.938753	0.926974	0.932662	0.884145
NDCG@5	0.858320	0.855145	0.856330	0.864225	0.866111	0.864106	0.843581	0.856282	0.852969	0.859563

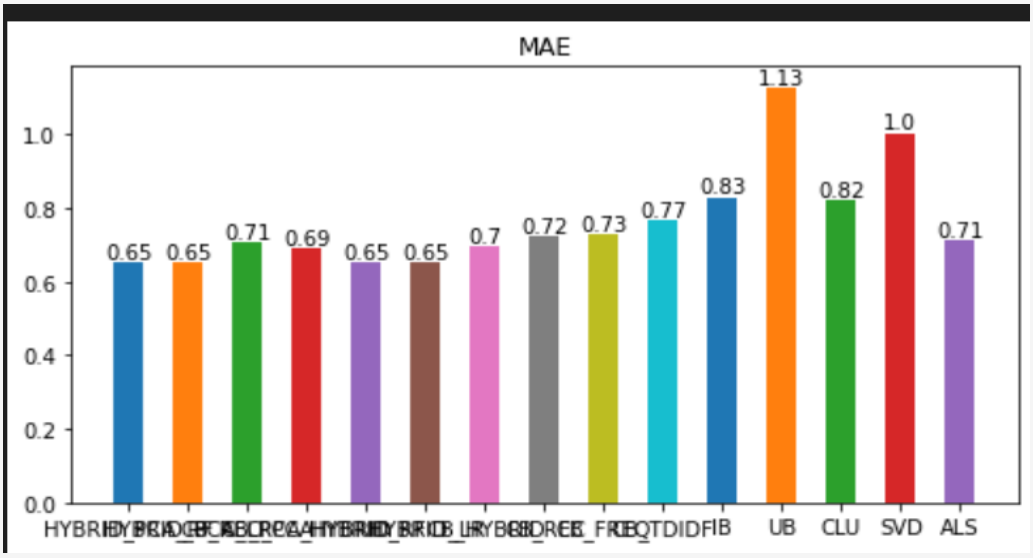
In addition, we also wanted to evaluate the performance of PCA on recency-based content-based approach to the normal recency-based content-based approach when combined with ALS. So, this is yet another HYBRID recommendation system that we created. We combined the prediction from ALS and PCA based Recency Based Content Based in a data frame and evaluated the performance of these models as discussed earlier. We applied linear regression, random forest and evaluated the results of PCA and ALS approach.

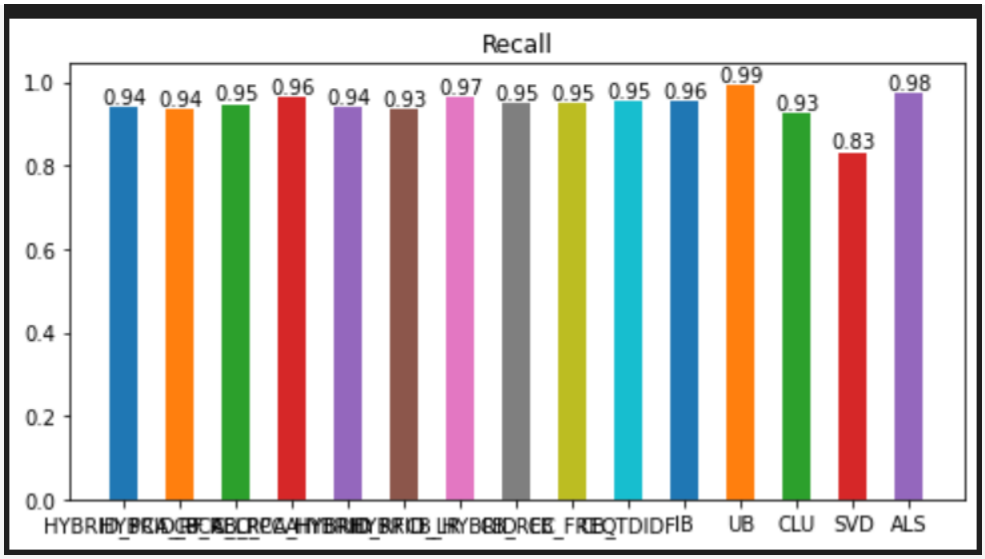
	HYBRID_PCA_RF	HYBRID_PCA_LR	CB_REC_PCA	CB_PCA_HYBRID	HYBRID_RF	HYBRID_LR	CB_HYBRID
RMSE	0.863515	0.868890	0.969446	0.918862	0.862606	0.868621	0.928663
MAE	0.651021	0.651761	0.708825	0.689146	0.652072	0.651895	0.695740
Recall	0.940498	0.935247	0.947831	0.963498	0.941623	0.934747	0.966082
Precision	0.974947	0.976761	0.959222	0.955930	0.974556	0.976537	0.951492
F1	0.957413	0.955554	0.953493	0.959700	0.957806	0.955185	0.958731
NDCG@5	0.857948	0.855824	0.882910	0.854903	0.858320	0.855145	0.856330

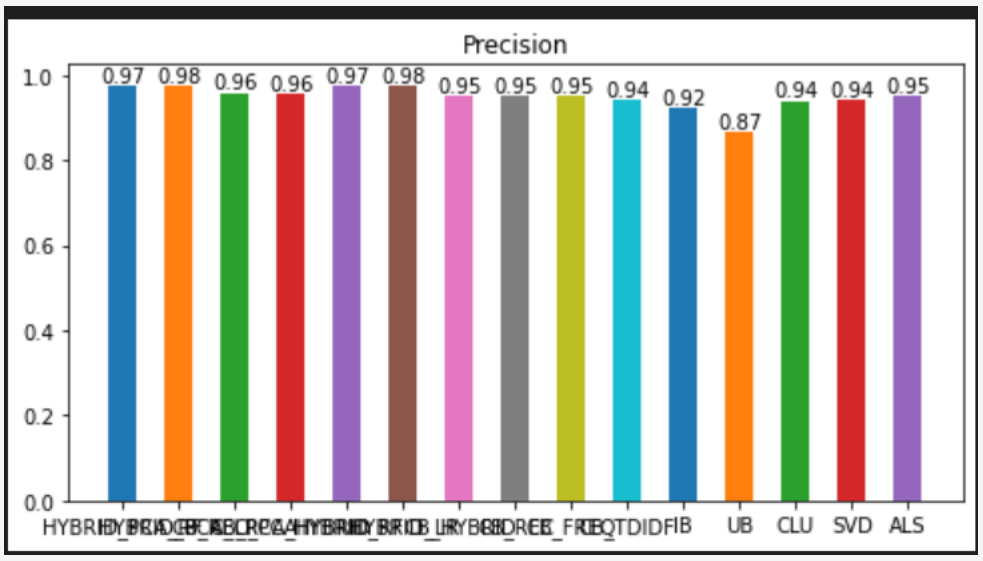
We see that PCA does improved the RMSE for the HYBRID but with linear and random forest the results are quite close, and we see that random forest without the PCA still performs better.

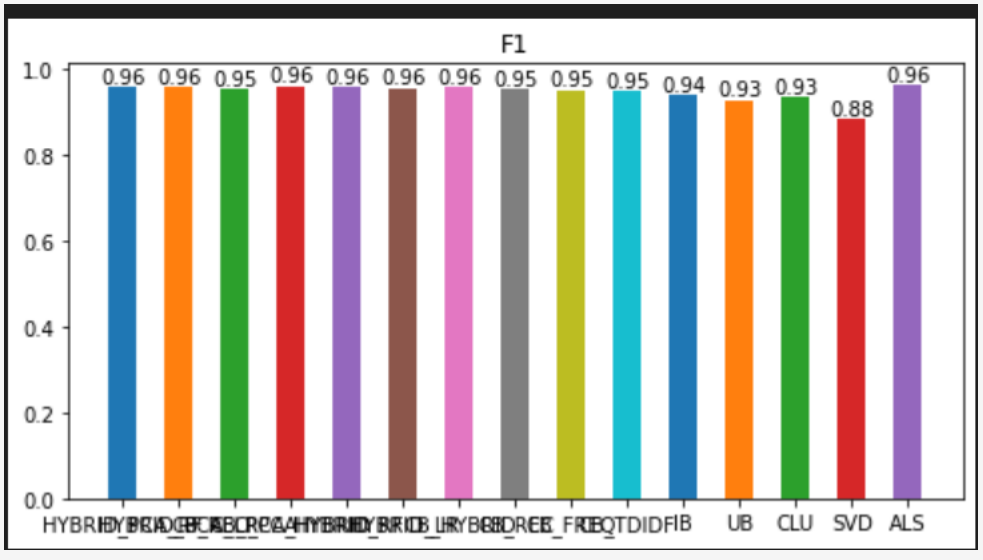
Lets check the graphs for RMSE, MAE, PRECISION, RECALL , F1 and NDCG@5 for all the models and we will pick the best performing models from each of the collaborative, content based and Hybrid to make artist recommendation for user and see how the recommendations hold up.

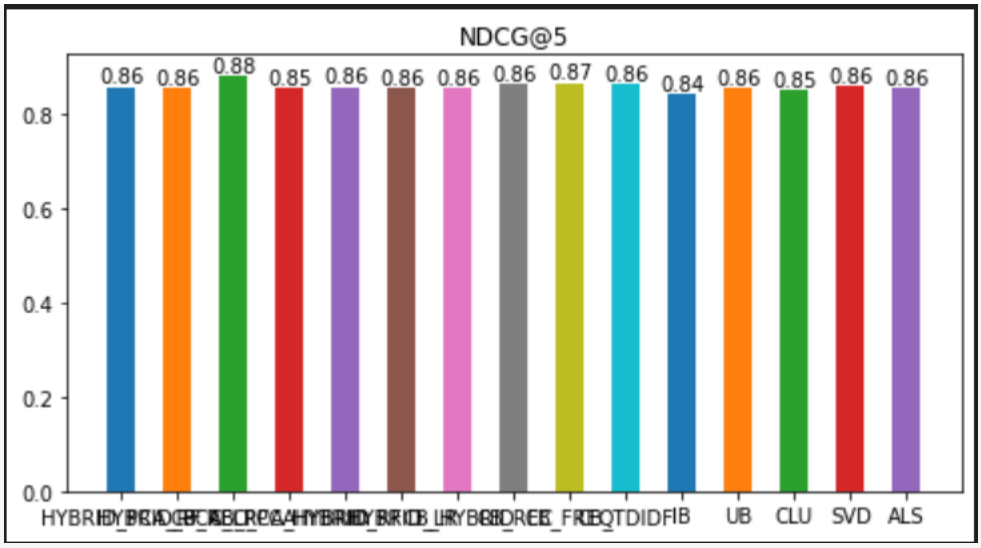

RMSE
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
0.86
0.87
0.97
0.92
0.86
0.87
0.93
1.0
1.0
1.06
1.16
1.5
1.11
1.42
0.94
UB
CLU
SVD
ALS


MAE
0.0
0.2
0.4
0.6
0.8
1.0
0.65
0.65
0.71
0.69
0.65
0.65
0.7
0.72
0.73
0.77
0.83
1.13
0.82
1.0
0.71
UB
CLU
SVD
ALS


Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.94
0.94
0.95
0.96
0.94
0.93
0.97
0.95
0.95
0.95
0.96
0.99
0.93
0.83
0.98
UB
CLU
SVD
ALS


Precision
1.0
0.8
0.6
0.4
0.2
0.0
0.97
0.98
0.96
0.96
0.97
0.98
0.95
0.95
0.95
0.94
0.92
0.87
0.94
0.94
0.95
UB
CLU
SVD
ALS


F1
1.0
0.8
0.6
0.4
0.2
0.0
0.96
0.96
0.95
0.96
0.96
0.96
0.96
0.95
0.95
0.95
0.94
0.93
0.93
0.88
0.96
UB
CLU
SVD
ALS



Based on the graphs as we discussed earlier we clearly see that ALS and Co Clustering are giving us the best performance from collaborative filtering approach, while in content based we have recency based content based and PCA enhanced recency based content based giving the best RMSE and in hybrid we have random forest which gives the best RMSE.

5. Recommendation For User

We will first see how collaborative filtering approaches like ALS and CO clustering are making artist predictions.

For the purpose of recommendations, we will be using the function

utils.get_top_n() The function returns the top-N recommendation for each user from a set of predictions.

Let's first check the tags for user id 200

	tagValue
userID	
200	[rock, classic rock, british, hard rock, nwobhm, ambient, new age, instrumental, progressive rock, rock]

1. Using ALS for user id 200 we see that artist id 65, 4125, 4130, 998, and 4131 are the recommeneded artsists.

```
from IESEGRecSys import utils

results = utils.get_top_n(als_pred,  n = 5,  user_col =  'userID', item_col='artistID')
results.head()
```

✓ 0.5s

	user	item
0	200	[65, 4125, 4130, 998, 4131]
1	1833	[292, 295, 4252, 299, 86]
2	293	[679, 227, 498, 230, 412]
3	1102	[906, 467, 2020, 298, 97]
4	938	[300, 5459, 291, 499, 680]

We will further investigate the tags associated with the artists to see if they have similar tags or not.

Tags for Artist Id 65

artistID	tagValue
65	[00s, 2002, 2005, 2008, 4m4zinq, 90s, acoustic, alternative, alternative rock, amazing, ambient, atmospheric, awesome, b-side, ballad, beatiful, beautiful, beautiful music, ben lee, best songs ever, brian eno, brilliant, brit rock, british, britpop, britrock, britsong, caaamiiiliiitaaa, chill, chillout, cities, classic rock, cold play, coldplay, cool, cover, cute, demo, dreamy, electronic, epic, eternal love, european, experimental, fail, favorite albums, favorite artist, favorite artists, favorite bands, favorite bands and artists, favorite song, favorite songs, favorites, favourite, favourite artists, favourite songs, favourites, folk, free, free download, free mp3, fucking awesome, fun, galactic, garden state soundtrack, genius, gold, good stuff, great song, hang on, happy, i love, indie, indie pop, indie rock, indierock, inlove, isa, johnny cash, jukebox project, just chillin, kraftwerk, lastfm elitist repellent, lights, listened more than 10 times in a row, love, love at first listen, love it, love ya lll, lovely, lpa, male vocalists, marisa mix, melancholic, mellow, metal, musikfuerrundumdieuhr, muzyka dla mietkich chlopcow, my own private sunshine, naked, ...]

Tags for Artist Id 4125

artistID	tagValue
4125	[1991, 60s, 70s, 80s, 90s, alan lancaster, andy bown, best christmas songs, british, classic rock, england, francis rossi, guitar, hard rock, jeff rich, john coghlan, male vocalists, matt letley, pop, pop rock, psychedelic, rhino edwards, rick parfitt, rock, rock and roll, rock n roll, soft rock]

Tags for Artist 4130

artistID	tagValue
4130	[1984, 70s, 70ss, 80s, b side, ballad, classic, classic rock, cold case soundtrack, favorites, good memories, hard rock, hidden gem, late night love, late night tunes, love making music, male vocals, mellow, one of their best, pop, progressive rock, random songs, rock, seen in concert, seen live, soft rock, ultimate rock ballads]

We clearly see that recommendations from ALS are closely related to the user tags and the artists have tags mostly common with user id 200. These artists seem to have British, Classic Rock and Rock in common. The artists also seem to share a lot of tags amongst themselves.

Let’s see further how co clustering is performing in terms of recommendations

```
results = utils.get_top_n(predclu, n = 5, user_col = 'userID', item_col='artistID')
results.head()
```

✓ 1.1s

	user	item
0	200	[4131, 4124, 998, 4130, 65]
1	1833	[292, 4252, 16916, 299, 295]
2	293	[5307, 230, 5311, 5309, 5323]
3	1102	[467, 58, 906, 97, 298]
4	938	[7191, 300, 9193, 499, 9187]

Looking at the results from Co clustering and ALS we notice that artists 998 and 4131 are there in Top 5 recommendations for user id 200 from both the algorithms. Let's further evaluate the tags and check if the user and artists have any tags in common.

The tags for artist ID 4131 are

artistID	tagValue
4131	[british, english]

The tags for artist ID 4124 are

artistID	tagValue
4124	[70s, classic rock, favorite songs, guilty pleasure, hard rock, heavy metal, late night tunes, progressive rock, really sad thing, rock, space rock, stoner rock, top]

The tags for artist ID 998 are

artistID	tagValue
998	[1008, 80s, 80s pop, alternative, alternative dance, b-side, beautiful, blitz playlist, british, british artists, cities, covers, death, drugs, edit, electronic, experimental, factory records, famous persons, fav artists, favorite bands, favorite tracks, instrumental, level one, liverpool, mad about, my loving, my pleasuredome, new romantic, new wave, old, omd, piano, po-kraftwerkski, pop, random songs, religion, science, slowies, soundtrack, synth pop, synthpop, synthpop artists, the beowulfs choice]

Based on the tags we notice that the artists do not have many tags in common with the user and themselves. Thus, it would be fair to say ALS is making better recommendations in terms of user tags in this case.

Next, we will be doing the same tasks for our best content-based model. In our case we will be using recency-based content-based model to get these recommendations as that was our best performing model in content based. We would also check the results for frequency-based content-based model to gain further insights. Lastly, we will use our PCA enhanced recency-based content-based model and evaluate the results.

The top5 artist recommendations based on recency based content based are

```
results = predict_user_topn(cb2, train, 200, topk=5, item_col='artistID')
results
```

✓ 1.6s

```
[4126, 723, 5173, 607, 3269]
```

Let's check the tags for these artists.

The tags for artist ID 4126 are

artistID	tagValue
4126	[1982, 60s, 80s, 90s, ambient, angelic, awesome, beautiful, best songs of the 80s, chef musinum, chill zone, chillout, electronic, electronica, epic, experimental, fallout, faves, instrumental, jean marc toma, lostfreeq, meditation, new age, new beat, piano, sensual, slowies, soundtrack, the best classical opera and film scores]

The tags for artist ID 723 are

artistID	tagValue
723	[arkadi, folk, instrumental, jazz, latin, world]

The tags for artist ID 5173 are

artistID	tagValue
5173	[drum and bass, electronic, experimental, instrumental, progressive, progressive rock, psychedelic]

Let's repeat the same activity with frequency-based content-based model as well and then we can draw our conclusions.

The top5 artist recommendations based on recency based content based are

```
results = predict_user_topn(cb1, train, 200, topk=5, item_col='artistID')
results
```

✓ 1.6s

```
[723, 963, 1148, 2343, 11484]
```

Let's check the tags for these artists as well

The tags for artist ID 723 are

artistID	tagValue
723	[arkadi, folk, instrumental, jazz, latin, world]

The tags for artist ID 963 are

artistID	tagValue
963	[amazing, ambient, breathtaking, chillout, dreamy, fascinating, instrumental, new age, piano, rain, seen live]

The tags for artist ID 723 are

artistID	tagValue
1148	[ambient, coisa de deus, electronic, electronica, epic, japanese, new age]

The results show that the user gets instrumental tag in common using recency based content based system and the frequency based recommendations don't seem to have any tags in common but there are quite a few overlapping tags like ambient, instrumental in the recommendations.

6. Conclusion

In conclusion we can say that HYBRID models give the best performance for our RMSE scores.

Random forest HYBRID model which comprised of ALS prediction result and recency-based content-based prediction result seems to perform the best.

Recommendations made to the user id 200 were in similar alliance with user tags. The user gets very personalized recommendation of artist using the collaborative approach of ALS and co clustering while content based systems like recency based and frequency based seems to be making personalized recommendations but it also seems that it is engaging the user with a much larger selection of tags thus diversifying the user interests. All in all we can draw a conclusion that using a recommendation system like this would help in the companies profiting as relevancy in the ratings indicated a call to action from the user side marking the importance of such systems.