PROGRESS REPORT

PREDICTION OF POTENTIAL DRUG-TARGET INTERACTIONS USING PROBABILISTIC MATRIX FACTORIZATION

INFSCI 2725 DATA ANALYTICS

JULY 21, 2019

GROUP MEMBERS:

PREM RAJGOPAL: PRR47@pitt.edu

TSHERING SHERPA: TTS12@pitt.edu

ZHENG HAN: ZHH41@pitt.edu

Problem Identification

We chose to study the unknown interactions between existing drugs and the well-known proteins in human or animals. This prediction will provide a guideline for the future drug discovery research and also help to explain the side effects of certain drugs.

Questions To Be Answered

Q1. For the currently reported interactions, find out the average number of targets for the drugs, and list the top 100 drugs according to the number of targets. Draw a histogram for the top 100 drugs in terms of the number of targets.

Q2. Predict the unknown interactions between drugs and targets. In other words, look for more drug-target pairs. Predict the interaction probability of each newly found drug-target pair.

Q3. The reported interactions serve as the positive samples. A certain number of negative samples need to be generated for training the model. The effect of the number of negative samples on the RMSE and AUC of the model will be studied.

What We Have Accomplished So Far

We have successfully answered Q1. For Q2 and Q3, we generated preliminary results with limited accuracy, and we are still trying to improve the accuracy.

Preliminary Results

Q1:

The original data set downloaded from DrugBank database was imported into a Jupyter Notebook and it looks like below:

```
data = pd.read_csv('uniprot links_2.csv')
data.head()
```

	DrugBankID	Name	Type	UniProtID	UniProtName
0	DB00001	Lepirudin	BiotechDrug	P00734	Prothrombin
1	DB00002	Cetuximab	BiotechDrug	P00533	Epidermal growth factor receptor
2	DB00002	Cetuximab	BiotechDrug	O75015	Low affinity immunoglobulin gamma Fc region re...
3	DB00002	Cetuximab	BiotechDrug	P00736	Complement C1r subcomponent
4	DB00002	Cetuximab	BiotechDrug	P02745	Complement C1q subcomponent subunit A

By using data.describe, we know that there are 7370 drugs represented by “DrugBankID” and 4763 proteins (targets) represented by “UniProtID” as shown below:

```
data.describe()
```

	DrugBankID	Name	Type	UniProtID	UniProtName
count	20061	20061	20061	20061	20061
unique	7370	7370	2	4763	4255
top	DB12010	Fostamatinib	SmallMoleculeDrug	P24941	Cyclin-dependent kinase 2
freq	306	306	19134	136	136

By grouping “DrugBankID”, we know how many targets for each drug. And then by using mean() function, we know the average number of targets for the drugs is about 2.72.

```
grouped=data.groupby(['DrugBankID']).count()
grouped.head()
```

DrugBankID	Name	Type	UniProtID	UniProtName
DB00001	1	1	1	1
DB00002	12	12	12	12
DB00004	3	3	3	3
DB00005	14	14	14	14
DB00006	1	1	1	1

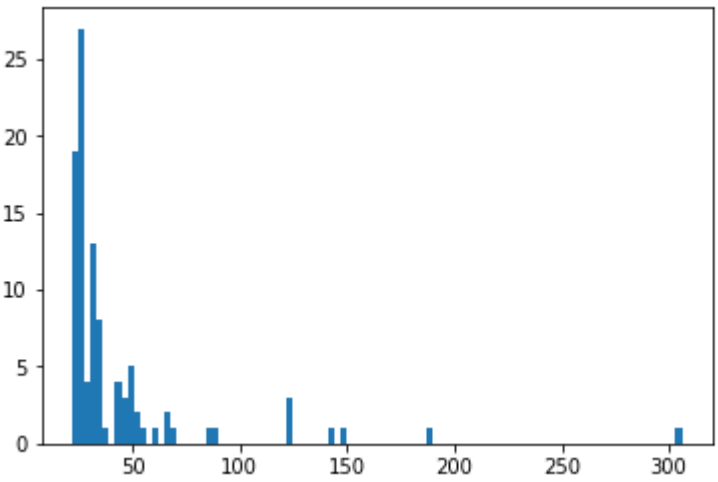
```
grouped['UniProtID'].mean()
```

```
2.7219810040705563
```

Then the drugs are sorted according to the number of targets and the top 100 drugs are displayed. A histogram is drawn to show the distribution of the number of targets.

```
first100=grouped.sort_values('Name', ascending=False).head(100)
first100.head()
```

DrugBankID	Name	Type	UniProtID	UniProtName
DB12010	306	306	306	306
DB11638	189	189	189	189
DB09130	147	147	147	147
DB00157	144	144	144	144
DB01593	124	124	124	124



Q2:

The Probabilistic Matrix Factorization (PMF) model [1] is used for our prediction. We found the implementation of PMF from Github [2]. This author used PMF to predict the user's rating on the movies that the user has not watched. The rating score ranges from 1 to 5 in his work. The prediction of drug and target interaction is similar to the prediction of user and movie relationship. But our scale is 0 to 1 with 0 indicating no interaction and 1 indicating interaction with a probability of 100%.

In order to use the PMF model, the drug ID, target ID and interaction scale have to be changed to integer. Below is what we did to change the labels.

All the DrugBankIDs are saved in a dictionary with the DrugBankID as the *key* and the integer value starting from 1 as the *value*:

```
dict1 = {}
counter = 1
for item in set(data['DrugBankID']):
    dict1[item] = counter
    counter += 1
print(dict1)
```

```
{'DB00913': 1, 'DB05890': 2, 'DB03028': 3, 'DB07312': 4, 'DB00687': 5, 'DB03876': 6, 'DB04891': 7, 'DB14562': 8, 'DB12442':
9, 'DB08049': 10, 'DB04852': 11, 'DB07806': 12, 'DB02767': 13, 'DB03254': 14, 'DB04459': 15, 'DB07440': 16, 'DB04547': 17, 'D
B07733': 18, 'DB02120': 19, 'DB14541': 20, 'DB09107': 21, 'DB03961': 22, 'DB03568': 23, 'DB05630': 24, 'DB02954': 25, 'DB0785
0': 26, 'DB07641': 27, 'DB03590': 28, 'DB00491': 29, 'DB07409': 30, 'DB00357': 31, 'DB03887': 32, 'DB07020': 33, 'DB07696': 3
4, 'DB00377': 35, 'DB00590': 36, 'DB08096': 37, 'DB03921': 38, 'DB06247': 39, 'DB04743': 40, 'DB09510': 41, 'DB05271': 42, 'D
B07111': 43, 'DB07748': 44, 'DB00723': 45, 'DB08145': 46, 'DB08607': 47, 'DB03022': 48, 'DB03827': 49, 'DB06938': 50, 'DB0634
5': 51, 'DB04284': 52, 'DB04482': 53, 'DB05434': 54, 'DB03766': 55, 'DB01145': 56, 'DB07628': 57, 'DB11580': 58, 'DB02741': 5
9, 'DB00957': 60, 'DB08637': 61, 'DB08299': 62, 'DB01613': 63, 'DB04223': 64, 'DB08810': 65, 'DB09026': 66, 'DB03210': 67, 'D
B06237': 68, 'DB04081': 69, 'DB05967': 70, 'DB00750': 71, 'DB04672': 72, 'DB00407': 73, 'DB04562': 74, 'DB03723': 75, 'DB0198
0': 76, 'DB11397': 77, 'DB08614': 78, 'DB07955': 79, 'DB00398': 80, 'DB07249': 81, 'DB01419': 82, 'DB01268': 83, 'DB08067': 8
4, 'DB07276': 85, 'DB02264': 86, 'DB08357': 87, 'DB03240': 88, 'DB00534': 89, 'DB04629': 90, 'DB06691': 91, 'DB13943': 92, 'D
B04271': 93, 'DB13165': 94, 'DB04235': 95, 'DB14004': 96, 'DB08727': 97, 'DB07205': 98, 'DB01158': 99, 'DB02790': 100, 'DB022
38': 101, 'DB08466': 102, 'DB12872': 103, 'DB03658': 104, 'DB03358': 105, 'DB08062': 106, 'DB03577': 107, 'DB08360': 108, 'DB
05448': 109, 'DB03331': 110, 'DB04793': 111, 'DB03556': 112, 'DB06977': 113, 'DB11071': 114, 'DB06710': 115, 'DB05276': 116,
'DB07889': 117, 'DB00485': 118, 'DB02366': 119, 'DB03480': 120, 'DB02494': 121, 'DB07602': 122, 'DB09472': 123, 'DB03821': 12
4, 'DB00290': 125, 'DB00474': 126, 'DB02425': 127, 'DB03181': 128, 'DB05944': 129, 'DB00613': 130, 'DB07268': 131, 'DB00940':
132, 'DB02375': 133, 'DB07968': 134, 'DB06674': 135, 'DB07175': 136, 'DB07219': 137, 'DB07326': 138, 'DB06226': 139, 'DB0780
8': 140, 'DB03668': 141, 'DB06263': 142, 'DB04261': 143, 'DB03416': 144, 'DB05023': 145, 'DB07507': 146, 'DB04040': 147, 'DB0
2095': 148, 'DB08295': 149, 'DB00889': 150, 'DB06446': 151, 'DB04903': 152, 'DB04895': 153, 'DB03868': 154, 'DB00961': 155,
```

Then by using the map() function, a new column of “DrugBankIDLabel” can be generated and its values are integers.

```
data["DrugBankIDLabel"]=data["DrugBankID"].map(dict1)
data.head()
```

	DrugBankID	Name	Type	UniProtID	UniProtName	DrugBankIDLabel
0	DB00001	Lepirudin	BiotechDrug	P00734	Prothrombin	593
1	DB00002	Cetuximab	BiotechDrug	P00533	Epidermal growth factor receptor	6639
2	DB00002	Cetuximab	BiotechDrug	O75015	Low affinity immunoglobulin gamma Fc region re...	6639
3	DB00002	Cetuximab	BiotechDrug	P00736	Complement C1r subcomponent	6639
4	DB00002	Cetuximab	BiotechDrug	P02745	Complement C1q subcomponent subunit A	6639

Similarly, a new column of “UniProtIDLabel” is generated and its values are integers.

```
data["UniProtIDLabel"]=data["UniProtID"].map(dict2)
data.head()
```

	DrugBankID	Name	Type	UniProtID	UniProtName	DrugBankIDLabel	UniProtIDLabel
0	DB00001	Lepirudin	BiotechDrug	P00734	Prothrombin	593	1299
1	DB00002	Cetuximab	BiotechDrug	P00533	Epidermal growth factor receptor	6639	3561
2	DB00002	Cetuximab	BiotechDrug	O75015	Low affinity immunoglobulin gamma Fc region re...	6639	4525
3	DB00002	Cetuximab	BiotechDrug	P00736	Complement C1r subcomponent	6639	3283
4	DB00002	Cetuximab	BiotechDrug	P02745	Complement C1q subcomponent subunit A	6639	2352

Because all the drug-target pairs from DrugBank are proved to be interacting with each other, the interaction probability is 100%. We generated another new column called “Interaction” and put 1 as the values for all these reported drug-target pairs. We call these positive samples. The data set with only positive samples looks like below (20061 rows × 3 columns):

	DrugBankIDLabel	UniProtIDLabel	Interaction
0	593	1299	1
1	6639	3561	1
2	6639	4525	1
3	6639	3283	1
4	6639	2352	1
5	6639	2487	1
6	6639	2212	1
7	6639	2450	1
8	6639	2193	1
9	6639	432	1
10	6639	114	1
11	6639	1277	1
12	6639	266	1
13	1146	2731	1
14	1146	3575	1
15	1146	2003	1
16	2515	971	1
17	2515	931	1
18	2515	432	1
19	2515	2450	1

In order to train a good model, we have to generate negative samples. In our case, the negative sample means a drug-target pair with an interaction probability of 0. The Drugbank database only provides positive samples while there are no experimentally-validated negative samples. So we

can only assume some drug-target pairs have 0 probability to be interacting with each other although they must not be real negative samples. The way we generate the negative samples is to randomly generate a drug-target pair and then verify if it exists in the positive sample set, if yes, then re-generate a new pair, if no, this pair is saved, and then go back the for loop to generate another pair until the loop ends. The below code iterates 30000 times and "data1" finally includes 20061 positive samples and 29992 negative samples.

```
import random
for i in range(30000):
    a=random.randint(1,7370)
    b=random.randint(1,4763)
    if (a,b) in dict3:
        continue
    data1.loc[20061+i]=[a,b,0]
data1
```

	DrugBankIDLabel	UniProtIDLabel	Interaction
0	593	1299	1
1	6639	3561	1
2	6639	4525	1
3	6639	3283	1
4	6639	2352	1
5	6639	2487	1
6	6639	2212	1
7	6639	2450	1
8	6639	2193	1

The above data set was split into train set and test set by:

(train, test) = train_test_split(data, train_size = 0.8)

We have to assign a value for the below parameters of the PMF model:

Parameter	Meaning
n_user	It is the total number of drugs in our case. The value is 7370.
n_item	It is the total number of targets in our case. The value is 4763.
n_feature	It tells how many features can uniquely describe a drug and a target. It composes an embedding vector for each drug and each target. Intrinsically, the inner product of the drug's embedding vector and the target's embedding vector gives the interaction probability of this drug and this target. We plan to run the PMF model a few times to find an optimal n_feature value.
n_iters	The number of iterations. The higher value of n_iters is supposed to give a better prediction but the running time will be increasing drastically.
max_rating	In our case, it is 1.
min_rating	In our case, it is 0.

We firstly tried this combination of parameters: n_user=7370, n_item=4763, n_feature=30, n_iters=10, max_rating=1, min_rating=0. And we obtained below results:

```
print("training pmf...")
pmf = PMF(n_user=n_user, n_item=n_items,n_feature = n_feat, epsilon = 5,max_rating=1, min_rating=0)
pmf.fit(train,n_iters=eval_iters)
```

```
training pmf...

<recommend.pmf.PMF at 0x267290190f0>
```

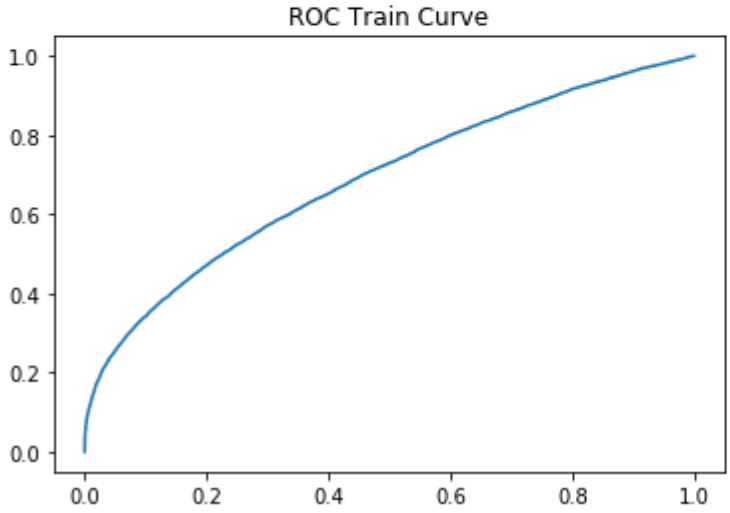
```python
from recommend.utils.evaluation import RMSE

train_preds = pmf.predict(train[:,:2])
train_rmse = RMSE(train_preds, train[:,2])

test_preds = pmf.predict(test[:,:2])
test_rmse = RMSE(test_preds,test[:,2])

print('Train RMSE: ', train_rmse)
print('Test RMSE: ', test_rmse)
print()
print('Train Preds: ')
print(train_preds)
print()
print("Test Preds: ")
print(test_preds)

print("Test min: ",test_preds.min())
print('Test max: ', test_preds.max())
print("Train min: ",train_preds.min())
print('Train max: ', train_preds.max())
```

```
Train RMSE:  0.49113256485741547
Test RMSE:  0.491773108704835

Train Preds:
[0.48347627 0.46815479 0.54032976 ... 0.48009259 0.47584957 0.48551968]


Test Preds:
[0.49458976 0.48517228 0.46688036 ... 0.46846345 0.504811   0.47957349]
Test min:  0.4341419607311734
Test max:  0.5835814951246646
Train min:  0.42807871183165375
Train max:  0.591980499102021
```

```
(fpr_train, tpr_train, thresholds_train) = roc_curve(train[:,2],train_preds)
auc_train = roc_auc_score(train[:,2],train_preds)
```

```
plt.plot(fpr_train,tpr_train)
plt.title('ROC Train Curve')
print("AUC Train: ", auc_train)
```

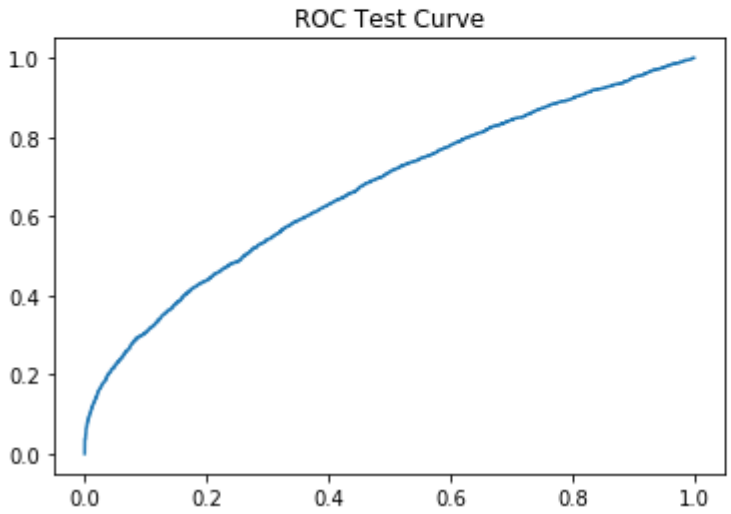
```
AUC Train:  0.6877667795585204
```



```
(fpr, tpr, thresholds) = roc_curve(test[:,2],test_preds)
auc_test = roc_auc_score(test[:,2],test_preds)
```

```
plt.plot(fpr,tpr)
plt.title("ROC Test Curve")
print('AUC Test: ', auc_test)
```

```
AUC Test:  0.6664035608358009
```



The results are summarized in below table:

Performance Indicator	Value
RMSE for train set	0.4911
RMSE for test set	0.4918
AUC score for train set	0.6878
AUC score for test set	0.6664

Results for: n_user=7370, n_item=4763, n_feature=30, n_iters=10, max_rating=1, min_rating=0

Similarly, we ran the model using different combinations of parameters and the results are listed below:

Performance Indicator	Value
RMSE for train set	0.5119
RMSE for test set	0.5155
AUC score for train set	0.7331
AUC score for test set	0.6875

Results for: n_user=7370, n_item=4763, n_feature=70, n_iters=10, max_rating=1, min_rating=0

Performance Indicator	Value
RMSE for train set	0.4643
RMSE for test set	0.4703
AUC score for train set	0.8773
AUC score for test set	0.7966

Results for: n_user=7370, n_item=4763, n_feature=30, n_iters=50, max_rating=1, min_rating=0

Performance Indicator	Value
RMSE for train set	0.4443
RMSE for test set	0.4585
AUC score for train set	0.9078
AUC score for test set	0.8010

Results for: n_user=7370, n_item=4763, n_feature=30, n_iters=100, max_rating=1, min_rating=0

Comparing the above four results, the current optimal result occurred at n_feature=30, n_iters=100. The predicted data was exported to "PMF Generated data_30000_fea30_ite100.csv". The current model can be used to predict the interaction probability for any drug-target pair but with limited accuracy. Taking DrugBankIDLabel #1 for example, we paired it with all the 4763 targets and predicted the interaction probability of all these pairs. Below is the screenshot of the result:

DrugBankIDLabel	UniProtIDLabel	Predicted interaction
1	1	0.432771164
1	2	0.447979408
1	3	0.428092253
1	4	0.518644897
1	5	0.429846909
1	6	0.425531057
1	7	0.429120209
1	8	0.463735628
1	9	0.454380813
1	10	0.430469087
1	11	0.416836863
1	12	0.45221196
1	13	0.424542773
1	14	0.562239136
1	15	0.457691837
1	16	0.427313679
1	17	0.439664403
1	18	0.44841151
1	19	0.457195833
1	20	0.444873125

Q3:

It has been reported that the quality of the negative data set affects the prediction results [3]. The chemical and structural similarity has been incorporated into the procedure of generating

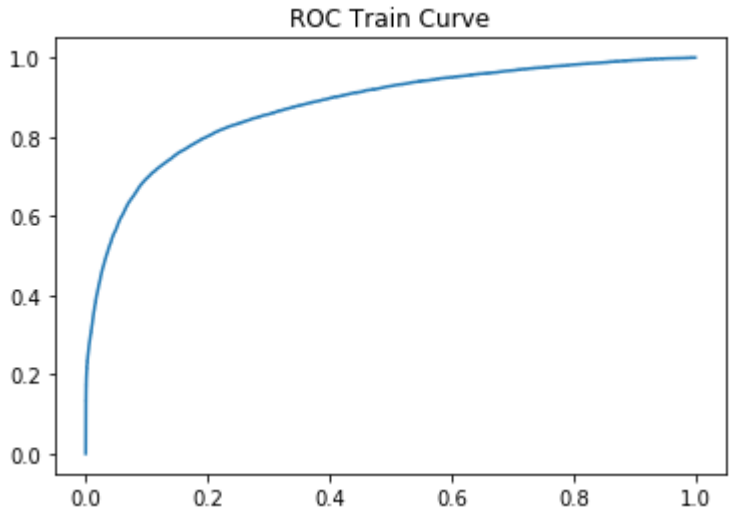
the negative data set [2]. In our study, we did randomly pairing of the drug and the target and assume their interaction probability to be 0. And we varied the number of negative samples to form different data sets aiming to see how the number of negative samples affects the prediction performance.

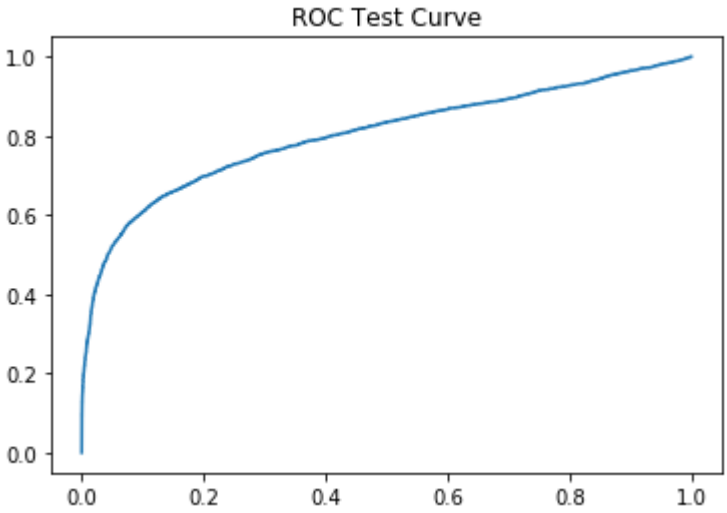
We generated another data set with 20061 positive samples and 79955 negative samples. We split the data set by using "(train, test) = train_test_split(data, train_size = 0.8) and used n_feature=30, n_iters=100 to fit the PMF model on the train data. The results are summarized as below:

Performance Indicator	Value
RMSE for train set	0.3692
RMSE for test set	0.3771
AUC score for train set	0.8760
AUC score for test set	0.8054

Results for: n_user=7370, n_item=4763, n_feature=30, n_iters=100, max_rating=1, min_rating=0

The ROC curves as below:





We can see that when the number of the negative samples increases from 29992 to 79955, the RMSE is lowered by about 17.8% and the AUC score almost stays the same. The reduction in RMSE might be because more negative samples bring more zero(s) into the calculation. So it looks to us that there is no improvement in the prediction accuracy when there are more negative samples.

Problem Report

The predicted interaction probability for the positive samples is supposed to be very close to 1 but it is actually much smaller than 1 as shown below. And as we have seen in the tables above, the RMSE and AUC score still have some room to improve. The future work will have to take the chemical similarity into consideration when generating the negative samples. For example, given a known interaction of drug *i* and target *j*, for a drug *k*, the more difference between drug *i* and drug *k*, the less possibility that drug *k* interacts target *j.* We expect to see improved prediction accuracy after the chemical similarity is taken into account.

	DrugBankIDLabel	UniProtIDLabel	Interaction	Predicted interaction
0	592	1298	1	0.701755798
1	6638	3560	1	0.700387909
2	6638	4524	1	0.591572019
3	6638	3282	1	0.659930284
4	6638	2351	1	0.598876777
5	6638	2486	1	0.630448546
6	6638	2211	1	0.61587401
7	6638	2449	1	0.620322638
8	6638	2192	1	0.580231696
9	6638	431	1	0.610358415
10	6638	113	1	0.570520987
11	6638	1276	1	0.636257339
12	6638	265	1	0.594370809
13	1145	2730	1	0.486199189
14	1145	3574	1	0.440785964
15	1145	2002	1	0.461842285
16	2514	970	1	0.656506231
17	2514	930	1	0.476426109
18	2514	431	1	0.592702431
19	2514	2449	1	0.59423371

50032	4509	2207	0	0.45115946
50033	5613	3410	0	0.448661049
50034	5817	1219	0	0.453267948
50035	1707	1696	0	0.501528024
50036	3961	1882	0	0.452753702
50037	5268	706	0	0.429287766
50038	4674	1263	0	0.437456006
50039	384	2921	0	0.455278827
50040	6756	1040	0	0.474904974
50041	6599	2538	0	0.450695837
50042	5438	625	0	0.444811952
50043	3922	2541	0	0.437489241
50044	3655	1630	0	0.429225442
50045	3816	4045	0	0.462735039
50046	57	4336	0	0.446154686
50047	49	2174	0	0.450375719
50048	8	990	0	0.427661522
50049	3198	2865	0	0.436243197
50050	142	781	0	0.435927561
50051	4059	3580	0	0.474465739
50052	6014	3479	0	0.482048911

Screenshots of the partial prediction results for the data set with 29992 negative samples and n_feature=30, n_iters=100.

References

1. Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In Advances in Neural Information Processing Systems; NIPS, 2007 20: 1257-1264

2. Implementation of PMF: https://github.com/chyikwei/recommend/tree/master/recommend

3. Zhanzhan Cheng, Kai Huang, Yang Wang, Hui Liu, Jihong Guan and Shuigeng Zhou. Selecting high-quality negative samples for effectively predicting protein-RNA interactions. BMC Systems Biology. 2017 11 (Suppl 2) :9