用这些算法搞训练集与测试集，进行性别区分

注意对于不同数据集的rmse和mae, 要进行合并:

$$\mathrm{RMSE}(X) = \sqrt{\frac{\sum_{t=1}^{n}(\hat{y}_t - y_t)^2}{n}}$$

$$\mathrm{RMSE}(XY) = \sqrt{\frac{\mathrm{RMSE}(X)^2 \times n_X + \mathrm{RMSE}(Y)^2 \times n_Y}{n_X + n_Y}}$$

$$\mathrm{MAE}(X) = \frac{\sum_{t=1}^{n} |\hat{y}_t - y_t|}{n}$$

$$\mathrm{MAE}(XY) = \frac{\mathrm{MAE}(X) \times n_X + \mathrm{MAE}(Y) \times n_Y}{n_X + n_Y}$$

In [2]:

```
import pandas as pd
import numpy as np
from glob import glob
from time import time

from surprise import Reader
from surprise import Dataset
from surprise.model_selection import cross_validate
from surprise import NormalPredictor
from surprise import KNNBasic
from surprise import KNNWithMeans
from surprise import KNNWithZScore
from surprise import KNNBaseline
from surprise import SVD
from surprise import BaselineOnly
from surprise import SVDpp
from surprise import NMF
from surprise import SlopeOne
from surprise import CoClustering
from surprise.accuracy import rmse, mae
from surprise import accuracy
from surprise.model_selection import train_test_split
from surprise.model_selection import GridSearchCV

from plotly.offline import init_notebook_mode, plot, iplot
import plotly.graph_objs as go
init_notebook_mode(connected=True)
```

In [4]:

```
def build_train_test(df_train, df_test):
    reader = Reader(rating_scale=(1, 5))
    data_train = Dataset.load_from_df(df_train[['user_id', 'movie_id', 'rating']], reader)
    data_train = data_train.build_full_trainset()
    data_test = Dataset.load_from_df(df_test[['user_id', 'movie_id', 'rating']], reader)
    data_test = data_test.build_full_trainset().build_testset()
    return data_train, data_test
```

In [5]:

```
algorithms = {'SVD':SVD(), 'SVDpp':SVDpp(), 'SlopeOne':SlopeOne(), 'NMF':NMF(), 'NormalPredictor':NormalPredi
ctor(),
              'KNNBaseline':KNNBaseline(), 'KNNBasic':KNNBasic(), 'KNNWithMeans':KNNWithMeans(),
              'KNNWithZScore':KNNWithZScore(), 'BaselineOnly':BaselineOnly(), 'CoClustering':CoClustering()}
```

In [20]:

```
def train_single_algorithm(algorithm_name, data_train, data_test, save_model=False):
    algorithms = {'SVD':SVD(), 'SVDpp':SVDpp(), 'SlopeOne':SlopeOne(), 'NMF':NMF(), 'NormalPredictor':NormalP
redictor(),
              'KNNBaseline':KNNBaseline(), 'KNNBasic':KNNBasic(), 'KNNWithMeans':KNNWithMeans(),
              'KNNWithZScore':KNNWithZScore(), 'BaselineOnly':BaselineOnly(), 'CoClustering':CoClustering()}
    assert(algorithm_name in algorithms), "{} does not exist!".format(algorithm_name)
    algo = algorithms[algorithm_name]
    # print("{} training started!".format(algorithm_name))
    start_time = time()
    # results = cross_validate(algo, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
    algo.fit(data_train)
    # print("{} testing started!".format(algorithm_name))
    predictions = algo.test(data_test)
    result = {}
    # result['name'] = algorithm_name
    result['rmse'] = accuracy.rmse(predictions, verbose=True)
    result['mae'] = accuracy.mae(predictions, verbose=True)
    result['n_samples'] = len(data_test)
    if save_model:
        result['model'] = algo
    print("{:<20}|{:.2f} mins|rmse: {:.4f}|mae: {:.4f}".format(algorithm_name,
                                                                (time() - start_time) / 60.,
                                                                result['rmse'],
                                                                result['mae']
                                                               ))
    return result
```

In [7]:

```
# note that this is for total dataset, not single gender!
def get_mean_results(algorithms, all_results_list):
    for curr_algo_name in algorithms.keys():
        curr_algo_rmse = []
        curr_algo_mae = []
        for curr_all_results in all_results_list:
            curr_algo_rmse.append(curr_all_results[curr_algo_name]['rmse'])
            curr_algo_mae.append(curr_all_results[curr_algo_name]['mae'])
        print("{:<20}|rmse: {:.4f}+-{:.4f}|mae: {:.4f}+-{:.4f}".format(curr_algo_name,
                                                                       np.mean(curr_algo_rmse), np.std(curr_algo_
rmse),
                                                                       np.mean(curr_algo_mae), np.std(curr_algo_m
ae),
                                                                      ))
```

In [17]:

```
def combine_rmse(rmse1, len1, rmse2, len2):
    return np.sqrt((rmse1 ** 2 * len1 + rmse2 ** 2 * len2) / (len1 + len2))

def combine_mae(mae1, len1, mae2, len2):
    return (mae1 * len1 + mae2 * len2) / (len1 + len2)
```

In [26]:

```
# note that this is for total dataset, not single gender!
def combine_mf(algorithms, all_results_m, all_results_f):
    all_results = {}
    for curr_algo_name in algorithms.keys():
        rmse_combined = combine_rmse(all_results_m[curr_algo_name]['rmse'],
                                     all_results_m[curr_algo_name]['n_samples'],
                                     all_results_f[curr_algo_name]['rmse'],
                                     all_results_f[curr_algo_name]['n_samples'])
        mae_combined = combine_mae(all_results_m[curr_algo_name]['mae'],
                                   all_results_m[curr_algo_name]['n_samples'],
                                   all_results_f[curr_algo_name]['mae'],
                                   all_results_f[curr_algo_name]['n_samples'])

        all_results[curr_algo_name] = {'rmse':rmse_combined, 'mae':mae_combined}
    return all_results
```

u1

In [8]:

```
# load
df_train = pd.read_csv("data/ml-100k_merged/u1.base")
df_test = pd.read_csv("data/ml-100k_merged/u1.test")
df_test.head(3)
```

Out[8]:

	movie_id	movie_title	user_id	age	sex	occupation	rating
0	1	Toy Story (1995)	5	33	F	other	4
1	2	GoldenEye (1995)	5	33	F	other	3
2	17	From Dusk Till Dawn (1996)	5	33	F	other	4

In [12]:

```
# split by gender
df_train_m = df_train[df_train['sex'] == 'M']
df_test_m = df_test[df_test['sex'] == 'M']
print("Male: train {}|test {}".format(len(df_train_m), len(df_test_m)))

df_train_f = df_train[df_train['sex'] == 'F']
df_test_f = df_test[df_test['sex'] == 'F']
print("Female: train {}|test {}".format(len(df_train_f), len(df_test_f)))
```

```
Male: train 59093|test 15167
Female: train 20907|test 4833
```

In [21]:

```
# male
data_train, data_test = build_train_test(df_train_m, df_test_m)
all_results_m = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_m[algorithm_name] = result
    print("===== ===== ===== =====")
```

```
RMSE: 0.9384
MAE:  0.7396
SVD                 |0.05 mins|rmse: 0.9384|mae: 0.7396
===== ===== ===== =====
RMSE: 0.9194
MAE:  0.7205
SVDpp               |1.70 mins|rmse: 0.9194|mae: 0.7205
===== ===== ===== =====
RMSE: 0.9413
MAE:  0.7383
SlopeOne            |0.02 mins|rmse: 0.9413|mae: 0.7383
===== ===== ===== =====
RMSE: 0.9589
MAE:  0.7523
NMF                 |0.04 mins|rmse: 0.9589|mae: 0.7523
===== ===== ===== =====
RMSE: 1.5005
MAE:  1.2051
NormalPredictor     |0.00 mins|rmse: 1.5005|mae: 1.2051
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9261
MAE:  0.7286
KNNBaseline         |0.03 mins|rmse: 0.9261|mae: 0.7286
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9718
MAE:  0.7703
KNNBasic            |0.03 mins|rmse: 0.9718|mae: 0.7703
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9517
MAE:  0.7486
KNNWithMeans        |0.03 mins|rmse: 0.9517|mae: 0.7486
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9510
MAE:  0.7453
KNNWithZScore       |0.03 mins|rmse: 0.9510|mae: 0.7453
===== ===== ===== =====
Estimating biases using als...
RMSE: 0.9412
MAE:  0.7458
BaselineOnly        |0.00 mins|rmse: 0.9412|mae: 0.7458
===== ===== ===== =====
RMSE: 0.9666
MAE:  0.7563
CoClustering        |0.01 mins|rmse: 0.9666|mae: 0.7563
===== ===== ===== =====
```

In [22]:
```
# female
data_train, data_test = build_train_test(df_train_f, df_test_f)
all_results_f = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_f[algorithm_name] = result
    print("===== ===== ===== =====")
```

```
RMSE: 1.0433
MAE:  0.8292
SVD                 |0.02 mins|rmse: 1.0433|mae: 0.8292
===== ===== ===== =====
RMSE: 1.0231
MAE:  0.8050
SVDpp               |0.66 mins|rmse: 1.0231|mae: 0.8050
===== ===== ===== =====
RMSE: 1.0475
MAE:  0.8222
SlopeOne            |0.01 mins|rmse: 1.0475|mae: 0.8222
===== ===== ===== =====
RMSE: 1.1118
MAE:  0.8688
NMF                 |0.02 mins|rmse: 1.1118|mae: 0.8688
===== ===== ===== =====
RMSE: 1.5975
MAE:  1.2784
NormalPredictor     |0.00 mins|rmse: 1.5975|mae: 1.2784
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0286
MAE:  0.8147
KNNBaseline         |0.01 mins|rmse: 1.0286|mae: 0.8147
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.1172
MAE:  0.8913
KNNBasic            |0.00 mins|rmse: 1.1172|mae: 0.8913
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0468
MAE:  0.8239
KNNWithMeans        |0.01 mins|rmse: 1.0468|mae: 0.8239
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0466
MAE:  0.8182
KNNWithZScore       |0.01 mins|rmse: 1.0466|mae: 0.8182
===== ===== ===== =====
Estimating biases using als...
RMSE: 1.0423
MAE:  0.8351
BaselineOnly        |0.00 mins|rmse: 1.0423|mae: 0.8351
===== ===== ===== =====
RMSE: 1.0837
MAE:  0.8510
CoClustering        |0.01 mins|rmse: 1.0837|mae: 0.8510
===== ===== ===== =====
```

In [25]:
```
print(all_results_m)
print(all_results_f)
```

```
{'SVD': {'rmse': 0.9383862926313519, 'mae': 0.7396122593767015, 'n_samples': 15167}, 'SVDpp': {'rmse': 0.919
4037497411428, 'mae': 0.7205192220296978, 'n_samples': 15167}, 'SlopeOne': {'rmse': 0.9412970931258932, 'ma
e': 0.7382574736595242, 'n_samples': 15167}, 'NMF': {'rmse': 0.9589445650742295, 'mae': 0.7523194873738819,
'n_samples': 15167}, 'NormalPredictor': {'rmse': 1.500544126463222, 'mae': 1.205103549940071, 'n_samples': 1
5167}, 'KNNBaseline': {'rmse': 0.9260552076751267, 'mae': 0.7285941131595712, 'n_samples': 15167}, 'KNNBasi
c': {'rmse': 0.9717718846950745, 'mae': 0.7702955038128292, 'n_samples': 15167}, 'KNNWithMeans': {'rmse': 0.
9517069774985772, 'mae': 0.7485886155989885, 'n_samples': 15167}, 'KNNWithZScore': {'rmse': 0.95103691471835
66, 'mae': 0.7453160009398803, 'n_samples': 15167}, 'BaselineOnly': {'rmse': 0.9412313340115859, 'mae': 0.74
57637435983588, 'n_samples': 15167}, 'CoClustering': {'rmse': 0.9666239938747948, 'mae': 0.7563200760011395,
'n_samples': 15167}}
{'SVD': {'rmse': 1.0432522116249232, 'mae': 0.8292149859524621, 'n_samples': 4833}, 'SVDpp': {'rmse': 1.0231
227380577936, 'mae': 0.8050122893758805, 'n_samples': 4833}, 'SlopeOne': {'rmse': 1.0475136483527672, 'mae':
0.8221744839339707, 'n_samples': 4833}, 'NMF': {'rmse': 1.1117874487075228, 'mae': 0.8687941894725169, 'n_sa
mples': 4833}, 'NormalPredictor': {'rmse': 1.5974751294318625, 'mae': 1.27843966771637, 'n_samples': 4833},
'KNNBaseline': {'rmse': 1.0286108953065138, 'mae': 0.8147285637859992, 'n_samples': 4833}, 'KNNBasic': {'rms
e': 1.1171658279411747, 'mae': 0.8913259771972365, 'n_samples': 4833}, 'KNNWithMeans': {'rmse': 1.0468459876
187062, 'mae': 0.8238796454936679, 'n_samples': 4833}, 'KNNWithZScore': {'rmse': 1.0466049872950942, 'mae':
0.8182216093005554, 'n_samples': 4833}, 'BaselineOnly': {'rmse': 1.0422503591682972, 'mae': 0.83506593420102
52, 'n_samples': 4833}, 'CoClustering': {'rmse': 1.08373730570858, 'mae': 0.8509535489131902, 'n_samples': 4
833}}
```

In [30]:

```
all_results = combine_mf(algorithms, all_results_m, all_results_f)
all_results
```

Out[30]:

```
{'SVD': {'rmse': 0.9647721165907454, 'mae': 0.7612647582537341},
 'SVDpp': {'rmse': 0.9455105200873228, 'mae': 0.7409369717539028},
 'SlopeOne': {'rmse': 0.9680327894188417, 'mae': 0.7585360191923441},
 'NMF': {'rmse': 0.9980260993049863, 'mae': 0.7804655991360171},
 'NormalPredictor': {'rmse': 1.5245323046877295, 'mae': 1.2228252227995702},
 'KNNBaseline': {'rmse': 0.9518507868768281, 'mae': 0.7494085031534475},
 'KNNBasic': {'rmse': 1.0088281647292157, 'mae': 0.7995425177061712},
 'KNNWithMeans': {'rmse': 0.9755478393108111, 'mae': 0.7667826929730378},
 'KNNWithZScore': {'rmse': 0.9749896433604367, 'mae': 0.7629336412002374},
 'BaselineOnly': {'rmse': 0.9666104109249869, 'mae': 0.7673436179574932},
 'CoClustering': {'rmse': 0.9961867574246206, 'mae': 0.7791882547303365}}
```

u2

In [31]:

```
df_train = pd.read_csv("data/ml-100k_merged/u2.base")
df_test = pd.read_csv("data/ml-100k_merged/u2.test")
df_train_m = df_train[df_train['sex'] == 'M']
df_test_m = df_test[df_test['sex'] == 'M']
print("Male: train {}|test {}".format(len(df_train_m), len(df_test_m)))
df_train_f = df_train[df_train['sex'] == 'F']
df_test_f = df_test[df_test['sex'] == 'F']
print("Female: train {}|test {}".format(len(df_train_f), len(df_test_f)))
```

```
Male: train 59287|test 14973
Female: train 20713|test 5027
```

In [32]:

```
data_train, data_test = build_train_test(df_train_m, df_test_m)
all_results_m = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_m[algorithm_name] = result
    print("===== ===== ===== =====")
# female
data_train, data_test = build_train_test(df_train_f, df_test_f)
all_results_f = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_f[algorithm_name] = result
    print("===== ===== ===== =====")
print(all_results_m)
print(all_results_f)
```

```
RMSE: 0.9292
MAE:  0.7331
SVD                 |0.05 mins|rmse: 0.9292|mae: 0.7331
===== ===== ===== =====
RMSE: 0.9163
MAE:  0.7182
SVDpp               |1.67 mins|rmse: 0.9163|mae: 0.7182
===== ===== ===== =====
RMSE: 0.9342
MAE:  0.7326
SlopeOne            |0.03 mins|rmse: 0.9342|mae: 0.7326
===== ===== ===== =====
RMSE: 0.9601
MAE:  0.7516
NMF                 |0.05 mins|rmse: 0.9601|mae: 0.7516
===== ===== ===== =====
RMSE: 1.4958
MAE:  1.1990
NormalPredictor     |0.00 mins|rmse: 1.4958|mae: 1.1990
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9212
MAE:  0.7236
KNNBaseline         |0.03 mins|rmse: 0.9212|mae: 0.7236
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9639
MAE:  0.7594
KNNBasic            |0.02 mins|rmse: 0.9639|mae: 0.7594
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9444
MAE:  0.7422
KNNWithMeans        |0.03 mins|rmse: 0.9444|mae: 0.7422
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9435
MAE:  0.7391
KNNWithZScore       |0.04 mins|rmse: 0.9435|mae: 0.7391
===== ===== ===== =====
Estimating biases using als...
RMSE: 0.9326
MAE:  0.7383
BaselineOnly        |0.00 mins|rmse: 0.9326|mae: 0.7383
===== ===== ===== =====
RMSE: 0.9550
MAE:  0.7464
CoClustering        |0.01 mins|rmse: 0.9550|mae: 0.7464
===== ===== ===== =====
RMSE: 1.0102
MAE:  0.7960
SVD                 |0.02 mins|rmse: 1.0102|mae: 0.7960
===== ===== ===== =====
RMSE: 0.9980
MAE:  0.7847
SVDpp               |0.54 mins|rmse: 0.9980|mae: 0.7847
===== ===== ===== =====
RMSE: 1.0494
MAE:  0.8165
SlopeOne            |0.01 mins|rmse: 1.0494|mae: 0.8165
===== ===== ===== =====
RMSE: 1.0968
MAE:  0.8584
NMF                 |0.02 mins|rmse: 1.0968|mae: 0.8584
===== ===== ===== =====
RMSE: 1.6112
MAE:  1.2966
NormalPredictor     |0.00 mins|rmse: 1.6112|mae: 1.2966
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0180
MAE:  0.7976
KNNBaseline         |0.01 mins|rmse: 1.0180|mae: 0.7976
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.1311
MAE:  0.8935
KNNBasic            |0.00 mins|rmse: 1.1311|mae: 0.8935
===== ===== ===== =====
Computing the msd similarity matrix...
```

```
Done computing similarity matrix.
RMSE: 1.0423
MAE:  0.8118
KNNWithMeans        |0.00 mins|rmse: 1.0423|mae: 0.8118
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0469
MAE:  0.8101
KNNWithZScore       |0.00 mins|rmse: 1.0469|mae: 0.8101
===== ===== ===== =====
Estimating biases using als...
RMSE: 1.0104
MAE:  0.8043
BaselineOnly        |0.00 mins|rmse: 1.0104|mae: 0.8043
===== ===== ===== =====
RMSE: 1.0698
MAE:  0.8328
CoClustering        |0.01 mins|rmse: 1.0698|mae: 0.8328
===== ===== ===== =====
{'SVD': {'rmse': 0.929214385614642, 'mae': 0.733123074917991, 'n_samples': 14973}, 'SVDpp': {'rmse': 0.91625
19721092842, 'mae': 0.7182473114192324, 'n_samples': 14973}, 'SlopeOne': {'rmse': 0.9341968334849607, 'mae':
0.7326311434629396, 'n_samples': 14973}, 'NMF': {'rmse': 0.9601034359067921, 'mae': 0.7515852134327144, 'n_s
amples': 14973}, 'NormalPredictor': {'rmse': 1.4958159980910508, 'mae': 1.1990425952013646, 'n_samples': 149
73}, 'KNNBaseline': {'rmse': 0.9211838414808493, 'mae': 0.7236166336955803, 'n_samples': 14973}, 'KNNBasic':
{'rmse': 0.9638759549629315, 'mae': 0.7593535707684028, 'n_samples': 14973}, 'KNNWithMeans': {'rmse': 0.9443
556935591163, 'mae': 0.7422467060312301, 'n_samples': 14973}, 'KNNWithZScore': {'rmse': 0.9435048977367296,
'mae': 0.7391487771938792, 'n_samples': 14973}, 'BaselineOnly': {'rmse': 0.9326465056126624, 'mae': 0.738261
0195013816, 'n_samples': 14973}, 'CoClustering': {'rmse': 0.9550367379040832, 'mae': 0.7463983587754381, 'n_
samples': 14973}}
{'SVD': {'rmse': 1.0102374409034336, 'mae': 0.7960316747134094, 'n_samples': 5027}, 'SVDpp': {'rmse': 0.9979
728978746284, 'mae': 0.7847173802667002, 'n_samples': 5027}, 'SlopeOne': {'rmse': 1.0494247863106883, 'mae':
0.8165105863155546, 'n_samples': 5027}, 'NMF': {'rmse': 1.0968460543647154, 'mae': 0.8584049710206093, 'n_sa
mples': 5027}, 'NormalPredictor': {'rmse': 1.6111828550868887, 'mae': 1.2966040255137412, 'n_samples': 502
7}, 'KNNBaseline': {'rmse': 1.0179722039033587, 'mae': 0.7975699254156436, 'n_samples': 5027}, 'KNNBasic':
{'rmse': 1.131070539525255, 'mae': 0.8935325683472519, 'n_samples': 5027}, 'KNNWithMeans': {'rmse': 1.042263
3744601377, 'mae': 0.8118404511439838, 'n_samples': 5027}, 'KNNWithZScore': {'rmse': 1.04687878615226, 'ma
e': 0.8101010200349981, 'n_samples': 5027}, 'BaselineOnly': {'rmse': 1.0103692953757475, 'mae': 0.8043330639
914479, 'n_samples': 5027}, 'CoClustering': {'rmse': 1.0698015765666293, 'mae': 0.8327644510625524, 'n_sampl
es': 5027}}
```

In [33]:

```
all_results2 = combine_mf(algorithms, all_results_m, all_results_f)
all_results2
```

Out[33]:

```
{'SVD': {'rmse': 0.9502297574784987, 'mae': 0.7489351514765694},
 'SVDpp': {'rmse': 0.9374630216852063, 'mae': 0.7349545632240433},
 'SlopeOne': {'rmse': 0.9644555230749028, 'mae': 0.7537142414239444},
 'NMF': {'rmse': 0.9962411809215601, 'mae': 0.7784343595024318},
 'NormalPredictor': {'rmse': 1.5256344822974615, 'mae': 1.2235646607103805},
 'KNNBaseline': {'rmse': 0.9464433331058341, 'mae': 0.7422047935694183},
 'KNNBasic': {'rmse': 1.0085115962492353, 'mae': 0.7930794618098466},
 'KNNWithMeans': {'rmse': 0.9698951358328414, 'mae': 0.7597390938653208},
 'KNNWithZScore': {'rmse': 0.9705244380024349, 'mae': 0.7569826234319944},
 'BaselineOnly': {'rmse': 0.9527788458004157, 'mae': 0.7548682278839597},
 'CoClustering': {'rmse': 0.9851415861150675, 'mae': 0.7681064760718043}}
```

u3

In [34]:

```
df_train = pd.read_csv("data/ml-100k_merged/u3.base")
df_test = pd.read_csv("data/ml-100k_merged/u3.test")
df_train_m = df_train[df_train['sex'] == 'M']
df_test_m = df_test[df_test['sex'] == 'M']
print("Male: train {}|test {}".format(len(df_train_m), len(df_test_m)))
df_train_f = df_train[df_train['sex'] == 'F']
df_test_f = df_test[df_test['sex'] == 'F']
print("Female: train {}|test {}".format(len(df_train_f), len(df_test_f)))
```

```
Male: train 59729|test 14531
Female: train 20271|test 5469
```

In [35]:

```
data_train, data_test = build_train_test(df_train_m, df_test_m)
all_results_m = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_m[algorithm_name] = result
    print("===== ===== ===== =====")
# female
data_train, data_test = build_train_test(df_train_f, df_test_f)
all_results_f = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_f[algorithm_name] = result
    print("===== ===== ===== =====")
print(all_results_m)
print(all_results_f)
```

```
RMSE: 0.9233
MAE:  0.7275
SVD                 |0.05 mins|rmse: 0.9233|mae: 0.7275
===== ===== ===== =====
RMSE: 0.9070
MAE:  0.7107
SVDpp               |1.88 mins|rmse: 0.9070|mae: 0.7107
===== ===== ===== =====
RMSE: 0.9288
MAE:  0.7281
SlopeOne            |0.03 mins|rmse: 0.9288|mae: 0.7281
===== ===== ===== =====
RMSE: 0.9496
MAE:  0.7456
NMF                 |0.05 mins|rmse: 0.9496|mae: 0.7456
===== ===== ===== =====
RMSE: 1.4929
MAE:  1.1974
NormalPredictor     |0.00 mins|rmse: 1.4929|mae: 1.1974
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9134
MAE:  0.7176
KNNBaseline         |0.04 mins|rmse: 0.9134|mae: 0.7176
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9564
MAE:  0.7538
KNNBasic            |0.03 mins|rmse: 0.9564|mae: 0.7538
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9355
MAE:  0.7346
KNNWithMeans        |0.03 mins|rmse: 0.9355|mae: 0.7346
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9354
MAE:  0.7316
KNNWithZScore       |0.03 mins|rmse: 0.9354|mae: 0.7316
===== ===== ===== =====
Estimating biases using als...
RMSE: 0.9262
MAE:  0.7318
BaselineOnly        |0.00 mins|rmse: 0.9262|mae: 0.7318
===== ===== ===== =====
RMSE: 0.9412
MAE:  0.7363
CoClustering        |0.02 mins|rmse: 0.9412|mae: 0.7363
===== ===== ===== =====
RMSE: 0.9935
MAE:  0.7884
SVD                 |0.02 mins|rmse: 0.9935|mae: 0.7884
===== ===== ===== =====
RMSE: 0.9840
MAE:  0.7769
SVDpp               |0.59 mins|rmse: 0.9840|mae: 0.7769
===== ===== ===== =====
RMSE: 1.0249
MAE:  0.8104
SlopeOne            |0.01 mins|rmse: 1.0249|mae: 0.8104
===== ===== ===== =====
RMSE: 1.0664
MAE:  0.8379
NMF                 |0.02 mins|rmse: 1.0664|mae: 0.8379
===== ===== ===== =====
RMSE: 1.5306
MAE:  1.2190
NormalPredictor     |0.00 mins|rmse: 1.5306|mae: 1.2190
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9979
MAE:  0.7908
KNNBaseline         |0.01 mins|rmse: 0.9979|mae: 0.7908
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0859
MAE:  0.8616
KNNBasic            |0.00 mins|rmse: 1.0859|mae: 0.8616
===== ===== ===== =====
Computing the msd similarity matrix...
```

```
Done computing similarity matrix.
RMSE: 1.0187
MAE:  0.8032
KNNWithMeans        |0.00 mins|rmse: 1.0187|mae: 0.8032
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0172
MAE:  0.7979
KNNWithZScore       |0.01 mins|rmse: 1.0172|mae: 0.7979
===== ===== ===== =====
Estimating biases using als...
RMSE: 0.9903
MAE:  0.7919
BaselineOnly        |0.00 mins|rmse: 0.9903|mae: 0.7919
===== ===== ===== =====
RMSE: 1.0570
MAE:  0.8308
CoClustering        |0.01 mins|rmse: 1.0570|mae: 0.8308
===== ===== ===== =====
{'SVD': {'rmse': 0.9233261231769833, 'mae': 0.7275448469780914, 'n_samples': 14531}, 'SVDpp': {'rmse': 0.907
0326872660632, 'mae': 0.7106518243841786, 'n_samples': 14531}, 'SlopeOne': {'rmse': 0.9287785845066504, 'ma
e': 0.7281195512351074, 'n_samples': 14531}, 'NMF': {'rmse': 0.949608800092403, 'mae': 0.7456066252387595,
'n_samples': 14531}, 'NormalPredictor': {'rmse': 1.4929410528707299, 'mae': 1.1973588148679182, 'n_samples':
14531}, 'KNNBaseline': {'rmse': 0.9134066622905466, 'mae': 0.717614963794572, 'n_samples': 14531}, 'KNNBasi
c': {'rmse': 0.9563849612453938, 'mae': 0.7538440337160236, 'n_samples': 14531}, 'KNNWithMeans': {'rmse': 0.
9355248189122948, 'mae': 0.7346419580941771, 'n_samples': 14531}, 'KNNWithZScore': {'rmse': 0.93537835183760
02, 'mae': 0.7315822823043118, 'n_samples': 14531}, 'BaselineOnly': {'rmse': 0.9261895489354907, 'mae': 0.73
18161621461988, 'n_samples': 14531}, 'CoClustering': {'rmse': 0.9411583951257683, 'mae': 0.7363249726861354,
'n_samples': 14531}}
{'SVD': {'rmse': 0.9934823456336317, 'mae': 0.7884181637938653, 'n_samples': 5469}, 'SVDpp': {'rmse': 0.9839
593050442988, 'mae': 0.7768769055975204, 'n_samples': 5469}, 'SlopeOne': {'rmse': 1.024852618894576, 'mae':
0.8104404179675362, 'n_samples': 5469}, 'NMF': {'rmse': 1.0663816713331462, 'mae': 0.837883783181098, 'n_sam
ples': 5469}, 'NormalPredictor': {'rmse': 1.5305830779069212, 'mae': 1.219002501132833, 'n_samples': 5469},
'KNNBaseline': {'rmse': 0.9979376921402456, 'mae': 0.7907651026249727, 'n_samples': 5469}, 'KNNBasic': {'rms
e': 1.0858646527432234, 'mae': 0.8615524231600198, 'n_samples': 5469}, 'KNNWithMeans': {'rmse': 1.0186857743
462245, 'mae': 0.8031694042024777, 'n_samples': 5469}, 'KNNWithZScore': {'rmse': 1.0172416850976695, 'mae':
0.7978582253698466, 'n_samples': 5469}, 'BaselineOnly': {'rmse': 0.9903285927085751, 'mae': 0.79194301221105
29, 'n_samples': 5469}, 'CoClustering': {'rmse': 1.05703543046311, 'mae': 0.8308400096470937, 'n_samples': 5
469}}
```

In [36]:

```
all_results3 = combine_mf(algorithms, all_results_m, all_results_f)
all_results3
```

Out[36]:

```
{'SVD': {'rmse': 0.9430289514361351, 'mae': 0.7441906554613648},
 'SVDpp': {'rmse': 0.9287014676255008, 'mae': 0.728761072841967},
 'SlopeOne': {'rmse': 0.956009609349784, 'mae': 0.7506301922430901},
 'NMF': {'rmse': 0.9829194050880655, 'mae': 0.770839814078092},
 'NormalPredictor': {'rmse': 1.5033278955186193, 'mae': 1.203277280877059},
 'KNNBaseline': {'rmse': 0.9372792939128142, 'mae': 0.7376178692577451},
 'KNNBasic': {'rmse': 0.9934689409204525, 'mae': 0.7832968928094843},
 'KNNWithMeans': {'rmse': 0.9589818275571133, 'mae': 0.7533807882324919},
 'KNNWithZScore': {'rmse': 0.9584587067442145, 'mae': 0.7497054389355823},
 'BaselineOnly': {'rmse': 0.9441612945800022, 'mae': 0.7482578492964332},
 'CoClustering': {'rmse': 0.9742150911683711, 'mae': 0.7621701095431095}}
```

u4

In [37]:

```
df_train = pd.read_csv("data/ml-100k_merged/u4.base")
df_test = pd.read_csv("data/ml-100k_merged/u4.test")
df_train_m = df_train[df_train['sex'] == 'M']
df_test_m = df_test[df_test['sex'] == 'M']
print("Male: train {}|test {}".format(len(df_train_m), len(df_test_m)))
df_train_f = df_train[df_train['sex'] == 'F']
df_test_f = df_test[df_test['sex'] == 'F']
print("Female: train {}|test {}".format(len(df_train_f), len(df_test_f)))
```

```
Male: train 59482|test 14778
Female: train 20518|test 5222
```

In [38]:

```
data_train, data_test = build_train_test(df_train_m, df_test_m)
all_results_m = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_m[algorithm_name] = result
    print("===== ===== ===== =====")
# female
data_train, data_test = build_train_test(df_train_f, df_test_f)
all_results_f = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_f[algorithm_name] = result
    print("===== ===== ===== =====")
print(all_results_m)
print(all_results_f)
```

```
RMSE: 0.9173
MAE:  0.7234
SVD                 |0.05 mins|rmse: 0.9173|mae: 0.7234
===== ===== ===== =====
RMSE: 0.9052
MAE:  0.7103
SVDpp               |1.72 mins|rmse: 0.9052|mae: 0.7103
===== ===== ===== =====
RMSE: 0.9245
MAE:  0.7262
SlopeOne            |0.02 mins|rmse: 0.9245|mae: 0.7262
===== ===== ===== =====
RMSE: 0.9432
MAE:  0.7416
NMF                 |0.04 mins|rmse: 0.9432|mae: 0.7416
===== ===== ===== =====
RMSE: 1.4933
MAE:  1.1976
NormalPredictor     |0.00 mins|rmse: 1.4933|mae: 1.1976
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9082
MAE:  0.7155
KNNBaseline         |0.03 mins|rmse: 0.9082|mae: 0.7155
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9513
MAE:  0.7515
KNNBasic            |0.02 mins|rmse: 0.9513|mae: 0.7515
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9298
MAE:  0.7311
KNNWithMeans        |0.03 mins|rmse: 0.9298|mae: 0.7311
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9293
MAE:  0.7278
KNNWithZScore       |0.03 mins|rmse: 0.9293|mae: 0.7278
===== ===== ===== =====
Estimating biases using als...
RMSE: 0.9198
MAE:  0.7306
BaselineOnly        |0.00 mins|rmse: 0.9198|mae: 0.7306
===== ===== ===== =====
RMSE: 0.9488
MAE:  0.7408
CoClustering        |0.01 mins|rmse: 0.9488|mae: 0.7408
===== ===== ===== =====
RMSE: 1.0049
MAE:  0.7947
SVD                 |0.02 mins|rmse: 1.0049|mae: 0.7947
===== ===== ===== =====
RMSE: 0.9881
MAE:  0.7787
SVDpp               |0.56 mins|rmse: 0.9881|mae: 0.7787
===== ===== ===== =====
RMSE: 1.0429
MAE:  0.8173
SlopeOne            |0.01 mins|rmse: 1.0429|mae: 0.8173
===== ===== ===== =====
RMSE: 1.0935
MAE:  0.8635
NMF                 |0.02 mins|rmse: 1.0935|mae: 0.8635
===== ===== ===== =====
RMSE: 1.5438
MAE:  1.2387
NormalPredictor     |0.00 mins|rmse: 1.5438|mae: 1.2387
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0180
MAE:  0.8026
KNNBaseline         |0.01 mins|rmse: 1.0180|mae: 0.8026
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.1005
MAE:  0.8716
KNNBasic            |0.00 mins|rmse: 1.1005|mae: 0.8716
===== ===== ===== =====
Computing the msd similarity matrix...
```

```
Done computing similarity matrix.
RMSE: 1.0312
MAE:  0.8115
KNNWithMeans        |0.00 mins|rmse: 1.0312|mae: 0.8115
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0311
MAE:  0.8075
KNNWithZScore       |0.01 mins|rmse: 1.0311|mae: 0.8075
===== ===== ===== =====
Estimating biases using als...
RMSE: 1.0045
MAE:  0.7990
BaselineOnly        |0.00 mins|rmse: 1.0045|mae: 0.7990
===== ===== ===== =====
RMSE: 1.0776
MAE:  0.8456
CoClustering        |0.01 mins|rmse: 1.0776|mae: 0.8456
===== ===== ===== =====
{'SVD': {'rmse': 0.9172906678233979, 'mae': 0.7233813704169727, 'n_samples': 14778}, 'SVDpp': {'rmse': 0.905
2092738538707, 'mae': 0.7103355971762854, 'n_samples': 14778}, 'SlopeOne': {'rmse': 0.9245354116711205, 'ma
e': 0.7262214680414103, 'n_samples': 14778}, 'NMF': {'rmse': 0.9431936802930033, 'mae': 0.7416340780219074,
'n_samples': 14778}, 'NormalPredictor': {'rmse': 1.4933105145357937, 'mae': 1.1975968487286086, 'n_samples':
14778}, 'KNNBaseline': {'rmse': 0.9081567450286757, 'mae': 0.7154985290889513, 'n_samples': 14778}, 'KNNBasi
c': {'rmse': 0.9512636623349967, 'mae': 0.7514560549938851, 'n_samples': 14778}, 'KNNWithMeans': {'rmse': 0.
9298174791628703, 'mae': 0.7311296564396331, 'n_samples': 14778}, 'KNNWithZScore': {'rmse': 0.92926731848876
22, 'mae': 0.7278053385183868, 'n_samples': 14778}, 'BaselineOnly': {'rmse': 0.919841396561 2983, 'mae': 0.73
0550928458531, 'n_samples': 14778}, 'CoClustering': {'rmse': 0.9488090628799054, 'mae': 0.740830050946203,
'n_samples': 14778}}
{'SVD': {'rmse': 1.004948080109479, 'mae': 0.7947326588963926, 'n_samples': 5222}, 'SVDpp': {'rmse': 0.98814
35323143724, 'mae': 0.7786946613784083, 'n_samples': 5222}, 'SlopeOne': {'rmse': 1.0428726259771341, 'mae':
0.817317878490599, 'n_samples': 5222}, 'NMF': {'rmse': 1.0935359218137817, 'mae': 0.8635215727684796, 'n_sam
ples': 5222}, 'NormalPredictor': {'rmse': 1.5437778783734848, 'mae': 1.2387099307347462, 'n_samples': 5222},
'KNNBaseline': {'rmse': 1.0180309076820477, 'mae': 0.8026144720380491, 'n_samples': 5222}, 'KNNBasic': {'rms
e': 1.1005049678985608, 'mae': 0.871648058594147, 'n_samples': 5222}, 'KNNWithMeans': {'rmse': 1.03116538558
71539, 'mae': 0.8114576637106115, 'n_samples': 5222}, 'KNNWithZScore': {'rmse': 1.0310905059033708, 'mae':
0.8074582837662433, 'n_samples': 5222}, 'BaselineOnly': {'rmse': 1.004482618714441, 'mae': 0.798958484211338
1, 'n_samples': 5222}, 'CoClustering': {'rmse': 1.0775996429595047, 'mae': 0.8456003388854058, 'n_samples':
5222}}
```

```
In [39]: all_results4 = combine_mf(algorithms, all_results_m, all_results_f)
         all_results4
```

```
Out[39]: {'SVD': {'rmse': 0.9409660572795044, 'mae': 0.7420111918389493},
          'SVDpp': {'rmse': 0.9275789705027216, 'mae': 0.7281841488394597},
          'SlopeOne': {'rmse': 0.9568460687537724, 'mae': 0.7500067408096934},
          'NMF': {'rmse': 0.9846648333638338, 'mae': 0.7734589029002374},
          'NormalPredictor': {'rmse': 1.506650620991791, 'mae': 1.2083314744404112},
          'KNNBaseline': {'rmse': 0.9380871079171389, 'mae': 0.7382445017929608},
          'KNNBasic': {'rmse': 0.9923979183445015, 'mae': 0.7828381871339135},
          'KNNWithMeans': {'rmse': 0.9573149702492377, 'mae': 0.7521032991380856},
          'KNNWithZScore': {'rmse': 0.9568991009721892, 'mae': 0.7486027225226021},
          'BaselineOnly': {'rmse': 0.9426746072535154, 'mae': 0.748412141265589},
          'CoClustering': {'rmse': 0.9840635803106952, 'mae': 0.7681855731271289}}
```

u5

```
In [40]: df_train = pd.read_csv("data/ml-100k_merged/u5.base")
         df_test = pd.read_csv("data/ml-100k_merged/u5.test")
         df_train_m = df_train[df_train['sex'] == 'M']
         df_test_m = df_test[df_test['sex'] == 'M']
         print("Male: train {}|test {}".format(len(df_train_m), len(df_test_m)))
         df_train_f = df_train[df_train['sex'] == 'F']
         df_test_f = df_test[df_test['sex'] == 'F']
         print("Female: train {}|test {}".format(len(df_train_f), len(df_test_f)))
```

```
Male: train 59449|test 14811
Female: train 20551|test 5189
```

In [41]:

```
data_train, data_test = build_train_test(df_train_m, df_test_m)
all_results_m = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_m[algorithm_name] = result
    print("===== ===== ===== =====")
# female
data_train, data_test = build_train_test(df_train_f, df_test_f)
all_results_f = {}
save_model = False
for algorithm_name in algorithms.keys():
    result = train_single_algorithm(algorithm_name, data_train, data_test, save_model)
    all_results_f[algorithm_name] = result
    print("===== ===== ===== =====")
print(all_results_m)
print(all_results_f)
```

```
RMSE: 0.9172
MAE:  0.7244
SVD                 |0.05 mins|rmse: 0.9172|mae: 0.7244
===== ===== ===== =====
RMSE: 0.9001
MAE:  0.7106
SVDpp               |1.70 mins|rmse: 0.9001|mae: 0.7106
===== ===== ===== =====
RMSE: 0.9202
MAE:  0.7267
SlopeOne            |0.02 mins|rmse: 0.9202|mae: 0.7267
===== ===== ===== =====
RMSE: 0.9444
MAE:  0.7437
NMF                 |0.04 mins|rmse: 0.9444|mae: 0.7437
===== ===== ===== =====
RMSE: 1.5005
MAE:  1.2080
NormalPredictor     |0.00 mins|rmse: 1.5005|mae: 1.2080
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9099
MAE:  0.7195
KNNBaseline         |0.03 mins|rmse: 0.9099|mae: 0.7195
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9554
MAE:  0.7561
KNNBasic            |0.02 mins|rmse: 0.9554|mae: 0.7561
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9260
MAE:  0.7327
KNNWithMeans        |0.03 mins|rmse: 0.9260|mae: 0.7327
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9255
MAE:  0.7297
KNNWithZScore       |0.03 mins|rmse: 0.9255|mae: 0.7297
===== ===== ===== =====
Estimating biases using als...
RMSE: 0.9239
MAE:  0.7346
BaselineOnly        |0.00 mins|rmse: 0.9239|mae: 0.7346
===== ===== ===== =====
RMSE: 0.9473
MAE:  0.7459
CoClustering        |0.01 mins|rmse: 0.9473|mae: 0.7459
===== ===== ===== =====
RMSE: 1.0077
MAE:  0.7997
SVD                 |0.02 mins|rmse: 1.0077|mae: 0.7997
===== ===== ===== =====
RMSE: 1.0013
MAE:  0.7939
SVDpp               |0.57 mins|rmse: 1.0013|mae: 0.7939
===== ===== ===== =====
RMSE: 1.0525
MAE:  0.8306
SlopeOne            |0.01 mins|rmse: 1.0525|mae: 0.8306
===== ===== ===== =====
RMSE: 1.0827
MAE:  0.8497
NMF                 |0.02 mins|rmse: 1.0827|mae: 0.8497
===== ===== ===== =====
RMSE: 1.5559
MAE:  1.2523
NormalPredictor     |0.00 mins|rmse: 1.5559|mae: 1.2523
===== ===== ===== =====
Estimating biases using als...
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0187
MAE:  0.8102
KNNBaseline         |0.01 mins|rmse: 1.0187|mae: 0.8102
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.1082
MAE:  0.8786
KNNBasic            |0.00 mins|rmse: 1.1082|mae: 0.8786
===== ===== ===== =====
Computing the msd similarity matrix...
```

```
Done computing similarity matrix.
RMSE: 1.0420
MAE:  0.8257
KNNWithMeans        |0.00 mins|rmse: 1.0420|mae: 0.8257
===== ===== ===== =====
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 1.0436
MAE:  0.8244
KNNWithZScore       |0.01 mins|rmse: 1.0436|mae: 0.8244
===== ===== ===== =====
Estimating biases using als...
RMSE: 1.0075
MAE:  0.8093
BaselineOnly        |0.00 mins|rmse: 1.0075|mae: 0.8093
===== ===== ===== =====
RMSE: 1.0842
MAE:  0.8543
CoClustering        |0.01 mins|rmse: 1.0842|mae: 0.8543
===== ===== ===== =====
{'SVD': {'rmse': 0.9171727751979186, 'mae': 0.7244282328283458, 'n_samples': 14811}, 'SVDpp': {'rmse': 0.900
1469781384343, 'mae': 0.7106379206235095, 'n_samples': 14811}, 'SlopeOne': {'rmse': 0.9202085061743455, 'ma
e': 0.7266878977677615, 'n_samples': 14811}, 'NMF': {'rmse': 0.9443974727978148, 'mae': 0.7437182618612755,
'n_samples': 14811}, 'NormalPredictor': {'rmse': 1.5004863956546761, 'mae': 1.2080349652728086, 'n_samples':
14811}, 'KNNBaseline': {'rmse': 0.9099486815653918, 'mae': 0.7194806821405724, 'n_samples': 14811}, 'KNNBasi
c': {'rmse': 0.9554352408011128, 'mae': 0.756076652656679, 'n_samples': 14811}, 'KNNWithMeans': {'rmse': 0.9
259900992113702, 'mae': 0.7326652235146106, 'n_samples': 14811}, 'KNNWithZScore': {'rmse': 0.92552021601362
63, 'mae': 0.7297471829326554, 'n_samples': 14811}, 'BaselineOnly': {'rmse': 0.9238687161947932, 'mae': 0.734
6214555351739, 'n_samples': 14811}, 'CoClustering': {'rmse': 0.9473201711801814, 'mae': 0.7459055693966298,
'n_samples': 14811}}
{'SVD': {'rmse': 1.0076576355938345, 'mae': 0.7997297843146715, 'n_samples': 5189}, 'SVDpp': {'rmse': 1.0012
542088686402, 'mae': 0.7939473969036261, 'n_samples': 5189}, 'SlopeOne': {'rmse': 1.0524591710951536, 'mae':
0.8305588084144364, 'n_samples': 5189}, 'NMF': {'rmse': 1.082749255928933, 'mae': 0.8497156983371831, 'n_sam
ples': 5189}, 'NormalPredictor': {'rmse': 1.555890754553883, 'mae': 1.2523096343457771, 'n_samples': 5189},
'KNNBaseline': {'rmse': 1.018691476943524, 'mae': 0.8102455291107117, 'n_samples': 5189}, 'KNNBasic': {'rms
e': 1.1081750969477435, 'mae': 0.8786242718706974, 'n_samples': 5189}, 'KNNWithMeans': {'rmse': 1.0420065662
264628, 'mae': 0.8257459645469473, 'n_samples': 5189}, 'KNNWithZScore': {'rmse': 1.0436377995808914, 'mae':
0.824413108193231, 'n_samples': 5189}, 'BaselineOnly': {'rmse': 1.0074648162083228, 'mae': 0.809330319138859
7, 'n_samples': 5189}, 'CoClustering': {'rmse': 1.08421903178285, 'mae': 0.8543184723699604, 'n_samples': 51
89}}
```

In [42]:

```
all_results5 = combine_mf(algorithms, all_results_m, all_results_f)
all_results5
```

Out[42]:

```
{'SVD': {'rmse': 0.9414848857097163, 'mae': 0.743965220361473},
 'SVDpp': {'rmse': 0.927438759969451, 'mae': 0.7322525642443858},
 'SlopeOne': {'rmse': 0.956279627679877, 'mae': 0.7536372055350413},
 'NMF': {'rmse': 0.9821668741966373, 'mae': 0.7712192967549497},
 'NormalPredictor': {'rmse': 1.5150557114217416, 'mae': 1.2195220281637904},
 'KNNBaseline': {'rmse': 0.9393720996949373, 'mae': 0.7430296216869751},
 'KNNBasic': {'rmse': 0.9973133829485235, 'mae': 0.7878716324617561},
 'KNNWithMeans': {'rmse': 0.9574420572984939, 'mae': 0.7568150217754503},
 'KNNWithZScore': {'rmse': 0.9575665580146528, 'mae': 0.7543082572415117},
 'BaselineOnly': {'rmse': 0.9462674640318125, 'mae': 0.7540046701971501},
 'CoClustering': {'rmse': 0.9846687502983118, 'mae': 0.7740332970730603}}
```

看下平均水平

In [43]:

```
all_results_list = [all_results, all_results2, all_results3, all_results4, all_results5]
get_mean_results(algorithms, all_results_list)
```

```
SVD                 |rmse: 0.9481+-0.0090|mae: 0.7481+-0.0070
SVDpp               |rmse: 0.9333+-0.0071|mae: 0.7330+-0.0047
SlopeOne            |rmse: 0.9603+-0.0050|mae: 0.7533+-0.0030
NMF                 |rmse: 0.9888+-0.0069|mae: 0.7749+-0.0039
NormalPredictor     |rmse: 1.5150+-0.0091|mae: 1.2155+-0.0082
KNNBaseline         |rmse: 0.9426+-0.0056|mae: 0.7421+-0.0042
KNNBasic            |rmse: 1.0001+-0.0072|mae: 0.7893+-0.0063
KNNWithMeans        |rmse: 0.9638+-0.0075|mae: 0.7578+-0.0052
KNNWithZScore       |rmse: 0.9637+-0.0076|mae: 0.7545+-0.0052
BaselineOnly        |rmse: 0.9505+-0.0088|mae: 0.7546+-0.0069
CoClustering        |rmse: 0.9849+-0.0070|mae: 0.7703+-0.0058
```

In [44]:
```
## divide the gender
a1m = {'SVD': {'rmse': 0.9383862926313519, 'mae': 0.7396122593767015, 'n_samples': 15167}, 'SVDpp': {'rmse':
0.9194037497411428, 'mae': 0.7205192220296978, 'n_samples': 15167}, 'SlopeOne': {'rmse': 0.9412970931258932,
'mae': 0.7382574736595242, 'n_samples': 15167}, 'NMF': {'rmse': 0.9589445650742295, 'mae': 0.7523194873738819
, 'n_samples': 15167}, 'NormalPredictor': {'rmse': 1.500544126463222, 'mae': 1.205103549940071, 'n_samples':
15167}, 'KNNBaseline': {'rmse': 0.9260552076751267, 'mae': 0.7285941131595712, 'n_samples': 15167}, 'KNNBasi
c': {'rmse': 0.9717718846950745, 'mae': 0.7702955038128292, 'n_samples': 15167}, 'KNNWithMeans': {'rmse': 0.9
517069774985772, 'mae': 0.7485886155989885, 'n_samples': 15167}, 'KNNWithZScore': {'rmse': 0.9510369147183566
, 'mae': 0.7453160009398803, 'n_samples': 15167}, 'BaselineOnly': {'rmse': 0.9412313340115859, 'mae': 0.74576
37435983588, 'n_samples': 15167}, 'CoClustering': {'rmse': 0.9666239938747948, 'mae': 0.7563200760011395, 'n_
samples': 15167}}
a1f = {'SVD': {'rmse': 1.0432522116249232, 'mae': 0.8292149859524621, 'n_samples': 4833}, 'SVDpp': {'rmse':
1.0231227380577936, 'mae': 0.8050122893758805, 'n_samples': 4833}, 'SlopeOne': {'rmse': 1.0475136483527672,
'mae': 0.8221744839339707, 'n_samples': 4833}, 'NMF': {'rmse': 1.1117874487075228, 'mae': 0.8687941894725169,
'n_samples': 4833}, 'NormalPredictor': {'rmse': 1.5974751294318625, 'mae': 1.278439667711637, 'n_samples': 48
33}, 'KNNBaseline': {'rmse': 1.0286108953065138, 'mae': 0.8147285637859992, 'n_samples': 4833}, 'KNNBasic': {
'rmse': 1.1171658279411747, 'mae': 0.8913259771972365, 'n_samples': 4833}, 'KNNWithMeans': {'rmse': 1.0468459
876187062, 'mae': 0.8238796454936679, 'n_samples': 4833}, 'KNNWithZScore': {'rmse': 1.0466049872950942, 'mae'
: 0.8182216093005554, 'n_samples': 4833}, 'BaselineOnly': {'rmse': 1.0422503591682972, 'mae': 0.8350659342010
252, 'n_samples': 4833}, 'CoClustering': {'rmse': 1.08373730570858, 'mae': 0.8509535489131902, 'n_samples': 4
833}}
a2m = {'SVD': {'rmse': 0.929214385614642, 'mae': 0.733123074917991, 'n_samples': 14973}, 'SVDpp': {'rmse': 0.
9162519721092842, 'mae': 0.7182473114192324, 'n_samples': 14973}, 'SlopeOne': {'rmse': 0.9341968334849607, 'm
ae': 0.7326311434629396, 'n_samples': 14973}, 'NMF': {'rmse': 0.9601034359067921, 'mae': 0.7515852134327144,
'n_samples': 14973}, 'NormalPredictor': {'rmse': 1.4958159980910508, 'mae': 1.1990425952013646, 'n_samples':
14973}, 'KNNBaseline': {'rmse': 0.9211838414808493, 'mae': 0.7236166336955803, 'n_samples': 14973}, 'KNNBasi
c': {'rmse': 0.9638759549629315, 'mae': 0.7593535707684028, 'n_samples': 14973}, 'KNNWithMeans': {'rmse': 0.9
443556935591163, 'mae': 0.7422467060312301, 'n_samples': 14973}, 'KNNWithZScore': {'rmse': 0.9435048977367296
, 'mae': 0.7391487771938792, 'n_samples': 14973}, 'BaselineOnly': {'rmse': 0.9326465056126624, 'mae': 0.73826
10195013816, 'n_samples': 14973}, 'CoClustering': {'rmse': 0.9550367379040832, 'mae': 0.7463983587754381, 'n_
samples': 14973}}
a2f = {'SVD': {'rmse': 1.0102374409034336, 'mae': 0.7960316747134094, 'n_samples': 5027}, 'SVDpp': {'rmse':
0.9979728978746284, 'mae': 0.7847173802667002, 'n_samples': 5027}, 'SlopeOne': {'rmse': 1.0494247863106883,
'mae': 0.8165105863155546, 'n_samples': 5027}, 'NMF': {'rmse': 1.0968460543647154, 'mae': 0.8584049710206093,
'n_samples': 5027}, 'NormalPredictor': {'rmse': 1.6111828550868887, 'mae': 1.2966040255137412, 'n_samples': 5
027}, 'KNNBaseline': {'rmse': 1.0179722039033587, 'mae': 0.7975699254156436, 'n_samples': 5027}, 'KNNBasic':
{'rmse': 1.131070539525255, 'mae': 0.8935325683472519, 'n_samples': 5027}, 'KNNWithMeans': {'rmse': 1.0422633
744601377, 'mae': 0.8118404511439838, 'n_samples': 5027}, 'KNNWithZScore': {'rmse': 1.04687878615226, 'mae':
0.8101010200349981, 'n_samples': 5027}, 'BaselineOnly': {'rmse': 1.0103692953757475, 'mae': 0.804333063991447
9, 'n_samples': 5027}, 'CoClustering': {'rmse': 1.0698015765666293, 'mae': 0.8327644510625524, 'n_samples': 5
027}}

a3m = {'SVD': {'rmse': 0.9233261231769833, 'mae': 0.7275448469780914, 'n_samples': 14531}, 'SVDpp': {'rmse':
0.9070326872660632, 'mae': 0.7106518243841786, 'n_samples': 14531}, 'SlopeOne': {'rmse': 0.9287785845066504,
'mae': 0.7281195512351074, 'n_samples': 14531}, 'NMF': {'rmse': 0.949608800092403, 'mae': 0.7456066252387595,
'n_samples': 14531}, 'NormalPredictor': {'rmse': 1.4929410528707299, 'mae': 1.1973588148679182, 'n_samples':
14531}, 'KNNBaseline': {'rmse': 0.9134066622905466, 'mae': 0.717614963794572, 'n_samples': 14531}, 'KNNBasic'
: {'rmse': 0.9563849612453938, 'mae': 0.7538440337160236, 'n_samples': 14531}, 'KNNWithMeans': {'rmse': 0.935
5248189122948, 'mae': 0.7346419580941771, 'n_samples': 14531}, 'KNNWithZScore': {'rmse': 0.9353783518376002,
'mae': 0.7315822823043118, 'n_samples': 14531}, 'BaselineOnly': {'rmse': 0.9261895489354907, 'mae': 0.7318161
621461988, 'n_samples': 14531}, 'CoClustering': {'rmse': 0.9411583951257683, 'mae': 0.7363249726861354, 'n_sa
mples': 14531}}
a3f = {'SVD': {'rmse': 0.9934823456336317, 'mae': 0.7884181637938653, 'n_samples': 5469}, 'SVDpp': {'rmse':
0.9839593050442988, 'mae': 0.7768769055975204, 'n_samples': 5469}, 'SlopeOne': {'rmse': 1.024852618894576, 'm
ae': 0.8104404179675362, 'n_samples': 5469}, 'NMF': {'rmse': 1.0663816713331462, 'mae': 0.837883783181098, 'n
_samples': 5469}, 'NormalPredictor': {'rmse': 1.5305830779069212, 'mae': 1.219002501132833, 'n_samples': 5469
}, 'KNNBaseline': {'rmse': 0.9979376921402456, 'mae': 0.7907651026249727, 'n_samples': 5469}, 'KNNBasic': {'r
mse': 1.0858646527432234, 'mae': 0.8615524231600198, 'n_samples': 5469}, 'KNNWithMeans': {'rmse': 1.018685774
3462245, 'mae': 0.8031694042024777, 'n_samples': 5469}, 'KNNWithZScore': {'rmse': 1.0172416850976695, 'mae':
0.7978582253698466, 'n_samples': 5469}, 'BaselineOnly': {'rmse': 0.9903285927085751, 'mae': 0.791943012211052
9, 'n_samples': 5469}, 'CoClustering': {'rmse': 1.05703543046311, 'mae': 0.8308400096470937, 'n_samples': 546
9}}

a4m = {'SVD': {'rmse': 0.9172906678233979, 'mae': 0.7233813704169727, 'n_samples': 14778}, 'SVDpp': {'rmse':
0.9052092738538707, 'mae': 0.7103355971762854, 'n_samples': 14778}, 'SlopeOne': {'rmse': 0.9245354116711205,
'mae': 0.7262214680414103, 'n_samples': 14778}, 'NMF': {'rmse': 0.9431936802930033, 'mae': 0.7416340780219074
, 'n_samples': 14778}, 'NormalPredictor': {'rmse': 1.4933105145357937, 'mae': 1.1975968487286086, 'n_samples'
: 14778}, 'KNNBaseline': {'rmse': 0.9081567450286757, 'mae': 0.7154985290889513, 'n_samples': 14778}, 'KNNBas
ic': {'rmse': 0.9512636623349967, 'mae': 0.7514560549938851, 'n_samples': 14778}, 'KNNWithMeans': {'rmse': 0.
929817479162870, 'mae': 0.7311296564396331, 'n_samples': 14778}, 'KNNWithZScore': {'rmse': 0.929267318488762
2, 'mae': 0.7278053385183868, 'n_samples': 14778}, 'BaselineOnly': {'rmse': 0.9198413965612983, 'mae': 0.7305
50928458531, 'n_samples': 14778}, 'CoClustering': {'rmse': 0.9488090628799054, 'mae': 0.740830050946203, 'n_s
amples': 14778}}
a4f = {'SVD': {'rmse': 1.004948080109479, 'mae': 0.7947326588963926, 'n_samples': 5222}, 'SVDpp': {'rmse': 0.
9881435323143724, 'mae': 0.7786946613784083, 'n_samples': 5222}, 'SlopeOne': {'rmse': 1.0428726259771341, 'ma
e': 0.817317878490599, 'n_samples': 5222}, 'NMF': {'rmse': 1.0935359218137817, 'mae': 0.8635215727684796, 'n_
samples': 5222}, 'NormalPredictor': {'rmse': 1.5437778783734848, 'mae': 1.2387099307347462, 'n_samples': 5222
}, 'KNNBaseline': {'rmse': 1.0180309076820477, 'mae': 0.8026144720380491, 'n_samples': 5222}, 'KNNBasic': {'r
mse': 1.1005049678985608, 'mae': 0.871648058594147, 'n_samples': 5222}, 'KNNWithMeans': {'rmse': 1.0311653855
871539, 'mae': 0.8114576637106115, 'n_samples': 5222}, 'KNNWithZScore': {'rmse': 1.0310905059033708, 'mae':
0.8074582837662433, 'n_samples': 5222}, 'BaselineOnly': {'rmse': 1.004482618714441, 'mae': 0.7989584842113381
, 'n_samples': 5222}, 'CoClustering': {'rmse': 1.0775996429595047, 'mae': 0.8456003388854058, 'n_samples': 52
22}}

a5m = {'SVD': {'rmse': 0.9171727751979186, 'mae': 0.7244282328283458, 'n_samples': 14811}, 'SVDpp': {'rmse':
0.9001469781384343, 'mae': 0.7106379206235095, 'n_samples': 14811}, 'SlopeOne': {'rmse': 0.9202085061743455,
'mae': 0.7266878977677615, 'n_samples': 14811}, 'NMF': {'rmse': 0.9443974727978148, 'mae': 0.7437182618612755
, 'n_samples': 14811}, 'NormalPredictor': {'rmse': 1.5004863956546761, 'mae': 1.2080349652728086, 'n_samples'
: 14811}, 'KNNBaseline': {'rmse': 0.9099486815653918, 'mae': 0.7194806821405724, 'n_samples': 14811}, 'KNNBas
ic': {'rmse': 0.9554352408011128, 'mae': 0.756076652656679, 'n_samples': 14811}, 'KNNWithMeans': {'rmse': 0.9
259900992113702, 'mae': 0.7326652235146106, 'n_samples': 14811}, 'KNNWithZScore': {'rmse': 0.9255202160136263
, 'mae': 0.7297471829326554, 'n_samples': 14811}, 'BaselineOnly': {'rmse': 0.9238687161947932, 'mae': 0.73462
14555351739, 'n_samples': 14811}, 'CoClustering': {'rmse': 0.9473201711801814, 'mae': 0.7459055693966298, 'n_
samples': 14811}}
a5f = {'SVD': {'rmse': 1.0076576355938345, 'mae': 0.7997297843146715, 'n_samples': 5189}, 'SVDpp': {'rmse':
1.0012542088686402, 'mae': 0.7939473969036261, 'n_samples': 5189}, 'SlopeOne': {'rmse': 1.0524591710951536,
'mae': 0.8305588084144364, 'n_samples': 5189}, 'NMF': {'rmse': 1.082749255928933, 'mae': 0.8497156983371831,
'n_samples': 5189}, 'NormalPredictor': {'rmse': 1.555890754553883, 'mae': 1.2523096343457771, 'n_samples': 51
89}, 'KNNBaseline': {'rmse': 1.018691476943524, 'mae': 0.8102455291107117, 'n_samples': 5189}, 'KNNBasic': {
'rmse': 1.1081750969477435, 'mae': 0.8786242718706974, 'n_samples': 5189}, 'KNNWithMeans': {'rmse': 1.0420065
662264628, 'mae': 0.8257459645469473, 'n_samples': 5189}, 'KNNWithZScore': {'rmse': 1.0436377995808914, 'mae'
: 0.824413108193231, 'n_samples': 5189}, 'BaselineOnly': {'rmse': 1.0074648162083228, 'mae': 0.80933031913885
97, 'n_samples': 5189}, 'CoClustering': {'rmse': 1.08421903178285, 'mae': 0.8543184723699604, 'n_samples': 51
89}}
```

In [45]:

```
# 仅考虑男性
all_results_list = [a1m, a2m, a3m, a4m, a5m]
get_mean_results(algorithms, all_results_list)
```

```
SVD                  |rmse: 0.9251+-0.0080|mae: 0.7296+-0.0060
SVDpp                |rmse: 0.9096+-0.0071|mae: 0.7141+-0.0044
SlopeOne             |rmse: 0.9298+-0.0074|mae: 0.7304+-0.0045
NMF                  |rmse: 0.9512+-0.0071|mae: 0.7470+-0.0043
NormalPredictor      |rmse: 1.4966+-0.0033|mae: 1.2014+-0.0043
KNNBaseline          |rmse: 0.9158+-0.0068|mae: 0.7210+-0.0047
KNNBasic             |rmse: 0.9597+-0.0073|mae: 0.7582+-0.0066
KNNWithMeans         |rmse: 0.9375+-0.0094|mae: 0.7379+-0.0066
KNNWithZScore        |rmse: 0.9369+-0.0093|mae: 0.7347+-0.0065
BaselineOnly         |rmse: 0.9288+-0.0075|mae: 0.7362+-0.0055
CoClustering         |rmse: 0.9518+-0.0086|mae: 0.7452+-0.0067
```

In [46]:

```
# 仅考虑女性
all_results_list = [a1f, a2f, a3f, a4f, a5f]
get_mean_results(algorithms, all_results_list)
```

```
SVD                  |rmse: 1.0119+-0.0167|mae: 0.8016+-0.0143
SVDpp                |rmse: 0.9989+-0.0137|mae: 0.7878+-0.0104
SlopeOne             |rmse: 1.0434+-0.0098|mae: 0.8194+-0.0067
NMF                  |rmse: 1.0903+-0.0151|mae: 0.8557+-0.0109
NormalPredictor      |rmse: 1.5678+-0.0312|mae: 1.2570+-0.0277
KNNBaseline          |rmse: 1.0162+-0.0100|mae: 0.8032+-0.0086
KNNBasic             |rmse: 1.1086+-0.0152|mae: 0.8793+-0.0120
KNNWithMeans         |rmse: 1.0362+-0.0102|mae: 0.8152+-0.0084
KNNWithZScore        |rmse: 1.0371+-0.0115|mae: 0.8116+-0.0091
BaselineOnly         |rmse: 1.0110+-0.0171|mae: 0.8079+-0.0147
CoClustering         |rmse: 1.0745+-0.0102|mae: 0.8429+-0.0095
```

In [ ]:

目前为止关于ml-100k性别的分析差不多到这里, 可以发现ml-100k中可能性别并不是一个好的分割方式

- 注意这里对比的是v2_2_ml
- 可能是用户数量过少导致的
- 接下来尝试下ml-1m