Final_Netflix_Movie_Recommendation

December 12, 2022

[1]:
```
from google.colab import drive
```

[2]:
```
drive.mount('/content/drive/')
```

```
Mounted at /content/drive/
```

[3]:
```
!ln -s /content/gdrive/MyDrive/ /mydrive
!ls /mydrive
```

```
/mydrive
```

[4]:
```
!ls
```

```
drive  sample_data
```

[5]:
```
%cd ..
```

```
/
```

[6]:
```
!ls
```

```
bin      dev   lib32  mydrive                    python-apt  srv    usr
boot     etc   lib64  NGC-DL-CONTAINER-LICENSE   root        sys    var
content  home  media  opt                        run         tmp
datalab  lib   mnt    proc                       sbin        tools
```

[7]:
```
%cd /content/drive/MyDrive/Colab Notebooks/Netflix_Movie_Recommendation
```

```
/content/drive/MyDrive/Colab Notebooks/Netflix_Movie_Recommendation
```

[ ]:
```
!pip install stellargraph
```

[12]:
```
import matplotlib.pyplot as plt
from math import isclose
from sklearn.decomposition import PCA
from sklearn import preprocessing, feature_extraction, model_selection
import os
import networkx as nx
```

```
import numpy as np
import pandas as pd
from stellargraph import StellarGraph, datasets
from stellargraph.data import EdgeSplitter
from collections import Counter
import multiprocessing
from IPython.display import display, HTML
from sklearn.model_selection import train_test_split

%matplotlib inline
```

[ ]:
```
!ls
```

[ ]:
```
!unzip data_folder/archive.zip
```

[ ]:
```
start = datetime.now()
if not os.path.isfile('data.csv'):
    # Creating a file 'data.csv' before reading it
    # Read all the files in netflix and store them in one big file('data.csv')
    # We re reading from each of the four files and appendig each rating to a global file 'train.csv'
    data = open('data.csv', mode='w')

    row = list()
    files=['data_folder/combined_data_1.txt','data_folder/combined_data_2.txt',
           'data_folder/combined_data_3.txt', 'data_folder/combined_data_4.txt']

    for file in files:

        print("Reading ratings from {}...".format(file))

        with open(file) as f:
            for line in f:
                del row[:] # We might not have to do this.
                line = line.strip()
                if line.endswith(':'):
                    # All below are ratings for this movie, until another movie appears.
                    movie_id = line.replace(':', '')
                else:
                    row = [x for x in line.split(',')]
                    row.insert(0, movie_id)
                    data.write(','.join(row))
                    data.write('\n')
        print("Done.\n")

    data.close()
```

```
print('Time taken :', datetime.now() - start)
```

[13]:
```
print("creating the dataframe from data.csv file..")
df = pd.read_csv('data.csv', sep=',', names=['movie', 'user','rating','date'])
df.date = pd.to_datetime(df.date)
print('Done.\n')

# we are arranging the ratings according to time.
print('Sorting the dataframe by date..')
df.sort_values(by='date', inplace=True)
print('Done..')
```

```
creating the dataframe from data.csv file..
Done.

Sorting the dataframe by date..
Done..
```

[14]:
```
start_user = 6
end_user   = 100000
mask = (df['user'] >= start_user) & (df['user'] <= end_user)
df2 = df.loc[mask]
```

[17]:
```
df2
```

[17]:

	movie	user	rating	date
0	11405	28966	4	1999-12-30
1	15532	28966	1	1999-12-30
2	8903	28966	1	1999-12-30
3	10809	1086	4	1999-12-31
4	829	1086	3	1999-12-31
…	…	…	…	…
3737134	5939	62030	1	2005-12-31
3737135	2873	23421	3	2005-12-31
3737136	6615	23421	5	2005-12-31
3737137	9745	3321	2	2005-12-31
3737138	6844	96575	3	2005-12-31

[3737139 rows x 4 columns]

[16]:
```
df2 = df2.reset_index(drop=True)
```

[ ]:
```
df2_explicit=df2[['movie','user','rating']]
df2_explicit['movie'] = 'm' + df2_explicit['movie'].astype(str)
df2_explicit['user'] = 'u' + df2_explicit['user'].astype(str)
df2_explicit
```

[19]:
```
df2_implicit=df2[['movie','user']]
df2_implicit['movie'] = 'm' + df2_implicit['movie'].astype(str)
df2_implicit['user'] = 'u' + df2_implicit['user'].astype(str)
df2_implicit.columns = ['source', 'target']
```

```
<ipython-input-19-094e0f3edad7>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df2_implicit['movie'] = 'm' + df2_implicit['movie'].astype(str)
<ipython-input-19-094e0f3edad7>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df2_implicit['user'] = 'u' + df2_implicit['user'].astype(str)
```

[20]:
```
df2_implicit
```

[20]:

	source	target
0	m11405	u28966
1	m15532	u28966
2	m8903	u28966
3	m10809	u1086
4	m829	u1086
…	…	…
3737134	m5939	u62030
3737135	m2873	u23421
3737136	m6615	u23421
3737137	m9745	u3321
3737138	m6844	u96575

[3737139 rows x 2 columns]

[21]:
```
G = StellarGraph(edges=df2_implicit)
print(G.info())
```

```
StellarGraph: Undirected multigraph
 Nodes: 35770, Edges: 3737139

 Node types:
  default: [35770]
    Features: none
    Edge types: default-default->default
```

```
 Edge types:
    default-default->default: [3737139]
        Weights: all 1 (default)
        Features: none
```

[22]:
```
# Define an edge splitter on the original graph:
edge_splitter_test = EdgeSplitter(G)

# Randomly sample a fraction p=0.2 of all positive links, and same number of␣
 ↪negative links, from graph, and obtain the
# reduced graph graph_test with the sampled links removed:
graph_test, examples_test, labels_test = edge_splitter_test.train_test_split(
    p=0.2, method="global", keep_connected = True
)

print(graph_test.info())
```

```
** Sampled 747427 positive and 747427 negative edges. **
StellarGraph: Undirected multigraph
 Nodes: 35770, Edges: 2989712

 Node types:
  default: [35770]
    Features: none
    Edge types: default-default->default

 Edge types:
    default-default->default: [2989712]
        Weights: all 1 (default)
        Features: none
```

[23]:
```
373713*2
```

[23]:
```
747426
```

[24]:
```
len(examples_test)
```

[24]:
```
1494854
```

[25]:
```
3363426-3027084
```

[25]:
```
336342
```

[26]:
```
# Do the same process to compute a training subset from within the test graph
edge_splitter_train = EdgeSplitter(graph_test, G)
graph_train, examples, labels = edge_splitter_train.train_test_split(
    p=0.2, method="global", keep_connected = True
```

```
)
(
    examples_train,
    examples_model_selection,
    labels_train,
    labels_model_selection,
) = train_test_split(examples, labels, train_size=0.8, test_size=0.2)

print(graph_train.info())
```

```
** Sampled 597942 positive and 597942 negative edges. **
StellarGraph: Undirected multigraph
 Nodes: 35770, Edges: 2391770

 Node types:
  default: [35770]
    Features: none
    Edge types: default-default->default

 Edge types:
    default-default->default: [2391770]
        Weights: all 1 (default)
        Features: none
```

[27]:
```
examples_model_selection
```

[27]:
```
array([['m187', 'u70328'],
       ['u39831', 'u28213'],
       ['m6386', 'u83805'],
       …,
       ['m6187', 'u37451'],
       ['m8954', 'u90742'],
       ['m8117', 'u42345']], dtype=object)
```

[28]:
```
labels_model_selection
```

[28]:
```
array([1, 0, 1, …, 0, 1, 1])
```

[29]:
```
336342*2
```

[29]:
```
672684
```

[30]:
```
len(examples)
```

[30]:
```
1195884
```

[31]:
```
len(examples_train)
```

[31]: 956707

1 Deep Walk

[32]:
```
dimensions = 32
num_walks = 10
walk_length = 40
window_size = 5
num_iter = 1
workers = multiprocessing.cpu_count()
```

[33]:
```
from gensim.models import Word2Vec
from stellargraph.data import UniformRandomWalk

def deepWalk_embedding_train(graph, name):
    rw = UniformRandomWalk(graph)
    walks = rw.run(graph.nodes(), n=num_walks, length=walk_length)
    print(f"Number of random walks for '{name}': {len(walks)}")

    model = Word2Vec(
        walks,
        size=dimensions,
        window=window_size,
        min_count=0,
        sg=1,
        workers=workers,
        iter=num_iter,
    )

    model.save("deepWalk_embedding_train.model")
    def get_embedding(u):
        return model.wv[u]

    return get_embedding

def deepWalk_embedding_test(graph, name):
    rw = UniformRandomWalk(graph)
    walks = rw.run(graph.nodes(), n=num_walks, length=walk_length)
    print(f"Number of random walks for '{name}': {len(walks)}")

    model = Word2Vec(
        walks,
        size=dimensions,
        window=window_size,
        min_count=0,
```

```
        sg=1,
        workers=workers,
        iter=num_iter,
    )

    model.save("deepWalk_embedding_test.model")
    def get_embedding(u):
        return model.wv[u]

    return get_embedding
```

[34]:
```
embedding_train_deepW = deepWalk_embedding_train(graph_train, "Train Graph")
```

```
Number of random walks for 'Train Graph': 357700
```

[35]:
```
embedding_train_deepW('u18702')
```

[35]:
```
array([ 0.32066625,  0.20943502,  0.11771876, -0.5377317 ,  0.10620353,
        0.02973612, -0.3471631 ,  0.01199626,  0.04199905, -0.03668234,
       -0.15197623, -0.13650276, -0.10623594,  0.0763692 , -0.0252196 ,
       -0.25288713, -0.66952896,  0.26896286, -0.02152866, -0.10220387,
       -0.558724  , -0.97082484, -0.06930302,  0.1850806 , -0.3388855 ,
       -0.01407279, -0.3689696 ,  0.15004921, -0.04440913, -0.5713346 ,
       -0.01487237, -0.5684083 ], dtype=float32)
```

[36]:
```
embedding_test_deepW = deepWalk_embedding_test(graph_test, "Test Graph")
```

```
Number of random walks for 'Test Graph': 357700
```

2 Implicit DW

[37]:
```
from keras.models import Sequential
from keras.layers import Dense
from tensorflow import keras

model_concat_dw = Sequential()

model_concat_dw.add(Dense(units=64, input_dim=64, kernel_initializer='normal',␣
 ↪activation='relu'))

model_concat_dw.add(Dense(units=32, kernel_initializer='normal',␣
 ↪activation='tanh'))

model_concat_dw.add(Dense(units=16, kernel_initializer='normal',␣
 ↪activation='tanh'))
```

```
model_concat_dw.add(Dense(units=8, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_dw.add(Dense(units=4, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_dw.add(Dense(1, kernel_initializer='normal'))

opt = keras.optimizers.Adam(learning_rate=0.01)

model_concat_dw.compile(loss='mean_squared_error', optimizer='adam')
```

[38]:
```
def train_model_concatdw(
    link_examples, link_labels, get_embedding, binary_operator
):
    link_features = link_examples_to_features(
        link_examples, get_embedding, binary_operator
    )
    model_concat_dw.fit(np.array(link_features), np.array(link_labels),␣
↪batch_size = 50, epochs = 50, verbose=1)
    return model_concat_dw
```

[39]:
```
def operator_concat(u, v):
    return np.concatenate((u, v))
def link_examples_to_features(link_examples, transform_node, binary_operator):
    return [
        binary_operator(transform_node(src), transform_node(dst))
        for src, dst in link_examples
    ]
```

[40]:
```
clf_dw_concat = train_model_concatdw(
        examples_train, labels_train, embedding_train_deepW, operator_concat
    )
```

```
Epoch 1/50
19135/19135 [==============================] - 42s 2ms/step - loss: 0.0367
Epoch 2/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0321
Epoch 3/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0315
Epoch 4/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0311
Epoch 5/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0306
Epoch 6/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0303
```

```
Epoch 7/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0301
Epoch 8/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0299
Epoch 9/50
19135/19135 [==============================] - 43s 2ms/step - loss: 0.0298
Epoch 10/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0296
Epoch 11/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0295
Epoch 12/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0294
Epoch 13/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0293
Epoch 14/50
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0293
Epoch 15/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0292
Epoch 16/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0291
Epoch 17/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0290
Epoch 18/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0290
Epoch 19/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0289
Epoch 20/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0288
Epoch 21/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0288
Epoch 22/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0287
Epoch 23/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0287
Epoch 24/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0286
Epoch 25/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0285
Epoch 26/50
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0285
Epoch 27/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0284
Epoch 28/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0284
Epoch 29/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0283
Epoch 30/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0283
```

```
Epoch 31/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0282
Epoch 32/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0282
Epoch 33/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0281
Epoch 34/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0281
Epoch 35/50
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0281
Epoch 36/50
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0280
Epoch 37/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0280
Epoch 38/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0280
Epoch 39/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0279
Epoch 40/50
19135/19135 [==============================] - 42s 2ms/step - loss: 0.0279
Epoch 41/50
19135/19135 [==============================] - 42s 2ms/step - loss: 0.0279
Epoch 42/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0278
Epoch 43/50
19135/19135 [==============================] - 41s 2ms/step - loss: 0.0278
Epoch 44/50
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0278
Epoch 45/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0278
Epoch 46/50
19135/19135 [==============================] - 42s 2ms/step - loss: 0.0277
Epoch 47/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0277
Epoch 48/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0277
Epoch 49/50
19135/19135 [==============================] - 42s 2ms/step - loss: 0.0276
Epoch 50/50
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0276
```

[41]:
```
clf_dw_concat.save('clf_dw_concat.model')
# from tensorflow import keras
# model = keras.models.load_model('clf_dw_concat.model')
```

[42]:
```
link_features_selection_dw = link_examples_to_features(
        examples_model_selection, embedding_train_deepW, operator_concat)
```

[43]:
```
predicted_dw = clf_dw_concat.predict(np.array(link_features_selection_dw))
```

```
7475/7475 [==============================] - 10s 1ms/step
```

[44]:
```
print(predicted_dw)
```

```
[[0.98886985]
 [0.00173074]
 [0.9897951 ]
 …
 [0.19212204]
 [0.9894196 ]
 [0.8924854 ]]
```

[45]:
```
predicted_dw=predicted_dw.flatten()
```

[46]:
```
loss_dw_con=np.sum((labels_model_selection-predicted_dw)**2)/len(predicted_dw)
loss_dw_con
```

[46]:
```
0.029017725830540318
```

[47]:
```
def get_label(i):
  if i>=0.5:
    i=1
  else:
    i=0
  return i
```

[48]:
```
predicted_label_concat_dw=np.array([get_label(i) for i in predicted_dw])
accuracy_concat_dw=1-np.
 ↪sum(abs(labels_model_selection-predicted_label_concat_dw))/len(predicted_dw)
accuracy_concat_dw
```

[48]:
```
0.9623793257712907
```

3 Test our DeepWalk model with implicit feedback

[49]:
```
movie_unique_dw=df2_implicit.source.unique()
array_true_im_testdw=[]
for i in range(len(labels_test)):
  if labels_test[i]==1:
    array_true_im_testdw.append(examples_test[i])

from collections import defaultdict
import json
```

```
def create_user_movie_dict_dw(array_true):

    # if not os.path.isfile('user_movie_train.json'):
    user_movie = defaultdict(list)
    for i in range(len(array_true)):
        user_movie[array_true[i][1]].append(array_true[i][0])
        # my_json = json.dumps(user_movie)
        # f = open("user_movie_train.json","w")
        # f.write(my_json)
        # f.close()
    # else:
    #     print("opening json file")
    #     with open('user_movie_train.json') as json_file:
    #       user_movie = json.load(json_file)
    # print('user_movie_train.json loaded')
    return user_movie

dict_dw_im_test=create_user_movie_dict_dw(array_true_im_testdw)

array_true_user_im=list(dict_dw_im_test.keys())

import random
test_implicit={}
for i in range(len(array_true_user_im)):
  negative_example=[]
  negative_example.append(dict_dw_im_test[array_true_user_im[i]][0])
  # print(negative_example)
  j=0
  movie_watched=dict_dw_im_test[array_true_user_im[i]]
  while(j<99):
    movie_picked=movie_unique_dw[random.randint(0, 17674)]
    if movie_picked in movie_watched:
      continue
    else:
      negative_example.append(movie_picked)
      j+=1
  test_implicit[array_true_user_im[i]]=negative_example
```

[ ]:
```
test_implicit
```

[51]:
```
examples_train
```

[51]:
```
array([['m14296', 'u88118'],
       ['m15339', 'u70392'],
       ['m8325', 'm9594'],
       …,
       ['m3860', 'u30218'],
```

```
       ['u42620', 'm11797'],
       ['m3282', 'u55445']], dtype=object)
```

[52]:
```
b=[['u77970', 'u97877'],
       ['m5760', 'u60561']]
b.append(['m5760', 'u60561'])
print(b)
```

```
[['u77970', 'u97877'], ['m5760', 'u60561'], ['m5760', 'u60561']]
```

[ ]:
```
HR=0
ndcg=0
HR_F=0
ndcg_F=0
for i in list(test_implicit.keys()):
  test=[]

  negatives=test_implicit[i]
  # print(negatives)
  for j in negatives:
    a=[]
    a.append(i)
    for z in range(1):
      a.append(j)
    #print(a)
    test.append(a)
  #test=np.array(test)
  #print(test)
  test_features=link_examples_to_features(np.array(test), embedding_test_deepW,␣
↪operator_concat)
  #print(test_features)
  predicted_scores=clf_dw_concat.predict(np.array(test_features))
  predicted_scores=np.array(predicted_scores.flatten())
  predicted_scores=np.abs(predicted_scores)
  #print(predicted_scores)
  #print(type(predicted_scores))
  true_score=predicted_scores[0]
  #print(true_score)
  predicted_scores=np.sort(predicted_scores)

  predicted_scores=predicted_scores[::-1]
  #print(predicted_scores)
  #print(predicted_scores[:10])
  if true_score in predicted_scores[:10]:
    HR+=1

    index = np.where(predicted_scores[:10] == true_score)
```

```
    #print(index)
    ndcg += np.reciprocal(np.log2(index[0][0]+2))

  # print(HR)
HR_F=HR/len(list(test_implicit.keys()))
ndcg_F=ndcg/len(list(test_implicit.keys()))
```

[3]:
```
print(HR_F)
print(ndcg_F)
```

```
0.48312628451333334
0.4621215123489127
```

4 Explicit Dw

5 build model

[56]:
```
from tensorflow import keras

model_concat_ex_dw = Sequential()

model_concat_ex_dw.add(Dense(units=64, input_dim=64,␣
↪kernel_initializer='normal', activation='relu'))

model_concat_ex_dw.add(Dense(units=32, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex_dw.add(Dense(units=16, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex_dw.add(Dense(units=8, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex_dw.add(Dense(units=4, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex_dw.add(Dense(1, kernel_initializer='normal'))

opt = keras.optimizers.Adam(learning_rate=0.01)

model_concat_ex_dw.compile(loss='mean_squared_error', optimizer='adam')
```

[57]:
```
def train_model_concat_ex_dw(
    link_examples, link_labels, get_embedding, binary_operator
):
```

```
    link_features = link_examples_to_features(
        link_examples, get_embedding, binary_operator
    )
    model_concat_ex_dw.fit(np.array(link_features), np.array(link_labels),␣
↪batch_size = 50, epochs = 40, verbose=1)
    return model_concat_ex_dw
```

[58]:
```
array_true_ex_train=[]
for i in range(len(labels_train)):
  if labels_train[i]==1:
    array_true_ex_train.append(examples_train[i])

from collections import defaultdict
import json
def create_user_movie_dict_df2ex(df):

    # if not os.path.isfile('user_movie_train.json'):
    user_movie = defaultdict(dict)
    for iter, row in df.iterrows():

        dict1={}
        dict1=user_movie[row[1]].copy()
        dict1[row[0]]=row[2]
        user_movie[row[1]]=dict1
    #     my_json = json.dumps(user_movie)
    #     f = open("user_movie_train.json","w")
    #     f.write(my_json)
    #     f.close()
    # else:
    #     print("opening json file")
    #     with open('user_movie_train.json') as json_file:
    #       user_movie = json.load(json_file)
    # print('user_movie_train.json loaded')
    return user_movie

df2_explicit_dict=create_user_movie_dict_df2ex(df2_explicit)

ratings_train=[]
for i in range(len(array_true_ex_train)):
  movie=array_true_ex_train[i][0]
  user=array_true_ex_train[i][1]
  ratings_train.append(df2_explicit_dict[user][movie])
len(ratings_train)

array_true_ex_select=[]
for i in range(len(labels_model_selection)):
  if labels_model_selection[i]==1:
```

```
    array_true_ex_select.append(examples_model_selection[i])

ratings_selection=[]
for i in range(len(array_true_ex_select)):
  movie=array_true_ex_select[i][0]
  user=array_true_ex_select[i][1]
  ratings_selection.append(df2_explicit_dict[user][movie])
len(ratings_selection)
```

[58]:
```
119661
```

[59]:
```
clf_concat_ex_dw = train_model_concat_ex_dw(
        array_true_ex_train, ratings_train, embedding_train_deepW,␣
 ↪operator_concat
    )
```

```
Epoch 1/40
9566/9566 [==============================] - 21s 2ms/step - loss: 1.4644
Epoch 2/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9975
Epoch 3/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9773
Epoch 4/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9693
Epoch 5/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9649
Epoch 6/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9616
Epoch 7/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9586
Epoch 8/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9566
Epoch 9/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9547
Epoch 10/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9530
Epoch 11/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9516
Epoch 12/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9501
Epoch 13/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9495
Epoch 14/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9483
Epoch 15/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9475
Epoch 16/40
```

```
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9471
Epoch 17/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9464
Epoch 18/40
9566/9566 [==============================] - 22s 2ms/step - loss: 0.9456
Epoch 19/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9450
Epoch 20/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9446
Epoch 21/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9439
Epoch 22/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9438
Epoch 23/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9434
Epoch 24/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9431
Epoch 25/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9426
Epoch 26/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9420
Epoch 27/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9421
Epoch 28/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9416
Epoch 29/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9412
Epoch 30/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9411
Epoch 31/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9405
Epoch 32/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9404
Epoch 33/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9401
Epoch 34/40
9566/9566 [==============================] - 21s 2ms/step - loss: 0.9401
Epoch 35/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9398
Epoch 36/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9399
Epoch 37/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9396
Epoch 38/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9393
Epoch 39/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9391
Epoch 40/40
```

```
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9391
```

[60]:
```
clf_concat_ex_dw.save('clf_concat_ex_dw.model')
# from tensorflow import keras
# model = keras.models.load_model('model_concat_ex_dw.model')
```

[61]:
```
array_true_ex_test=[]
for i in range(len(labels_test)):
  if labels_test[i]==1:
    array_true_ex_test.append(examples_test[i])

ratings_test=[]
for i in range(len(array_true_ex_test)):
  movie=array_true_ex_test[i][0]
  user=array_true_ex_test[i][1]
  ratings_test.append(df2_explicit_dict[user][movie])
len(ratings_test)

link_features_test_ex = link_examples_to_features(
        array_true_ex_test, embedding_test_deepW, operator_concat)
predicted_concat_test_ex = clf_concat_ex_dw.predict(np.
 ↪array(link_features_test_ex))
predicted_concat_test_ex=predicted_concat_test_ex.flatten()

loss_concat_test_ex=np.sum((ratings_test-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex)
loss_concat_test_ex

RMSE=np.sqrt(np.sum((np.array(ratings_test)-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex))
RMSE

MAPE=np.sum(np.abs((np.array(ratings_test)-predicted_concat_test_ex))/
 ↪ratings_test)/len(predicted_concat_test_ex)
MAPE
```

```
23358/23358 [==============================] - 28s 1ms/step
```

[61]:
```
0.40126858761523726
```

[62]:
```
loss_concat_test_ex
```

[62]:
```
1.4459614105866245
```

[63]:
```
RMSE
```

[63]:
```
1.2024813556087364
```

[64]:
```
MAPE
```

[64]:
```
0.40126858761523726
```

6 Node2Vec

[65]:
```
p = 1.0
q = 0.5
dimensions = 32
num_walks = 10
walk_length = 40
window_size = 5
num_iter = 1
workers = multiprocessing.cpu_count()
```

[66]:
```
from stellargraph.data import BiasedRandomWalk
from gensim.models import Word2Vec


def node2vec_embedding_train(graph, name):
    rw = BiasedRandomWalk(graph)
    walks = rw.run(graph.nodes(), n=num_walks, length=walk_length, p=p, q=q)
    print(f"Number of random walks for '{name}': {len(walks)}")

    model = Word2Vec(
        walks,
        size=dimensions,
        window=window_size,
        min_count=0,
        sg=1,
        workers=workers,
        iter=num_iter,
    )

    model.save("node2vec_embedding_train.model")
    def get_embedding(u):
        return model.wv[u]

    return get_embedding

def node2vec_embedding_test(graph, name):
    rw = BiasedRandomWalk(graph)
    walks = rw.run(graph.nodes(), n=num_walks, length=walk_length, p=p, q=q)
    print(f"Number of random walks for '{name}': {len(walks)}")
```

```
    model = Word2Vec(
        walks,
        size=dimensions,
        window=window_size,
        min_count=0,
        sg=1,
        workers=workers,
        iter=num_iter,
    )

    model.save("node2vec_embedding_test.model")
    def get_embedding(u):
        return model.wv[u]

    return get_embedding
```

[67]:
```
embedding_train = node2vec_embedding_train(graph_train, "Train Graph")
```

```
Number of random walks for 'Train Graph': 357700
```

[68]:
```
embedding_test = node2vec_embedding_test(graph_test, "Test Graph")
```

```
Number of random walks for 'Test Graph': 357700
```

[69]:
```
embedding_train('u54902')
```

[69]:
```
array([ 0.23925376,  0.40896732,  0.35928875, -0.26472804,  0.3561669 ,
       -0.0639363 , -0.35883102, -0.46279582, -0.02292539,  0.22478591,
       -0.18858702, -0.26382217,  0.21360822,  0.08878147, -0.3409715 ,
       -0.2615404 , -0.14342514, -0.0294617 ,  0.09424245, -0.43551356,
       -0.16138169, -0.8301645 ,  0.11483356,  0.07110158, -0.8059631 ,
        0.01564322, -0.33640248, -0.04425808, -0.32322735, -0.67067045,
       -0.00175937, -0.5483301 ], dtype=float32)
```

7 Node2Vec implicit

[70]:
```
def operator_l1(u, v):
    return np.abs(u - v)

def operator_concat(u, v):
    return np.concatenate((u, v))

def link_examples_to_features(link_examples, transform_node, binary_operator):
    return [
        binary_operator(transform_node(src), transform_node(dst))
```

```
        for src, dst in link_examples
    ]
```

[71]:
```
from keras.models import Sequential
from keras.layers import Dense
from tensorflow import keras

model_concat = Sequential()

model_concat.add(Dense(units=64, input_dim=64, kernel_initializer='normal',␣
↪activation='relu'))

model_concat.add(Dense(units=32, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat.add(Dense(units=16, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat.add(Dense(units=8, kernel_initializer='normal', activation='tanh'))

model_concat.add(Dense(units=4, kernel_initializer='normal', activation='tanh'))

model_concat.add(Dense(1, kernel_initializer='normal'))

model_concat.compile(loss='mean_squared_error', optimizer='adam')
```

[72]:
```
model_l1 = Sequential()

model_l1.add(Dense(units=64, input_dim=32, kernel_initializer='normal',␣
↪activation='relu'))

model_l1.add(Dense(units=32, kernel_initializer='normal', activation='tanh'))

model_l1.add(Dense(units=16, kernel_initializer='normal', activation='tanh'))

model_l1.add(Dense(units=8, kernel_initializer='normal', activation='tanh'))

model_l1.add(Dense(units=4, kernel_initializer='normal', activation='tanh'))

model_l1.add(Dense(1, kernel_initializer='normal'))

model_l1.compile(loss='mean_squared_error', optimizer='adam')
```

[73]:
```
def train_model_l1(
    link_examples, link_labels, get_embedding, binary_operator
):
    print('1')
```

```
    link_features = link_examples_to_features(
        link_examples, get_embedding, binary_operator
    )
    print(link_features[0])
    print(np.array(link_features)[0])
    print(link_labels.shape)
    print('2')
    model_l1.fit(np.array(link_features), np.array(link_labels), batch_size =␣
↪50, epochs = 20, verbose=1)
    print('3')
    return model_l1

def train_model_concat(
    link_examples, link_labels, get_embedding, binary_operator
):
    link_features = link_examples_to_features(
        link_examples, get_embedding, binary_operator
    )
    model_concat.fit(np.array(link_features), np.array(link_labels), batch_size␣
↪= 50, epochs = 20, verbose=1)
    return model_concat
```

[74]:
```
len(examples_train)
```

[74]:
```
956707
```

[75]:
```
labels_train
```

[75]:
```
array([1, 1, 0, …, 1, 0, 1])
```

[76]:
```
examples_train[:15]
```

[76]:
```
array([['m14296', 'u88118'],
       ['m15339', 'u70392'],
       ['m8325', 'm9594'],
       ['m4892', 'm8756'],
       ['m9028', 'u45538'],
       ['m14283', 'm16492'],
       ['u39606', 'u14924'],
       ['m8905', 'u80290'],
       ['m4640', 'u31985'],
       ['m14171', 'm7800'],
       ['m1440', 'u25075'],
       ['m8832', 'u98977'],
       ['m7617', 'u2807'],
       ['u97046', 'u65078'],
       ['m12845', 'm15985']], dtype=object)
```

8 Operator L1

[77]:
```
clf_l1 = train_model_l1(
        examples_train, labels_train, embedding_train, operator_l1
    )
```

```
1
[0.4116749  0.17243585 0.24843372 0.3366567  0.29452354 0.52074116
 0.5503845  0.03506192 0.22441533 0.07882293 0.33449212 0.0445069
 0.19600487 0.44226813 0.55260766 0.26600304 0.4072196  0.03820133
 0.01790228 0.17435056 0.47702584 0.12407637 0.45112982 0.03233886
 0.26577783 0.09833881 0.3821273  0.21264076 0.12191042 0.0495812
 0.16194168 0.22479443]
[0.4116749  0.17243585 0.24843372 0.3366567  0.29452354 0.52074116
 0.5503845  0.03506192 0.22441533 0.07882293 0.33449212 0.0445069
 0.19600487 0.44226813 0.55260766 0.26600304 0.4072196  0.03820133
 0.01790228 0.17435056 0.47702584 0.12407637 0.45112982 0.03233886
 0.26577783 0.09833881 0.3821273  0.21264076 0.12191042 0.0495812
 0.16194168 0.22479443]
(956707,)
2
Epoch 1/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0746
Epoch 2/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0660
Epoch 3/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0639
Epoch 4/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0629
Epoch 5/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0622
Epoch 6/20
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0618
Epoch 7/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0613
Epoch 8/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0611
Epoch 9/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0608
Epoch 10/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0604
Epoch 11/20
19135/19135 [==============================] - 36s 2ms/step - loss: 0.0602
Epoch 12/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0600
Epoch 13/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0597
```

```
Epoch 14/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0596
Epoch 15/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0594
Epoch 16/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0592
Epoch 17/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0590
Epoch 18/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0589
Epoch 19/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0587
Epoch 20/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0586
3
```

[78]:
```
clf_l1.save('clf_l1.model')
# from tensorflow import keras
# model = keras.models.load_model('clf_l1.model')
```

[79]:
```
link_features_test1 = link_examples_to_features(
        examples_model_selection, embedding_train, operator_l1)
```

[80]:
```
type(link_features_test1[0])
```

[80]:
```
numpy.ndarray
```

[81]:
```
labels_model_selection
```

[81]:
```
array([1, 0, 1, …, 0, 1, 1])
```

[82]:
```
predicted = clf_l1.predict(np.array(link_features_test1))
```

```
7475/7475 [==============================] - 9s 1ms/step
```

[83]:
```
predicted=predicted.flatten()
```

[84]:
```
len(predicted)
```

[84]:
```
239177
```

[85]:
```
len(labels_model_selection)
```

[85]:
```
239177
```

[86]:
```
loss_l1=np.sum((labels_model_selection-predicted)**2)/len(predicted)
loss_l1
```

[86]:
```
0.059104245894771305
```

[87]:
```
def get_label(i):
  if i>=0.5:
    i=1
  else:
    i=0
  return i
```

[88]:
```
predicted_label=np.array([get_label(i) for i in predicted])
```

[89]:
```
predicted_label
```

[89]:
```
array([1, 0, 1, …, 0, 1, 1])
```

[90]:
```
labels_model_selection
```

[90]:
```
array([1, 0, 1, …, 0, 1, 1])
```

[91]:
```
accuracy_l1=1-np.sum(abs(labels_model_selection-predicted_label))/len(predicted)
accuracy_l1
```

[91]:
```
0.9223671172395339
```

9 Operator Concat

[92]:
```
clf_concat = train_model_concat(
        examples_train, labels_train, embedding_train, operator_concat
    )
```

```
Epoch 1/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0362
Epoch 2/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0310
Epoch 3/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0304
Epoch 4/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0299
Epoch 5/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0296
Epoch 6/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0294
Epoch 7/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0292
Epoch 8/20
```

```
19135/19135 [==============================] - 40s 2ms/step - loss: 0.0290
Epoch 9/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0289
Epoch 10/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0288
Epoch 11/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0287
Epoch 12/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0286
Epoch 13/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0286
Epoch 14/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0285
Epoch 15/20
19135/19135 [==============================] - 37s 2ms/step - loss: 0.0284
Epoch 16/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0283
Epoch 17/20
19135/19135 [==============================] - 38s 2ms/step - loss: 0.0283
Epoch 18/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0282
Epoch 19/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0281
Epoch 20/20
19135/19135 [==============================] - 39s 2ms/step - loss: 0.0281
```

[93]:
```
clf_concat.save('clf_concat.model')
# from tensorflow import keras
# clf_concat = keras.models.load_model('clf_concat.model')
```

[94]:
```
link_features_test2 = link_examples_to_features(
        examples_model_selection, embedding_train, operator_concat)
```

[95]:
```
predicted_concat = clf_concat.predict(np.array(link_features_test2))
predicted_concat=predicted_concat.flatten()
```

```
7475/7475 [==============================] - 9s 1ms/step
```

[96]:
```
loss_concat=np.sum((labels_model_selection-predicted_concat)**2)/
 ↪len(predicted_concat)
loss_concat
```

[96]:
```
0.029102988450533673
```

[97]:
```
labels_model_selection[:10]
```

[97]:
```
array([1, 0, 1, 1, 1, 0, 0, 0, 0, 0])
```

[98]:
```
predicted_concat[:10]
```

[98]:
```
array([ 9.9859428e-01, -6.3061714e-05,  9.9928582e-01,  9.2774773e-01,
        9.8056495e-01,  9.6511841e-04,  1.4388561e-03,  2.8230494e-01,
        5.3220987e-04,  1.6236305e-04], dtype=float32)
```

[99]:
```
predicted_label_concat=np.array([get_label(i) for i in predicted_concat])
accuracy_concat=1-np.sum(abs(labels_model_selection-predicted_label_concat))/
 ↪len(predicted_concat)
accuracy_concat
```

[99]:
```
0.961756356171371
```

10 test our model, node2vec implicit

[100]:
```
movie_unique=df2_implicit.source.unique()
array_true_im_test=[]
for i in range(len(labels_test)):
  if labels_test[i]==1:
    array_true_im_test.append(examples_test[i])

from collections import defaultdict
import json
def create_user_movie_dict_array_true(array_true):

    # if not os.path.isfile('user_movie_train.json'):
    user_movie = defaultdict(list)
    for i in range(len(array_true)):
        user_movie[array_true[i][1]].append(array_true[i][0])
        # my_json = json.dumps(user_movie)
        # f = open("user_movie_train.json","w")
        # f.write(my_json)
        # f.close()
    # else:
    #     print("opening json file")
    #     with open('user_movie_train.json') as json_file:
    #       user_movie = json.load(json_file)
    # print('user_movie_train.json loaded')
    return user_movie

dict_array_true_im_test=create_user_movie_dict_array_true(array_true_im_test)

array_true_user_im=list(dict_array_true_im_test.keys())

import random
```

```
test_implicit={}
for i in range(len(array_true_user_im)):
  negative_example=[]
  negative_example.append(dict_array_true_im_test[array_true_user_im[i]][0])
  # print(negative_example)
  j=0
  movie_watched=dict_array_true_im_test[array_true_user_im[i]]
  while(j<99):
    movie_picked=movie_unique[random.randint(0, 17674)]
    if movie_picked in movie_watched:
      continue
    else:
      negative_example.append(movie_picked)
      j+=1
  test_implicit[array_true_user_im[i]]=negative_example
```

[101]:
```
dict_array_true_im_test['u42128']
```

[101]:
```
['m2698',
 'm8569',
 'm16265',
 'm1722',
 'm9003',
 'm5924',
 'm13007',
 'm7430',
 'm6756',
 'm4490',
 'm4883',
 'm17261',
 'm7330',
 'm4033',
 'm11701',
 'm2808',
 'm3787',
 'm2916',
 'm13974',
 'm15156',
 'm9593',
 'm5277',
 'm5087',
 'm459',
 'm7917',
 'm5187',
 'm831',
 'm17097',
 'm12442',
```

```
'm13423',
'm9032',
'm11209',
'm6076',
'm11186',
'm15296',
'm6206',
'm9350',
'm4577',
'm8317',
'm6302',
'm10013',
'm1901',
'm13023',
'm10255',
'm5997',
'm15758',
'm10504',
'm3825',
'm14233',
'm13546',
'm14480',
'm16260',
'm12527',
'm1700',
'm14660',
'm5788',
'm9960',
'm10889',
'm12497',
'm550',
'm8904',
'm17474',
'm30',
'm12671',
'm7047',
'm14943',
'm11446',
'm11271',
'm191',
'm12425',
'm9617',
'm12359',
'm10947',
'm14467',
'm5414',
'm6255',
```

```
'm918',
'm6366',
'm13959',
'm6574',
'm16344',
'm15062',
'm4267',
'm11717',
'm10921',
'm6231',
'm15788',
'm16008',
'm13129',
'm11198',
'm6974',
'm6158',
'm10820',
'm15124',
'm8989',
'm14928',
'm5695',
'm11702',
'm8857',
'm7401',
'm6497',
'm8079',
'm5477',
'm3312',
'm17769',
'm8596',
'm16163',
'm8388',
'm5710',
'm7691',
'm13594',
'm13927',
'm4432',
'm15861',
'm17387',
'm15381',
'm443',
'm3563',
'm14873',
'm3364',
'm3253',
'm10730',
'm1877',
```

```
'm4465',
'm17303',
'm6920',
'm6989',
'm299',
'm8827',
'm17431',
'm8016',
'm11222',
'm14855',
'm4919',
'm8470',
'm7057',
'm1324',
'm15064',
'm5991',
'm3223',
'm3355',
'm4660',
'm12778',
'm10239',
'm11149',
'm4185',
'm3254',
'm16793',
'm6287',
'm16567',
'm4627',
'm1770',
'm8644',
'm12828',
'm13593',
'm2122',
'm6874',
'm12184',
'm9037',
'm14618',
'm1905',
'm12732',
'm6057',
'm7434',
'm16960',
'm5292',
'm9681',
'm16439',
'm11443',
'm2212',
```

```
'm2395',
'm7060',
'm12125',
'm3385',
'm14212',
'm6872',
'm11881',
'm13177',
'm3226',
'm11164',
'm13534',
'm11172',
'm7071',
'm2890',
'm14999',
'm16606',
'm12244',
'm11630',
'm11914',
'm1526',
'm357',
'm13081',
'm8221',
'm14185',
'm2920',
'm11490',
'm16640',
'm3860',
'm348',
'm13359',
'm7854',
'm14364',
'm10271',
'm6669',
'm12676',
'm2782',
'm11875',
'm2942',
'm4315',
'm16669',
'm6615',
'm10747',
'm16313',
'm15689',
'm241',
'm2743',
'm17324',
```

```
 'm12090',
 'm12570',
 'm10739',
 'm7828',
 'm8976',
 'm2152',
 'm1044',
 'm3605',
 'm12913',
 'm4951',
 'm10655',
 'm16765']
```

[102]:
```
len(test_implicit['u42128'])
```

[102]: 100

11 L1

[ ]:
```
HR=0
ndcg=0
for i in list(test_implicit.keys()):
  test=[]

  negatives=test_implicit[i]
  # print(negatives)
  for j in negatives:
    a=[]
    a.append(i)
    for z in range(1):
      a.append(j)
    #print(a)
    test.append(a)
  #test=np.array(test)
  #print(test)
  test_features=link_examples_to_features(np.array(test), embedding_test_deepW,␣
↪operator_l1)
  #print(test_features)
  predicted_scores=clf_l1.predict(np.array(test_features))
  predicted_scores=np.array(predicted_scores.flatten())
  predicted_scores=np.abs(predicted_scores)
  #print(predicted_scores)
  #print(type(predicted_scores))
  true_score=predicted_scores[0]
  #print(true_score)
```

```
  predicted_scores=np.sort(predicted_scores)

  predicted_scores=predicted_scores[::-1]
  #print(predicted_scores)
  #print(predicted_scores[:10])
  if true_score in predicted_scores[:10]:
    HR+=1

    index = np.where(predicted_scores[:10] == true_score)
    #print(index)
    ndcg += np.reciprocal(np.log2(index[0][0]+2))

  # print(HR)
HR_F=HR/len(list(test_implicit.keys()))
ndcg_F=ndcg/len(list(test_implicit.keys()))
print(HR_F)
print(ndcg_F)
```

[104]:
```
print(HR_F)
print(ndcg_F)
```

```
0.40456852791878173
0.20708756915196785
```

12 Concat

[ ]:
```
HR=0
ndcg=0
for i in list(test_implicit.keys()):
  test=[]

  negatives=test_implicit[i]
  # print(negatives)
  for j in negatives:
    a=[]
    a.append(i)
    for z in range(1):
      a.append(j)
    #print(a)
    test.append(a)
  #test=np.array(test)
  #print(test)
  test_features=link_examples_to_features(np.array(test), embedding_test_deepW,␣
↪operator_concat)
  #print(test_features)
  predicted_scores=clf_concat.predict(np.array(test_features))
```

```
  predicted_scores=np.array(predicted_scores.flatten())
  predicted_scores=np.abs(predicted_scores)
  #print(predicted_scores)
  #print(type(predicted_scores))
  true_score=predicted_scores[0]
  #print(true_score)
  predicted_scores=np.sort(predicted_scores)

  predicted_scores=predicted_scores[::-1]
  #print(predicted_scores)
  #print(predicted_scores[:10])
  if true_score in predicted_scores[:10]:
    HR+=1

    index = np.where(predicted_scores[:10] == true_score)
    #print(index)
    ndcg += np.reciprocal(np.log2(index[0][0]+2))

  # print(HR)
HR_F=HR/len(list(test_implicit.keys()))
ndcg_F=ndcg/len(list(test_implicit.keys()))
print(HR_F)
print(ndcg_F)
```

[106]:
```
print(HR_F)
print(ndcg_F)
```

```
0.5145516074450085
0.33373923137959716
```

[107]:
```
0.49412694827196746
0.4720186086826375
```

[107]: 0.4720186086826375

13 explicit feedback

14 build model

[108]:
```
from tensorflow import keras

model_concat_ex = Sequential()

model_concat_ex.add(Dense(units=64, input_dim=64, kernel_initializer='normal',␣
 ↪activation='relu'))
```

```
model_concat_ex.add(Dense(units=32, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex.add(Dense(units=16, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex.add(Dense(units=8, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex.add(Dense(units=4, kernel_initializer='normal',␣
↪activation='tanh'))

model_concat_ex.add(Dense(1, kernel_initializer='normal'))

opt = keras.optimizers.Adam(learning_rate=0.01)

model_concat_ex.compile(loss='mean_squared_error', optimizer='adam')
```

[109]:
```
def train_model_concat_ex(
    link_examples, link_labels, get_embedding, binary_operator
):
    link_features = link_examples_to_features(
        link_examples, get_embedding, binary_operator
    )
    model_concat.fit(np.array(link_features), np.array(link_labels), batch_size␣
↪= 50, epochs = 40, verbose=1)
    return model_concat
```

[110]:
```
from tensorflow import keras

model_l1_ex = Sequential()

model_l1_ex.add(Dense(units=64, input_dim=32, kernel_initializer='normal',␣
↪activation='relu'))

model_l1_ex.add(Dense(units=32, kernel_initializer='normal', activation='tanh'))

model_l1_ex.add(Dense(units=16, kernel_initializer='normal', activation='tanh'))

model_l1_ex.add(Dense(units=8, kernel_initializer='normal', activation='tanh'))

model_l1_ex.add(Dense(units=4, kernel_initializer='normal', activation='tanh'))

model_l1_ex.add(Dense(1, kernel_initializer='normal'))
```

```
opt = keras.optimizers.Adam(learning_rate=0.01)

model_l1_ex.compile(loss='mean_squared_error', optimizer='adam')
```

[111]:
```
def train_model_l1_ex(
    link_examples, link_labels, get_embedding, binary_operator
):
    link_features = link_examples_to_features(
        link_examples, get_embedding, binary_operator
    )
    model_l1_ex.fit(np.array(link_features), np.array(link_labels), batch_size
 ↪= 50, epochs = 40, verbose=1)
    return model_l1_ex
```

15 Prepare training dataset

[112]:
```
len(examples_train)
```

[112]: 956707

[113]:
```
len(labels_train)
```

[113]: 956707

[114]:
```
array_true_ex_train=[]
for i in range(len(labels_train)):
  if labels_train[i]==1:
    array_true_ex_train.append(examples_train[i])

from collections import defaultdict
import json
def create_user_movie_dict_df2ex(df):

    # if not os.path.isfile('user_movie_train.json'):
    user_movie = defaultdict(dict)
    for iter, row in df.iterrows():

        dict1={}
        dict1=user_movie[row[1]].copy()
        dict1[row[0]]=row[2]
        user_movie[row[1]]=dict1
    #     my_json = json.dumps(user_movie)
    #     f = open("user_movie_train.json","w")
    #     f.write(my_json)
    #     f.close()
```

```
    # else:
    #     print("opening json file")
    #     with open('user_movie_train.json') as json_file:
    #       user_movie = json.load(json_file)
    # print('user_movie_train.json loaded')
    return user_movie

df2_explicit_dict=create_user_movie_dict_df2ex(df2_explicit)

ratings_train=[]
for i in range(len(array_true_ex_train)):
  movie=array_true_ex_train[i][0]
  user=array_true_ex_train[i][1]
  ratings_train.append(df2_explicit_dict[user][movie])
len(ratings_train)

array_true_ex_select=[]
for i in range(len(labels_model_selection)):
  if labels_model_selection[i]==1:
    array_true_ex_select.append(examples_model_selection[i])

ratings_selection=[]
for i in range(len(array_true_ex_select)):
  movie=array_true_ex_select[i][0]
  user=array_true_ex_select[i][1]
  ratings_selection.append(df2_explicit_dict[user][movie])
len(ratings_selection)
```

[114]:
```
119661
```

[115]:
```
array_true_ex_train[0]
```

[115]:
```
array(['m14296', 'u88118'], dtype=object)
```

[116]:
```
array_true_ex_train
```

[116]:
```
[array(['m14296', 'u88118'], dtype=object),
 array(['m15339', 'u70392'], dtype=object),
 array(['m9028', 'u45538'], dtype=object),
 array(['m4640', 'u31985'], dtype=object),
 array(['m8832', 'u98977'], dtype=object),
 array(['m7617', 'u2807'], dtype=object),
 array(['m937', 'u89994'], dtype=object),
 array(['m14473', 'u92339'], dtype=object),
 array(['m15568', 'u77457'], dtype=object),
 array(['m14712', 'u21722'], dtype=object),
 array(['m443', 'u12525'], dtype=object),
```

```
array(['m13436', 'u25354'], dtype=object),
array(['m3254', 'u42072'], dtype=object),
array(['m2152', 'u10897'], dtype=object),
array(['m15739', 'u9424'], dtype=object),
array(['m5571', 'u89759'], dtype=object),
array(['m5239', 'u49803'], dtype=object),
array(['m12911', 'u27310'], dtype=object),
array(['m16879', 'u28277'], dtype=object),
array(['m17355', 'u89855'], dtype=object),
array(['m5633', 'u34201'], dtype=object),
array(['m13648', 'u46011'], dtype=object),
array(['m12292', 'u78456'], dtype=object),
array(['m6859', 'u52263'], dtype=object),
array(['m12280', 'u26592'], dtype=object),
array(['m5515', 'u32668'], dtype=object),
array(['m17321', 'u10574'], dtype=object),
array(['m7110', 'u23074'], dtype=object),
array(['m5601', 'u46846'], dtype=object),
array(['m7067', 'u5917'], dtype=object),
array(['m14103', 'u43298'], dtype=object),
array(['m11468', 'u89551'], dtype=object),
array(['m13955', 'u80753'], dtype=object),
array(['m11277', 'u20774'], dtype=object),
array(['m14412', 'u65027'], dtype=object),
array(['m2580', 'u75255'], dtype=object),
array(['m16535', 'u39242'], dtype=object),
array(['m15035', 'u42585'], dtype=object),
array(['m12476', 'u75086'], dtype=object),
array(['m12305', 'u4706'], dtype=object),
array(['m13580', 'u93429'], dtype=object),
array(['m8107', 'u40422'], dtype=object),
array(['m11313', 'u91940'], dtype=object),
array(['m7240', 'u89995'], dtype=object),
array(['m471', 'u42487'], dtype=object),
array(['m13728', 'u24536'], dtype=object),
array(['m3926', 'u71743'], dtype=object),
array(['m4356', 'u97578'], dtype=object),
array(['m758', 'u70658'], dtype=object),
array(['m16377', 'u48961'], dtype=object),
array(['m2874', 'u75337'], dtype=object),
array(['m708', 'u34128'], dtype=object),
array(['m6475', 'u67687'], dtype=object),
array(['m4472', 'u27979'], dtype=object),
array(['m2016', 'u4247'], dtype=object),
array(['m9381', 'u30978'], dtype=object),
array(['m11077', 'u84316'], dtype=object),
array(['m4393', 'u37600'], dtype=object),
```

```
array(['m15496', 'u70911'], dtype=object),
array(['m30', 'u76249'], dtype=object),
array(['m4956', 'u62769'], dtype=object),
array(['m10436', 'u27334'], dtype=object),
array(['m1773', 'u75436'], dtype=object),
array(['m12232', 'u94686'], dtype=object),
array(['m10094', 'u26210'], dtype=object),
array(['m15205', 'u42019'], dtype=object),
array(['m11077', 'u18293'], dtype=object),
array(['m6972', 'u5916'], dtype=object),
array(['m5187', 'u54426'], dtype=object),
array(['m1145', 'u78896'], dtype=object),
array(['m12244', 'u23573'], dtype=object),
array(['m8904', 'u15550'], dtype=object),
array(['m9617', 'u35511'], dtype=object),
array(['m6771', 'u28258'], dtype=object),
array(['m14538', 'u83227'], dtype=object),
array(['m14780', 'u14440'], dtype=object),
array(['m1017', 'u93889'], dtype=object),
array(['m12232', 'u4374'], dtype=object),
array(['m6196', 'u29185'], dtype=object),
array(['m6366', 'u53763'], dtype=object),
array(['m14185', 'u61829'], dtype=object),
array(['m1799', 'u85594'], dtype=object),
array(['m12500', 'u87770'], dtype=object),
array(['m15922', 'u21133'], dtype=object),
array(['m3917', 'u77418'], dtype=object),
array(['m5112', 'u97535'], dtype=object),
array(['m16707', 'u83926'], dtype=object),
array(['m6972', 'u75853'], dtype=object),
array(['m17189', 'u41179'], dtype=object),
array(['m1300', 'u1527'], dtype=object),
array(['m12512', 'u83992'], dtype=object),
array(['m8387', 'u39427'], dtype=object),
array(['m8640', 'u97199'], dtype=object),
array(['m2913', 'u81835'], dtype=object),
array(['m17251', 'u56522'], dtype=object),
array(['m6206', 'u99939'], dtype=object),
array(['m2326', 'u10947'], dtype=object),
array(['m3161', 'u25572'], dtype=object),
array(['m12881', 'u73649'], dtype=object),
array(['m7624', 'u17699'], dtype=object),
array(['m1180', 'u49281'], dtype=object),
array(['m5385', 'u84858'], dtype=object),
array(['m12926', 'u79224'], dtype=object),
array(['m4640', 'u57160'], dtype=object),
array(['m6336', 'u49890'], dtype=object),
```

```
array(['m15034', 'u26210'], dtype=object),
array(['m1425', 'u50400'], dtype=object),
array(['m8736', 'u60973'], dtype=object),
array(['m16793', 'u16107'], dtype=object),
array(['m14961', 'u10485'], dtype=object),
array(['m7749', 'u7'], dtype=object),
array(['m13081', 'u23042'], dtype=object),
array(['m14538', 'u17176'], dtype=object),
array(['m270', 'u64311'], dtype=object),
array(['m3148', 'u22964'], dtype=object),
array(['m2342', 'u94189'], dtype=object),
array(['m14574', 'u38647'], dtype=object),
array(['m3675', 'u66326'], dtype=object),
array(['m14250', 'u1457'], dtype=object),
array(['m3864', 'u60157'], dtype=object),
array(['m11325', 'u63786'], dtype=object),
array(['m3938', 'u67315'], dtype=object),
array(['m672', 'u32885'], dtype=object),
array(['m12513', 'u52008'], dtype=object),
array(['m17203', 'u53936'], dtype=object),
array(['m14313', 'u51927'], dtype=object),
array(['m16954', 'u74441'], dtype=object),
array(['m14302', 'u27002'], dtype=object),
array(['m862', 'u1534'], dtype=object),
array(['m10785', 'u60106'], dtype=object),
array(['m14240', 'u72207'], dtype=object),
array(['m788', 'u89390'], dtype=object),
array(['m724', 'u91803'], dtype=object),
array(['m7040', 'u56790'], dtype=object),
array(['m17152', 'u69026'], dtype=object),
array(['m15436', 'u47409'], dtype=object),
array(['m13981', 'u43064'], dtype=object),
array(['m15282', 'u40563'], dtype=object),
array(['m12870', 'u22058'], dtype=object),
array(['m6352', 'u37213'], dtype=object),
array(['m14198', 'u99865'], dtype=object),
array(['m1200', 'u80683'], dtype=object),
array(['m13466', 'u38464'], dtype=object),
array(['m305', 'u39896'], dtype=object),
array(['m12562', 'u18856'], dtype=object),
array(['m8171', 'u72530'], dtype=object),
array(['m6221', 'u59681'], dtype=object),
array(['m10830', 'u8382'], dtype=object),
array(['m270', 'u3743'], dtype=object),
array(['m1073', 'u77898'], dtype=object),
array(['m1470', 'u50669'], dtype=object),
array(['m6978', 'u82509'], dtype=object),
```

```
array(['m6359', 'u5727'], dtype=object),
array(['m15758', 'u12168'], dtype=object),
array(['m6221', 'u12649'], dtype=object),
array(['m6070', 'u85915'], dtype=object),
array(['m14538', 'u93532'], dtype=object),
array(['m9526', 'u41065'], dtype=object),
array(['m12280', 'u26487'], dtype=object),
array(['m5732', 'u79209'], dtype=object),
array(['m5025', 'u11875'], dtype=object),
array(['m483', 'u23354'], dtype=object),
array(['m1307', 'u47163'], dtype=object),
array(['m11781', 'u7714'], dtype=object),
array(['m12034', 'u81690'], dtype=object),
array(['m16449', 'u24679'], dtype=object),
array(['m3917', 'u14831'], dtype=object),
array(['m4805', 'u41422'], dtype=object),
array(['m7356', 'u17119'], dtype=object),
array(['m7757', 'u6841'], dtype=object),
array(['m9645', 'u60237'], dtype=object),
array(['m9645', 'u37883'], dtype=object),
array(['m6094', 'u14714'], dtype=object),
array(['m12470', 'u80657'], dtype=object),
array(['m14602', 'u27901'], dtype=object),
array(['m13622', 'u29815'], dtype=object),
array(['m16793', 'u24294'], dtype=object),
array(['m15345', 'u60617'], dtype=object),
array(['m5528', 'u49459'], dtype=object),
array(['m14016', 'u42930'], dtype=object),
array(['m4996', 'u30858'], dtype=object),
array(['m10168', 'u45051'], dtype=object),
array(['m14550', 'u80346'], dtype=object),
array(['m11077', 'u32421'], dtype=object),
array(['m9481', 'u39967'], dtype=object),
array(['m9201', 'u11765'], dtype=object),
array(['m3239', 'u44840'], dtype=object),
array(['m4033', 'u1730'], dtype=object),
array(['m14999', 'u98586'], dtype=object),
array(['m16272', 'u36978'], dtype=object),
array(['m17149', 'u69090'], dtype=object),
array(['m3513', 'u28995'], dtype=object),
array(['m2372', 'u74730'], dtype=object),
array(['m353', 'u59218'], dtype=object),
array(['m7032', 'u80055'], dtype=object),
array(['m13636', 'u51927'], dtype=object),
array(['m16668', 'u78286'], dtype=object),
array(['m17178', 'u32461'], dtype=object),
array(['m3364', 'u67547'], dtype=object),
```

```
array(['m758', 'u65834'], dtype=object),
array(['m2139', 'u78922'], dtype=object),
array(['m8320', 'u35994'], dtype=object),
array(['m299', 'u79799'], dtype=object),
array(['m2884', 'u3184'], dtype=object),
array(['m10038', 'u69065'], dtype=object),
array(['m15474', 'u93798'], dtype=object),
array(['m13923', 'u99954'], dtype=object),
array(['m9303', 'u73441'], dtype=object),
array(['m15496', 'u71393'], dtype=object),
array(['m11613', 'u27434'], dtype=object),
array(['m12911', 'u20504'], dtype=object),
array(['m10550', 'u735'], dtype=object),
array(['m3208', 'u79257'], dtype=object),
array(['m11215', 'u78250'], dtype=object),
array(['m1518', 'u12207'], dtype=object),
array(['m8177', 'u99865'], dtype=object),
array(['m1148', 'u61322'], dtype=object),
array(['m12317', 'u66921'], dtype=object),
array(['m17526', 'u93695'], dtype=object),
array(['m8531', 'u79967'], dtype=object),
array(['m636', 'u33540'], dtype=object),
array(['m4683', 'u70327'], dtype=object),
array(['m4384', 'u22994'], dtype=object),
array(['m15545', 'u11540'], dtype=object),
array(['m10282', 'u68979'], dtype=object),
array(['m12785', 'u88380'], dtype=object),
array(['m1590', 'u33396'], dtype=object),
array(['m16344', 'u49949'], dtype=object),
array(['m851', 'u71005'], dtype=object),
array(['m4590', 'u51296'], dtype=object),
array(['m9368', 'u7903'], dtype=object),
array(['m6484', 'u33780'], dtype=object),
array(['m17697', 'u91479'], dtype=object),
array(['m14741', 'u4421'], dtype=object),
array(['m9947', 'u47312'], dtype=object),
array(['m6336', 'u52191'], dtype=object),
array(['m10168', 'u68267'], dtype=object),
array(['m108', 'u94487'], dtype=object),
array(['m7786', 'u66540'], dtype=object),
array(['m4656', 'u87160'], dtype=object),
array(['m6923', 'u61765'], dtype=object),
array(['m7627', 'u59876'], dtype=object),
array(['m12232', 'u94170'], dtype=object),
array(['m3138', 'u70153'], dtype=object),
array(['m7057', 'u80542'], dtype=object),
array(['m5472', 'u8095'], dtype=object),
```

```
array(['m11745', 'u32885'], dtype=object),
array(['m11658', 'u37461'], dtype=object),
array(['m11812', 'u5109'], dtype=object),
array(['m9644', 'u72541'], dtype=object),
array(['m17293', 'u13781'], dtype=object),
array(['m15818', 'u36598'], dtype=object),
array(['m16384', 'u98394'], dtype=object),
array(['m1877', 'u9450'], dtype=object),
array(['m5401', 'u71911'], dtype=object),
array(['m14367', 'u22842'], dtype=object),
array(['m11182', 'u95209'], dtype=object),
array(['m413', 'u22785'], dtype=object),
array(['m16242', 'u59913'], dtype=object),
array(['m14149', 'u35483'], dtype=object),
array(['m5154', 'u83540'], dtype=object),
array(['m8181', 'u15104'], dtype=object),
array(['m15181', 'u4739'], dtype=object),
array(['m7624', 'u89282'], dtype=object),
array(['m12408', 'u90832'], dtype=object),
array(['m8525', 'u89240'], dtype=object),
array(['m13129', 'u63103'], dtype=object),
array(['m6475', 'u27281'], dtype=object),
array(['m4266', 'u93704'], dtype=object),
array(['m14137', 'u73103'], dtype=object),
array(['m17023', 'u40190'], dtype=object),
array(['m17157', 'u52896'], dtype=object),
array(['m1202', 'u56298'], dtype=object),
array(['m299', 'u26049'], dtype=object),
array(['m6266', 'u50081'], dtype=object),
array(['m10835', 'u81287'], dtype=object),
array(['m361', 'u7394'], dtype=object),
array(['m13302', 'u11430'], dtype=object),
array(['m9242', 'u10016'], dtype=object),
array(['m2779', 'u81663'], dtype=object),
array(['m6745', 'u50888'], dtype=object),
array(['m5903', 'u25825'], dtype=object),
array(['m8537', 'u32885'], dtype=object),
array(['m14621', 'u40666'], dtype=object),
array(['m15058', 'u13227'], dtype=object),
array(['m1202', 'u6663'], dtype=object),
array(['m14149', 'u25623'], dtype=object),
array(['m5582', 'u85237'], dtype=object),
array(['m15233', 'u87994'], dtype=object),
array(['m1406', 'u56283'], dtype=object),
array(['m8846', 'u25382'], dtype=object),
array(['m789', 'u13891'], dtype=object),
array(['m12317', 'u93011'], dtype=object),
```

```
array(['m2015', 'u80365'], dtype=object),
array(['m2816', 'u78202'], dtype=object),
array(['m11544', 'u99939'], dtype=object),
array(['m8405', 'u57565'], dtype=object),
array(['m11820', 'u26126'], dtype=object),
array(['m15393', 'u46231'], dtype=object),
array(['m2580', 'u38448'], dtype=object),
array(['m15582', 'u71104'], dtype=object),
array(['m5430', 'u5727'], dtype=object),
array(['m11910', 'u91114'], dtype=object),
array(['m4306', 'u64200'], dtype=object),
array(['m758', 'u69990'], dtype=object),
array(['m15116', 'u88652'], dtype=object),
array(['m4865', 'u52153'], dtype=object),
array(['m8832', 'u9796'], dtype=object),
array(['m3638', 'u38085'], dtype=object),
array(['m1632', 'u2133'], dtype=object),
array(['m6824', 'u98044'], dtype=object),
array(['m9614', 'u71594'], dtype=object),
array(['m3622', 'u78404'], dtype=object),
array(['m14624', 'u81981'], dtype=object),
array(['m6464', 'u9561'], dtype=object),
array(['m16668', 'u35638'], dtype=object),
array(['m12470', 'u14154'], dtype=object),
array(['m13526', 'u78772'], dtype=object),
array(['m16640', 'u46429'], dtype=object),
array(['m2186', 'u82514'], dtype=object),
array(['m5327', 'u78235'], dtype=object),
array(['m4118', 'u481'], dtype=object),
array(['m4721', 'u48355'], dtype=object),
array(['m11222', 'u98579'], dtype=object),
array(['m6902', 'u68093'], dtype=object),
array(['m14149', 'u62609'], dtype=object),
array(['m13628', 'u49355'], dtype=object),
array(['m1642', 'u91701'], dtype=object),
array(['m13015', 'u25825'], dtype=object),
array(['m13462', 'u16236'], dtype=object),
array(['m4577', 'u87613'], dtype=object),
array(['m1652', 'u35111'], dtype=object),
array(['m13298', 'u4753'], dtype=object),
array(['m15070', 'u14642'], dtype=object),
array(['m11182', 'u68083'], dtype=object),
array(['m6134', 'u61015'], dtype=object),
array(['m14716', 'u66109'], dtype=object),
array(['m12355', 'u42275'], dtype=object),
array(['m4056', 'u25343'], dtype=object),
array(['m6099', 'u199'], dtype=object),
```

```
array(['m15875', 'u21377'], dtype=object),
array(['m8570', 'u56371'], dtype=object),
array(['m10271', 'u60529'], dtype=object),
array(['m11089', 'u77225'], dtype=object),
array(['m1810', 'u59592'], dtype=object),
array(['m3905', 'u82188'], dtype=object),
array(['m843', 'u61315'], dtype=object),
array(['m4996', 'u76052'], dtype=object),
array(['m7642', 'u9326'], dtype=object),
array(['m15114', 'u73316'], dtype=object),
array(['m14538', 'u51651'], dtype=object),
array(['m13602', 'u57081'], dtype=object),
array(['m14103', 'u74332'], dtype=object),
array(['m5075', 'u16397'], dtype=object),
array(['m10730', 'u54821'], dtype=object),
array(['m14016', 'u5530'], dtype=object),
array(['m1096', 'u8382'], dtype=object),
array(['m15919', 'u89221'], dtype=object),
array(['m2462', 'u85376'], dtype=object),
array(['m15440', 'u54064'], dtype=object),
array(['m15472', 'u16074'], dtype=object),
array(['m10172', 'u6173'], dtype=object),
array(['m2252', 'u20774'], dtype=object),
array(['m758', 'u98250'], dtype=object),
array(['m3905', 'u45349'], dtype=object),
array(['m15450', 'u59052'], dtype=object),
array(['m5112', 'u93356'], dtype=object),
array(['m13534', 'u36978'], dtype=object),
array(['m14691', 'u35794'], dtype=object),
array(['m9111', 'u51300'], dtype=object),
array(['m4727', 'u72395'], dtype=object),
array(['m16912', 'u90297'], dtype=object),
array(['m3526', 'u52029'], dtype=object),
array(['m9051', 'u84970'], dtype=object),
array(['m9620', 'u57633'], dtype=object),
array(['m14010', 'u99542'], dtype=object),
array(['m4341', 'u20369'], dtype=object),
array(['m1798', 'u98289'], dtype=object),
array(['m8644', 'u26479'], dtype=object),
array(['m11907', 'u69656'], dtype=object),
array(['m10596', 'u97578'], dtype=object),
array(['m4127', 'u48198'], dtype=object),
array(['m2896', 'u74320'], dtype=object),
array(['m15646', 'u72559'], dtype=object),
array(['m4906', 'u74122'], dtype=object),
array(['m6860', 'u75681'], dtype=object),
array(['m16969', 'u90565'], dtype=object),
```

```
array(['m11283', 'u44551'], dtype=object),
array(['m8016', 'u70069'], dtype=object),
array(['m16308', 'u23061'], dtype=object),
array(['m8524', 'u33758'], dtype=object),
array(['m11931', 'u80276'], dtype=object),
array(['m13402', 'u19974'], dtype=object),
array(['m3282', 'u80124'], dtype=object),
array(['m3282', 'u19296'], dtype=object),
array(['m7399', 'u29827'], dtype=object),
array(['m9051', 'u22129'], dtype=object),
array(['m1145', 'u66985'], dtype=object),
array(['m10583', 'u61030'], dtype=object),
array(['m17262', 'u3694'], dtype=object),
array(['m2800', 'u46389'], dtype=object),
array(['m6206', 'u27377'], dtype=object),
array(['m6574', 'u9582'], dtype=object),
array(['m381', 'u62171'], dtype=object),
array(['m6231', 'u89152'], dtype=object),
array(['m6574', 'u1894'], dtype=object),
array(['m4753', 'u9861'], dtype=object),
array(['m4825', 'u39297'], dtype=object),
array(['m1700', 'u19268'], dtype=object),
array(['m8230', 'u16272'], dtype=object),
array(['m16713', 'u53694'], dtype=object),
array(['m11907', 'u13528'], dtype=object),
array(['m11314', 'u61468'], dtype=object),
array(['m11337', 'u38087'], dtype=object),
array(['m15385', 'u80478'], dtype=object),
array(['m958', 'u4350'], dtype=object),
array(['m12785', 'u93917'], dtype=object),
array(['m6287', 'u74294'], dtype=object),
array(['m15132', 'u55436'], dtype=object),
array(['m4266', 'u33942'], dtype=object),
array(['m8806', 'u7903'], dtype=object),
array(['m3713', 'u77645'], dtype=object),
array(['m2128', 'u26280'], dtype=object),
array(['m14667', 'u65205'], dtype=object),
array(['m17085', 'u55194'], dtype=object),
array(['m14660', 'u86538'], dtype=object),
array(['m1300', 'u98347'], dtype=object),
array(['m6339', 'u45191'], dtype=object),
array(['m886', 'u42096'], dtype=object),
array(['m7635', 'u54902'], dtype=object),
array(['m11337', 'u76715'], dtype=object),
array(['m11443', 'u9748'], dtype=object),
array(['m3216', 'u32361'], dtype=object),
array(['m10344', 'u92047'], dtype=object),
```

```
array(['m10253', 'u1333'], dtype=object),
array(['m14410', 'u58957'], dtype=object),
array(['m1116', 'u82197'], dtype=object),
array(['m607', 'u74420'], dtype=object),
array(['m10080', 'u85373'], dtype=object),
array(['m5317', 'u8095'], dtype=object),
array(['m15366', 'u4576'], dtype=object),
array(['m5614', 'u29242'], dtype=object),
array(['m11952', 'u66871'], dtype=object),
array(['m12633', 'u53546'], dtype=object),
array(['m14825', 'u61863'], dtype=object),
array(['m7904', 'u18122'], dtype=object),
array(['m16380', 'u36000'], dtype=object),
array(['m2200', 'u33927'], dtype=object),
array(['m10832', 'u47091'], dtype=object),
array(['m15671', 'u46105'], dtype=object),
array(['m11164', 'u25215'], dtype=object),
array(['m16740', 'u32102'], dtype=object),
array(['m7879', 'u27679'], dtype=object),
array(['m1495', 'u76880'], dtype=object),
array(['m5628', 'u48022'], dtype=object),
array(['m17031', 'u88733'], dtype=object),
array(['m15816', 'u13334'], dtype=object),
array(['m9593', 'u61350'], dtype=object),
array(['m12435', 'u27575'], dtype=object),
array(['m15151', 'u39842'], dtype=object),
array(['m7158', 'u11234'], dtype=object),
array(['m8387', 'u95997'], dtype=object),
array(['m13102', 'u40408'], dtype=object),
array(['m7991', 'u57528'], dtype=object),
array(['m14808', 'u37591'], dtype=object),
array(['m4157', 'u93355'], dtype=object),
array(['m13050', 'u30423'], dtype=object),
array(['m4705', 'u10943'], dtype=object),
array(['m2015', 'u93695'], dtype=object),
array(['m607', 'u7846'], dtype=object),
array(['m4369', 'u79451'], dtype=object),
array(['m1476', 'u3101'], dtype=object),
array(['m1962', 'u2469'], dtype=object),
array(['m6050', 'u65809'], dtype=object),
array(['m15394', 'u33972'], dtype=object),
array(['m4310', 'u13227'], dtype=object),
array(['m17355', 'u32548'], dtype=object),
array(['m12243', 'u6504'], dtype=object),
array(['m16707', 'u26570'], dtype=object),
array(['m10503', 'u18427'], dtype=object),
array(['m17562', 'u18103'], dtype=object),
```

```
array(['m7767', 'u64596'], dtype=object),
array(['m8379', 'u62212'], dtype=object),
array(['m3860', 'u16386'], dtype=object),
array(['m12161', 'u98642'], dtype=object),
array(['m3825', 'u41480'], dtype=object),
array(['m5317', 'u75410'], dtype=object),
array(['m11279', 'u18360'], dtype=object),
array(['m9471', 'u20365'], dtype=object),
array(['m5862', 'u11502'], dtype=object),
array(['m1180', 'u40240'], dtype=object),
array(['m7397', 'u46951'], dtype=object),
array(['m16201', 'u51405'], dtype=object),
array(['m10664', 'u13620'], dtype=object),
array(['m14928', 'u27061'], dtype=object),
array(['m5926', 'u54942'], dtype=object),
array(['m17221', 'u87003'], dtype=object),
array(['m6166', 'u95862'], dtype=object),
array(['m2868', 'u64917'], dtype=object),
array(['m4214', 'u11731'], dtype=object),
array(['m13948', 'u74533'], dtype=object),
array(['m9433', 'u92339'], dtype=object),
array(['m16139', 'u37730'], dtype=object),
array(['m7057', 'u81608'], dtype=object),
array(['m9717', 'u81287'], dtype=object),
array(['m8644', 'u77425'], dtype=object),
array(['m8585', 'u96402'], dtype=object),
array(['m1255', 'u23051'], dtype=object),
array(['m6221', 'u61765'], dtype=object),
array(['m5886', 'u18631'], dtype=object),
array(['m4043', 'u14081'], dtype=object),
array(['m5924', 'u31223'], dtype=object),
array(['m16265', 'u50375'], dtype=object),
array(['m5509', 'u71527'], dtype=object),
array(['m11337', 'u47981'], dtype=object),
array(['m7234', 'u62171'], dtype=object),
array(['m10162', 'u61457'], dtype=object),
array(['m2612', 'u333'], dtype=object),
array(['m11886', 'u42416'], dtype=object),
array(['m4545', 'u53332'], dtype=object),
array(['m12785', 'u71661'], dtype=object),
array(['m3703', 'u64996'], dtype=object),
array(['m9508', 'u12765'], dtype=object),
array(['m4881', 'u73822'], dtype=object),
array(['m3638', 'u75327'], dtype=object),
array(['m8333', 'u31301'], dtype=object),
array(['m2348', 'u72395'], dtype=object),
array(['m9617', 'u84152'], dtype=object),
```

```
array(['m13384', 'u14422'], dtype=object),
array(['m8682', 'u80354'], dtype=object),
array(['m14160', 'u72668'], dtype=object),
array(['m3079', 'u663'], dtype=object),
array(['m16702', 'u65594'], dtype=object),
array(['m1144', 'u43604'], dtype=object),
array(['m7635', 'u47599'], dtype=object),
array(['m12084', 'u34856'], dtype=object),
array(['m12582', 'u46086'], dtype=object),
array(['m7032', 'u16996'], dtype=object),
array(['m9049', 'u7544'], dtype=object),
array(['m8687', 'u56108'], dtype=object),
array(['m13887', 'u52664'], dtype=object),
array(['m10094', 'u43177'], dtype=object),
array(['m7701', 'u49355'], dtype=object),
array(['m5732', 'u76267'], dtype=object),
array(['m1743', 'u22709'], dtype=object),
array(['m6386', 'u49324'], dtype=object),
array(['m5477', 'u73437'], dtype=object),
array(['m6350', 'u11142'], dtype=object),
array(['m7145', 'u97613'], dtype=object),
array(['m8387', 'u56859'], dtype=object),
array(['m10461', 'u21322'], dtype=object),
array(['m6574', 'u36581'], dtype=object),
array(['m5472', 'u53629'], dtype=object),
array(['m5582', 'u80055'], dtype=object),
array(['m9800', 'u68131'], dtype=object),
array(['m11271', 'u18415'], dtype=object),
array(['m7031', 'u82233'], dtype=object),
array(['m4564', 'u25049'], dtype=object),
array(['m1409', 'u81760'], dtype=object),
array(['m15124', 'u83247'], dtype=object),
array(['m10378', 'u74022'], dtype=object),
array(['m6219', 'u27434'], dtype=object),
array(['m2922', 'u64120'], dtype=object),
array(['m7713', 'u57608'], dtype=object),
array(['m13705', 'u2693'], dtype=object),
array(['m8181', 'u58995'], dtype=object),
array(['m7032', 'u78516'], dtype=object),
array(['m457', 'u69690'], dtype=object),
array(['m16587', 'u19006'], dtype=object),
array(['m15853', 'u46536'], dtype=object),
array(['m17324', 'u94821'], dtype=object),
array(['m1709', 'u11137'], dtype=object),
array(['m621', 'u78770'], dtype=object),
array(['m10418', 'u49890'], dtype=object),
array(['m6386', 'u41701'], dtype=object),
```

```
array(['m12672', 'u66451'], dtype=object),
array(['m10094', 'u78859'], dtype=object),
array(['m9960', 'u81097'], dtype=object),
array(['m12221', 'u7683'], dtype=object),
array(['m8151', 'u66885'], dtype=object),
array(['m2643', 'u42501'], dtype=object),
array(['m528', 'u29027'], dtype=object),
array(['m9591', 'u32668'], dtype=object),
array(['m8', 'u85419'], dtype=object),
array(['m11995', 'u56069'], dtype=object),
array(['m7635', 'u79194'], dtype=object),
array(['m8753', 'u65249'], dtype=object),
array(['m15342', 'u40136'], dtype=object),
array(['m5435', 'u43729'], dtype=object),
array(['m10027', 'u78610'], dtype=object),
array(['m763', 'u59539'], dtype=object),
array(['m13486', 'u50057'], dtype=object),
array(['m12338', 'u29183'], dtype=object),
array(['m299', 'u42416'], dtype=object),
array(['m6339', 'u29208'], dtype=object),
array(['m6464', 'u24393'], dtype=object),
array(['m7511', 'u41422'], dtype=object),
array(['m17189', 'u39450'], dtype=object),
array(['m9074', 'u67900'], dtype=object),
array(['m9471', 'u78970'], dtype=object),
array(['m12090', 'u60489'], dtype=object),
array(['m16265', 'u98012'], dtype=object),
array(['m17062', 'u95372'], dtype=object),
array(['m273', 'u88489'], dtype=object),
array(['m5103', 'u83006'], dtype=object),
array(['m2675', 'u94109'], dtype=object),
array(['m5239', 'u24696'], dtype=object),
array(['m12186', 'u76819'], dtype=object),
array(['m10170', 'u14682'], dtype=object),
array(['m16954', 'u93489'], dtype=object),
array(['m16232', 'u13843'], dtype=object),
array(['m6251', 'u9911'], dtype=object),
array(['m3106', 'u31938'], dtype=object),
array(['m13050', 'u86332'], dtype=object),
array(['m3756', 'u3458'], dtype=object),
array(['m14047', 'u70126'], dtype=object),
array(['m2680', 'u2693'], dtype=object),
array(['m7169', 'u17890'], dtype=object),
array(['m5154', 'u21870'], dtype=object),
array(['m13519', 'u62691'], dtype=object),
array(['m9800', 'u80354'], dtype=object),
array(['m1770', 'u90253'], dtype=object),
```

```
array(['m16711', 'u96124'], dtype=object),
array(['m78', 'u76196'], dtype=object),
array(['m16882', 'u69867'], dtype=object),
array(['m9628', 'u49727'], dtype=object),
array(['m2881', 'u55053'], dtype=object),
array(['m8016', 'u55282'], dtype=object),
array(['m14538', 'u25442'], dtype=object),
array(['m2391', 'u71400'], dtype=object),
array(['m12911', 'u83186'], dtype=object),
array(['m6303', 'u52168'], dtype=object),
array(['m13244', 'u78404'], dtype=object),
array(['m3106', 'u98321'], dtype=object),
array(['m7699', 'u51817'], dtype=object),
array(['m7346', 'u77383'], dtype=object),
array(['m6720', 'u56514'], dtype=object),
array(['m16565', 'u94474'], dtype=object),
array(['m6235', 'u13445'], dtype=object),
array(['m12891', 'u54632'], dtype=object),
array(['m6797', 'u86379'], dtype=object),
array(['m361', 'u63653'], dtype=object),
array(['m4577', 'u31810'], dtype=object),
array(['m16711', 'u40175'], dtype=object),
array(['m1798', 'u31336'], dtype=object),
array(['m16765', 'u25572'], dtype=object),
array(['m4656', 'u94170'], dtype=object),
array(['m312', 'u85582'], dtype=object),
array(['m357', 'u4983'], dtype=object),
array(['m1861', 'u31230'], dtype=object),
array(['m14827', 'u24705'], dtype=object),
array(['m4123', 'u67725'], dtype=object),
array(['m10418', 'u40043'], dtype=object),
array(['m17405', 'u8074'], dtype=object),
array(['m7718', 'u70172'], dtype=object),
array(['m11077', 'u32816'], dtype=object),
array(['m12161', 'u63193'], dtype=object),
array(['m14621', 'u64531'], dtype=object),
array(['m14691', 'u39830'], dtype=object),
array(['m13359', 'u82817'], dtype=object),
array(['m14482', 'u13061'], dtype=object),
array(['m6874', 'u83006'], dtype=object),
array(['m8301', 'u74095'], dtype=object),
array(['m9378', 'u39068'], dtype=object),
array(['m9205', 'u87201'], dtype=object),
array(['m10947', 'u67291'], dtype=object),
array(['m499', 'u47607'], dtype=object),
array(['m15563', 'u22560'], dtype=object),
array(['m17503', 'u78673'], dtype=object),
```

```
array(['m720', 'u22193'], dtype=object),
array(['m9049', 'u35256'], dtype=object),
array(['m1799', 'u14868'], dtype=object),
array(['m1202', 'u47607'], dtype=object),
array(['m9395', 'u31081'], dtype=object),
array(['m3071', 'u6460'], dtype=object),
array(['m16380', 'u50693'], dtype=object),
array(['m2333', 'u84248'], dtype=object),
array(['m2763', 'u85125'], dtype=object),
array(['m15300', 'u61274'], dtype=object),
array(['m13705', 'u81608'], dtype=object),
array(['m16438', 'u52781'], dtype=object),
array(['m7745', 'u35003'], dtype=object),
array(['m7560', 'u38764'], dtype=object),
array(['m16468', 'u64996'], dtype=object),
array(['m6205', 'u76901'], dtype=object),
array(['m15953', 'u60802'], dtype=object),
array(['m13580', 'u57736'], dtype=object),
array(['m14149', 'u42487'], dtype=object),
array(['m14149', 'u11965'], dtype=object),
array(['m15084', 'u16273'], dtype=object),
array(['m14312', 'u1507'], dtype=object),
array(['m5582', 'u32548'], dtype=object),
array(['m13195', 'u54605'], dtype=object),
array(['m331', 'u61322'], dtype=object),
array(['m7852', 'u93804'], dtype=object),
array(['m2348', 'u8158'], dtype=object),
array(['m14109', 'u77457'], dtype=object),
array(['m5496', 'u38600'], dtype=object),
array(['m17169', 'u86286'], dtype=object),
array(['m5103', 'u78573'], dtype=object),
array(['m9886', 'u48076'], dtype=object),
array(['m11855', 'u74441'], dtype=object),
array(['m7032', 'u15539'], dtype=object),
array(['m5793', 'u24454'], dtype=object),
array(['m6117', 'u52352'], dtype=object),
array(['m2874', 'u99844'], dtype=object),
array(['m14218', 'u40121'], dtype=object),
array(['m2423', 'u53629'], dtype=object),
array(['m12065', 'u16287'], dtype=object),
array(['m6976', 'u74533'], dtype=object),
array(['m6475', 'u32408'], dtype=object),
array(['m14660', 'u19630'], dtype=object),
array(['m5169', 'u64148'], dtype=object),
array(['m9051', 'u19596'], dtype=object),
array(['m3611', 'u50465'], dtype=object),
array(['m8428', 'u31059'], dtype=object),
```

```
array(['m14240', 'u12813'], dtype=object),
array(['m2200', 'u62638'], dtype=object),
array(['m12476', 'u32307'], dtype=object),
array(['m11833', 'u22001'], dtype=object),
array(['m6030', 'u5507'], dtype=object),
array(['m16869', 'u80871'], dtype=object),
array(['m6235', 'u85760'], dtype=object),
array(['m14313', 'u49691'], dtype=object),
array(['m3333', 'u46666'], dtype=object),
array(['m2379', 'u97578'], dtype=object),
array(['m17724', 'u72566'], dtype=object),
array(['m3138', 'u9360'], dtype=object),
array(['m16303', 'u7620'], dtype=object),
array(['m12494', 'u62279'], dtype=object),
array(['m10832', 'u47236'], dtype=object),
array(['m14852', 'u18922'], dtype=object),
array(['m5836', 'u16558'], dtype=object),
array(['m13382', 'u90745'], dtype=object),
array(['m2360', 'u82662'], dtype=object),
array(['m4745', 'u86506'], dtype=object),
array(['m4145', 'u15402'], dtype=object),
array(['m6670', 'u62238'], dtype=object),
array(['m7544', 'u94821'], dtype=object),
array(['m16604', 'u18415'], dtype=object),
array(['m16969', 'u59186'], dtype=object),
array(['m1754', 'u39391'], dtype=object),
array(['m3970', 'u2307'], dtype=object),
array(['m14538', 'u31655'], dtype=object),
array(['m5732', 'u13461'], dtype=object),
array(['m5162', 'u14690'], dtype=object),
array(['m3925', 'u47771'], dtype=object),
array(['m14995', 'u49585'], dtype=object),
array(['m2400', 'u9280'], dtype=object),
array(['m8099', 'u91371'], dtype=object),
array(['m6500', 'u2225'], dtype=object),
array(['m16703', 'u19938'], dtype=object),
array(['m1794', 'u60199'], dtype=object),
array(['m8316', 'u55723'], dtype=object),
array(['m12161', 'u20826'], dtype=object),
array(['m6240', 'u84930'], dtype=object),
array(['m1163', 'u21705'], dtype=object),
array(['m7230', 'u74441'], dtype=object),
array(['m8818', 'u65710'], dtype=object),
array(['m2680', 'u41580'], dtype=object),
array(['m14827', 'u44516'], dtype=object),
array(['m10832', 'u61528'], dtype=object),
array(['m14088', 'u52523'], dtype=object),
```

```
array(['m6971', 'u55096'], dtype=object),
array(['m17355', 'u7577'], dtype=object),
array(['m14284', 'u89256'], dtype=object),
array(['m14454', 'u86474'], dtype=object),
array(['m3905', 'u50095'], dtype=object),
array(['m9051', 'u53190'], dtype=object),
array(['m14670', 'u40950'], dtype=object),
array(['m15803', 'u73865'], dtype=object),
array(['m12077', 'u71390'], dtype=object),
array(['m10359', 'u88915'], dtype=object),
array(['m1324', 'u96823'], dtype=object),
array(['m4671', 'u27888'], dtype=object),
array(['m329', 'u91613'], dtype=object),
array(['m4240', 'u85842'], dtype=object),
array(['m17508', 'u34885'], dtype=object),
array(['m14905', 'u75828'], dtype=object),
array(['m1770', 'u32946'], dtype=object),
array(['m11149', 'u81227'], dtype=object),
array(['m15627', 'u32885'], dtype=object),
array(['m16952', 'u14756'], dtype=object),
array(['m4123', 'u8004'], dtype=object),
array(['m11526', 'u33473'], dtype=object),
array(['m14313', 'u75271'], dtype=object),
array(['m9644', 'u8207'], dtype=object),
array(['m2470', 'u5915'], dtype=object),
array(['m13359', 'u35131'], dtype=object),
array(['m7991', 'u96111'], dtype=object),
array(['m7599', 'u16273'], dtype=object),
array(['m16204', 'u21667'], dtype=object),
array(['m7331', 'u60106'], dtype=object),
array(['m3290', 'u485'], dtype=object),
array(['m468', 'u69175'], dtype=object),
array(['m12911', 'u83151'], dtype=object),
array(['m9115', 'u24542'], dtype=object),
array(['m6329', 'u34473'], dtype=object),
array(['m17672', 'u32563'], dtype=object),
array(['m10344', 'u97799'], dtype=object),
array(['m9481', 'u78487'], dtype=object),
array(['m11406', 'u10638'], dtype=object),
array(['m16242', 'u91990'], dtype=object),
array(['m199', 'u87036'], dtype=object),
array(['m3912', 'u17864'], dtype=object),
array(['m10666', 'u79380'], dtype=object),
array(['m16995', 'u20766'], dtype=object),
array(['m13081', 'u96743'], dtype=object),
array(['m15853', 'u79242'], dtype=object),
array(['m4577', 'u71005'], dtype=object),
```

```
array(['m10928', 'u37513'], dtype=object),
array(['m4216', 'u98164'], dtype=object),
array(['m5112', 'u63666'], dtype=object),
array(['m1530', 'u12812'], dtype=object),
array(['m5515', 'u18187'], dtype=object),
array(['m15322', 'u49450'], dtype=object),
array(['m1102', 'u83978'], dtype=object),
array(['m6859', 'u83586'], dtype=object),
array(['m5226', 'u54974'], dtype=object),
array(['m10921', 'u25952'], dtype=object),
array(['m3626', 'u86837'], dtype=object),
array(['m16796', 'u29933'], dtype=object),
array(['m9828', 'u74831'], dtype=object),
array(['m13115', 'u67134'], dtype=object),
array(['m14358', 'u72923'], dtype=object),
array(['m14691', 'u90659'], dtype=object),
array(['m2366', 'u49142'], dtype=object),
array(['m788', 'u57565'], dtype=object),
array(['m9882', 'u80155'], dtype=object),
array(['m4745', 'u43608'], dtype=object),
array(['m7586', 'u78922'], dtype=object),
array(['m12232', 'u21983'], dtype=object),
array(['m14868', 'u64765'], dtype=object),
array(['m16181', 'u57347'], dtype=object),
array(['m13628', 'u26124'], dtype=object),
array(['m5515', 'u62546'], dtype=object),
array(['m11789', 'u56514'], dtype=object),
array(['m10158', 'u35353'], dtype=object),
array(['m11678', 'u84102'], dtype=object),
array(['m4656', 'u48915'], dtype=object),
array(['m3684', 'u27660'], dtype=object),
array(['m7511', 'u46988'], dtype=object),
array(['m11820', 'u37883'], dtype=object),
array(['m16552', 'u46711'], dtype=object),
array(['m17088', 'u54255'], dtype=object),
array(['m8034', 'u3593'], dtype=object),
array(['m16139', 'u60217'], dtype=object),
array(['m6692', 'u7063'], dtype=object),
array(['m4683', 'u22727'], dtype=object),
array(['m11089', 'u53074'], dtype=object),
array(['m16459', 'u56283'], dtype=object),
array(['m6287', 'u24759'], dtype=object),
array(['m13955', 'u63025'], dtype=object),
array(['m12140', 'u42479'], dtype=object),
array(['m6262', 'u6056'], dtype=object),
array(['m10890', 'u6663'], dtype=object),
array(['m6350', 'u94565'], dtype=object),
```

```
array(['m6060', 'u49089'], dtype=object),
array(['m12718', 'u32885'], dtype=object),
array(['m11677', 'u36689'], dtype=object),
array(['m3756', 'u38203'], dtype=object),
array(['m143', 'u63178'], dtype=object),
array(['m10774', 'u58915'], dtype=object),
array(['m10820', 'u42771'], dtype=object),
array(['m10846', 'u69860'], dtype=object),
array(['m15373', 'u11356'], dtype=object),
array(['m9224', 'u60212'], dtype=object),
array(['m3756', 'u16843'], dtype=object),
array(['m15827', 'u71005'], dtype=object),
array(['m12470', 'u37051'], dtype=object),
array(['m14691', 'u50385'], dtype=object),
array(['m17008', 'u54221'], dtype=object),
array(['m10173', 'u92774'], dtype=object),
array(['m11265', 'u88818'], dtype=object),
array(['m2862', 'u67749'], dtype=object),
array(['m7364', 'u78922'], dtype=object),
array(['m11064', 'u57347'], dtype=object),
array(['m11283', 'u47998'], dtype=object),
array(['m16169', 'u76344'], dtype=object),
array(['m8325', 'u50253'], dtype=object),
array(['m15557', 'u43216'], dtype=object),
array(['m290', 'u97199'], dtype=object),
array(['m6708', 'u67104'], dtype=object),
array(['m9593', 'u30221'], dtype=object),
array(['m11752', 'u5977'], dtype=object),
array(['m16022', 'u49890'], dtype=object),
array(['m11314', 'u15402'], dtype=object),
array(['m3907', 'u18659'], dtype=object),
array(['m15421', 'u4057'], dtype=object),
array(['m6971', 'u57366'], dtype=object),
array(['m13651', 'u28873'], dtype=object),
array(['m1256', 'u12554'], dtype=object),
array(['m175', 'u82700'], dtype=object),
array(['m12293', 'u10901'], dtype=object),
array(['m16016', 'u53694'], dtype=object),
array(['m1060', 'u28338'], dtype=object),
array(['m416', 'u97530'], dtype=object),
array(['m6670', 'u23589'], dtype=object),
array(['m6287', 'u45538'], dtype=object),
array(['m330', 'u78143'], dtype=object),
array(['m5847', 'u43865'], dtype=object),
array(['m1798', 'u33694'], dtype=object),
array(['m17707', 'u88005'], dtype=object),
array(['m4049', 'u46858'], dtype=object),
```

```
array(['m15345', 'u28656'], dtype=object),
array(['m5360', 'u49142'], dtype=object),
array(['m6029', 'u55922'], dtype=object),
array(['m2660', 'u83756'], dtype=object),
array(['m17338', 'u86987'], dtype=object),
array(['m4972', 'u89427'], dtype=object),
array(['m4591', 'u83926'], dtype=object),
array(['m7399', 'u41400'], dtype=object),
array(['m12015', 'u99805'], dtype=object),
array(['m4356', 'u92710'], dtype=object),
array(['m10926', 'u21199'], dtype=object),
array(['m10436', 'u73216'], dtype=object),
array(['m6642', 'u71605'], dtype=object),
array(['m705', 'u88528'], dtype=object),
array(['m501', 'u44300'], dtype=object),
array(['m12014', 'u77210'], dtype=object),
array(['m4315', 'u20187'], dtype=object),
array(['m8526', 'u41058'], dtype=object),
array(['m6221', 'u97199'], dtype=object),
array(['m7363', 'u58889'], dtype=object),
array(['m14909', 'u85743'], dtype=object),
array(['m8782', 'u91502'], dtype=object),
array(['m3938', 'u53419'], dtype=object),
array(['m16158', 'u15104'], dtype=object),
array(['m8045', 'u27971'], dtype=object),
array(['m10214', 'u66885'], dtype=object),
array(['m7509', 'u49828'], dtype=object),
array(['m8387', 'u20658'], dtype=object),
array(['m2615', 'u57633'], dtype=object),
array(['m17036', 'u14682'], dtype=object),
array(['m15784', 'u46928'], dtype=object),
array(['m5926', 'u46228'], dtype=object),
array(['m11398', 'u7963'], dtype=object),
array(['m15844', 'u664'], dtype=object),
array(['m7249', 'u24612'], dtype=object),
array(['m13580', 'u77582'], dtype=object),
array(['m2735', 'u62499'], dtype=object),
array(['m6134', 'u17369'], dtype=object),
array(['m7589', 'u18753'], dtype=object),
array(['m10774', 'u22632'], dtype=object),
array(['m13192', 'u7620'], dtype=object),
array(['m12172', 'u73629'], dtype=object),
array(['m17302', 'u78398'], dtype=object),
array(['m11077', 'u6968'], dtype=object),
array(['m894', 'u58995'], dtype=object),
array(['m5172', 'u3402'], dtype=object),
array(['m14302', 'u31158'], dtype=object),
```

```
array(['m14527', 'u79080'], dtype=object),
array(['m11315', 'u86712'], dtype=object),
array(['m14466', 'u60250'], dtype=object),
array(['m11446', 'u59569'], dtype=object),
array(['m16459', 'u35824'], dtype=object),
array(['m7032', 'u9172'], dtype=object),
array(['m7917', 'u89853'], dtype=object),
array(['m3900', 'u28489'], dtype=object),
array(['m5169', 'u45234'], dtype=object),
array(['m758', 'u30021'], dtype=object),
array(['m14364', 'u63265'], dtype=object),
array(['m1682', 'u79224'], dtype=object),
array(['m8635', 'u71202'], dtype=object),
array(['m17221', 'u42064'], dtype=object),
array(['m11337', 'u21823'], dtype=object),
array(['m8535', 'u77333'], dtype=object),
array(['m12582', 'u24298'], dtype=object),
array(['m10418', 'u42147'], dtype=object),
array(['m4736', 'u66187'], dtype=object),
array(['m14476', 'u1658'], dtype=object),
array(['m4306', 'u72284'], dtype=object),
array(['m6134', 'u27526'], dtype=object),
array(['m9048', 'u91234'], dtype=object),
array(['m2372', 'u42099'], dtype=object),
array(['m12074', 'u88159'], dtype=object),
array(['m11234', 'u30081'], dtype=object),
array(['m14621', 'u23293'], dtype=object),
array(['m5477', 'u97090'], dtype=object),
array(['m2319', 'u16786'], dtype=object),
array(['m2340', 'u8334'], dtype=object),
array(['m16885', 'u17665'], dtype=object),
array(['m10947', 'u79484'], dtype=object),
array(['m15064', 'u22853'], dtype=object),
array(['m15151', 'u74320'], dtype=object),
array(['m1256', 'u27870'], dtype=object),
array(['m15674', 'u75976'], dtype=object),
array(['m14936', 'u7634'], dtype=object),
array(['m5092', 'u48607'], dtype=object),
array(['m3638', 'u69888'], dtype=object),
array(['m13081', 'u97611'], dtype=object),
array(['m1542', 'u62224'], dtype=object),
array(['m13532', 'u49142'], dtype=object),
array(['m11164', 'u56790'], dtype=object),
array(['m16768', 'u92394'], dtype=object),
array(['m8904', 'u62583'], dtype=object),
array(['m2112', 'u61189'], dtype=object),
array(['m2452', 'u32779'], dtype=object),
```

```
 array(['m8454', 'u42416'], dtype=object),
 array(['m3267', 'u65257'], dtype=object),
 …]
```

[117]:
```
from collections import defaultdict
import json
def create_user_movie_dict_df2ex(df):

    # if not os.path.isfile('user_movie_train.json'):
    user_movie = defaultdict(dict)
    for iter, row in df.iterrows():

        dict1={}
        dict1=user_movie[row[1]].copy()
        dict1[row[0]]=row[2]
        user_movie[row[1]]=dict1
    #     my_json = json.dumps(user_movie)
    #     f = open("user_movie_train.json","w")
    #     f.write(my_json)
    #     f.close()
    # else:
    #     print("opening json file")
    #     with open('user_movie_train.json') as json_file:
    #       user_movie = json.load(json_file)
    # print('user_movie_train.json loaded')
    return user_movie
```

[118]:
```
df2_explicit_dict=create_user_movie_dict_df2ex(df2_explicit)
```

[119]:
```
ratings_train=[]
for i in range(len(array_true_ex_train)):
  movie=array_true_ex_train[i][0]
  user=array_true_ex_train[i][1]
  ratings_train.append(df2_explicit_dict[user][movie])
len(ratings_train)
```

[119]: 478281

[120]:
```
labels_model_selection[:15]
```

[120]: array([1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1])

[121]:
```
len(examples_model_selection)
```

[121]: 239177

[122]:
```
len(labels_model_selection)
```

[122]: 239177

[123]:
```
array_true_ex_select=[]
for i in range(len(labels_model_selection)):
  if labels_model_selection[i]==1:
    array_true_ex_select.append(examples_model_selection[i])
```

[124]:
```
len(array_true_ex_select)
```

[124]: 119661

[125]:
```
ratings_selection=[]
for i in range(len(array_true_ex_select)):
  movie=array_true_ex_select[i][0]
  user=array_true_ex_select[i][1]
  ratings_selection.append(df2_explicit_dict[user][movie])
len(ratings_selection)
```

[125]: 119661

[ ]:
```
ratings_selection
```

16 Train Node2Vec Explicit

17 L1

[127]:
```
clf_l1_ex = train_model_l1_ex(
        array_true_ex_train, ratings_train, embedding_train, operator_l1
    )
```

```
Epoch 1/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.4479
Epoch 2/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.1416
Epoch 3/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0845
Epoch 4/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0781
Epoch 5/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0748
Epoch 6/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0729
Epoch 7/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0718
Epoch 8/40
```

```
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0713
Epoch 9/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0704
Epoch 10/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0695
Epoch 11/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0694
Epoch 12/40
9566/9566 [==============================] - 20s 2ms/step - loss: 1.0690
Epoch 13/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0685
Epoch 14/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0680
Epoch 15/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0678
Epoch 16/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0671
Epoch 17/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0670
Epoch 18/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0664
Epoch 19/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0662
Epoch 20/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0659
Epoch 21/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0658
Epoch 22/40
9566/9566 [==============================] - 20s 2ms/step - loss: 1.0657
Epoch 23/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0656
Epoch 24/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0654
Epoch 25/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0654
Epoch 26/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0654
Epoch 27/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0654
Epoch 28/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0654
Epoch 29/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0652
Epoch 30/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0650
Epoch 31/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0649
Epoch 32/40
```

```
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0650
Epoch 33/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0649
Epoch 34/40
9566/9566 [==============================] - 18s 2ms/step - loss: 1.0649
Epoch 35/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0650
Epoch 36/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0647
Epoch 37/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0646
Epoch 38/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0647
Epoch 39/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0647
Epoch 40/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0643
```

[128]:
```
link_features_select_ex = link_examples_to_features(
        array_true_ex_select, embedding_train, operator_l1)
predicted_concat_selection_ex = clf_l1_ex.predict(np.
 ↪array(link_features_select_ex))
predicted_concat_selection_ex=predicted_concat_selection_ex.flatten()
loss_concat_selection_ex=np.
 ↪sum((ratings_selection-predicted_concat_selection_ex)**2)/
 ↪len(predicted_concat_selection_ex)
loss_concat_selection_ex
RMSE=np.sqrt(np.sum((np.
 ↪array(ratings_selection)-predicted_concat_selection_ex)**2)/
 ↪len(predicted_concat_selection_ex))
RMSE
MAPE=np.sum(np.abs((np.array(ratings_selection)-predicted_concat_selection_ex))/
 ↪ratings_selection)/len(predicted_concat_selection_ex)
MAPE
```

```
3740/3740 [==============================] - 4s 1ms/step
```

[128]:
```
0.32976345612399244
```

18 Test L1 Ex

[129]:
```
array_true_ex_test=[]
for i in range(len(labels_test)):
  if labels_test[i]==1:
    array_true_ex_test.append(examples_test[i])
```

```
ratings_test=[]
for i in range(len(array_true_ex_test)):
  movie=array_true_ex_test[i][0]
  user=array_true_ex_test[i][1]
  ratings_test.append(df2_explicit_dict[user][movie])
len(ratings_test)

link_features_test_ex = link_examples_to_features(
        array_true_ex_test, embedding_test, operator_l1)
predicted_concat_test_ex = clf_l1_ex.predict(np.array(link_features_test_ex))
predicted_concat_test_ex=predicted_concat_test_ex.flatten()

loss_concat_test_ex=np.sum((ratings_test-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex)
loss_concat_test_ex

RMSE=np.sqrt(np.sum((np.array(ratings_test)-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex))
RMSE

MAPE=np.sum(np.abs((np.array(ratings_test)-predicted_concat_test_ex))/
 ↪ratings_test)/len(predicted_concat_test_ex)
MAPE
```

```
23358/23358 [==============================] - 27s 1ms/step
```

[129]:
```
0.3403822886340566
```

[130]:
```
loss_concat_test_ex
```

[130]:
```
1.148279111827102
```

[131]:
```
RMSE
```

[131]:
```
1.0715778608328477
```

[132]:
```
MAPE
```

[132]:
```
0.3403822886340566
```

19 Concat

[133]:
```
clf_concat_ex = train_model_concat_ex(
        array_true_ex_train, ratings_train, embedding_train, operator_concat
    )
```

```
Epoch 1/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.1961
Epoch 2/40
9566/9566 [==============================] - 19s 2ms/step - loss: 1.0232
Epoch 3/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9791
Epoch 4/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9695
Epoch 5/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9635
Epoch 6/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9589
Epoch 7/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9552
Epoch 8/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9517
Epoch 9/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9491
Epoch 10/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9466
Epoch 11/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9445
Epoch 12/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9424
Epoch 13/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9403
Epoch 14/40
9566/9566 [==============================] - 18s 2ms/step - loss: 0.9391
Epoch 15/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9378
Epoch 16/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9365
Epoch 17/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9351
Epoch 18/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9335
Epoch 19/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9326
Epoch 20/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9312
Epoch 21/40
```

```
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9304
Epoch 22/40
9566/9566 [==============================] - 20s 2ms/step - loss: 0.9294
Epoch 23/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9284
Epoch 24/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9276
Epoch 25/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9267
Epoch 26/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9262
Epoch 27/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9254
Epoch 28/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9245
Epoch 29/40
9566/9566 [==============================] - 18s 2ms/step - loss: 0.9238
Epoch 30/40
9566/9566 [==============================] - 18s 2ms/step - loss: 0.9232
Epoch 31/40
9566/9566 [==============================] - 18s 2ms/step - loss: 0.9227
Epoch 32/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9218
Epoch 33/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9213
Epoch 34/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9208
Epoch 35/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9203
Epoch 36/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9199
Epoch 37/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9191
Epoch 38/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9189
Epoch 39/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9184
Epoch 40/40
9566/9566 [==============================] - 19s 2ms/step - loss: 0.9177
```

[134]:
```
clf_concat_ex.save('clf_concat_ex.model')
# from tensorflow import keras
# clf_concat_ex = keras.models.load_model('clf_concat_ex.model')
```

[135]:
```
link_features_select_ex = link_examples_to_features(
        array_true_ex_select, embedding_train, operator_concat)
```

```
predicted_concat_selection_ex = clf_concat_ex.predict(np.
 ↪array(link_features_select_ex))
predicted_concat_selection_ex=predicted_concat_selection_ex.flatten()
loss_concat_selection_ex=np.
 ↪sum((ratings_selection-predicted_concat_selection_ex)**2)/
 ↪len(predicted_concat_selection_ex)
loss_concat_selection_ex
RMSE=np.sqrt(np.sum((np.
 ↪array(ratings_selection)-predicted_concat_selection_ex)**2)/
 ↪len(predicted_concat_selection_ex))
RMSE
MAPE=np.sum(np.abs((np.array(ratings_selection)-predicted_concat_selection_ex))/
 ↪ratings_selection)/len(predicted_concat_selection_ex)
MAPE
```

```
3740/3740 [==============================] - 4s 1ms/step
```

[135]:
```
0.2974818025646902
```

[136]:
```
RMSE
```

[136]:
```
0.9663831112167076
```

[137]:
```
predicted_concat_selection_ex = clf_concat_ex.predict(np.
 ↪array(link_features_select_ex))
predicted_concat_selection_ex=predicted_concat_selection_ex.flatten()
```

```
3740/3740 [==============================] - 4s 1ms/step
```

[138]:
```
predicted_concat_selection_ex
```

[138]:
```
array([3.8493724, 3.4844003, 3.611464 , …, 3.867021 , 3.897336 ,
       4.0989695], dtype=float32)
```

[139]:
```
loss_concat_selection_ex=np.
 ↪sum((ratings_selection-predicted_concat_selection_ex)**2)/
 ↪len(predicted_concat_selection_ex)
loss_concat_selection_ex
```

[139]:
```
0.9338963176448836
```

[140]:
```
ratings_selection[-4:]
```

[140]:
```
[5, 5, 5, 4]
```

[141]:
```

```

```
RMSE=np.sqrt(np.sum((np.
 ↪array(ratings_selection)-predicted_concat_selection_ex)**2)/
 ↪len(predicted_concat_selection_ex))
RMSE
```

[141]: 0.9663831112167076

[142]:
```
MAPE=np.sum(np.abs((np.array(ratings_selection)-predicted_concat_selection_ex))/
 ↪ratings_selection)/len(predicted_concat_selection_ex)
MAPE
```

[142]: 0.2974818025646902

20 Test Explicit Node2Vec

[143]:
```
len(examples_test)
```

[143]: 1494854

[144]:
```
len(labels_test)
```

[144]: 1494854

[145]:
```
array_true_ex_test=[]
for i in range(len(labels_test)):
  if labels_test[i]==1:
    array_true_ex_test.append(examples_test[i])

ratings_test=[]
for i in range(len(array_true_ex_test)):
  movie=array_true_ex_test[i][0]
  user=array_true_ex_test[i][1]
  ratings_test.append(df2_explicit_dict[user][movie])
len(ratings_test)

link_features_test_ex = link_examples_to_features(
        array_true_ex_test, embedding_test, operator_concat)
predicted_concat_test_ex = clf_concat_ex.predict(np.
 ↪array(link_features_test_ex))
predicted_concat_test_ex=predicted_concat_test_ex.flatten()

loss_concat_test_ex=np.sum((ratings_test-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex)
loss_concat_test_ex
```

```
RMSE=np.sqrt(np.sum((np.array(ratings_test)-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex))
RMSE

MAPE=np.sum(np.abs((np.array(ratings_test)-predicted_concat_test_ex))/
 ↪ratings_test)/len(predicted_concat_test_ex)
MAPE
```

```
23358/23358 [==============================] - 28s 1ms/step
```

[145]:
```
0.3468014838950562
```

[146]:
```
loss_concat_test_ex
```

[146]:
```
1.4266383999617176
```

[147]:
```
RMSE
```

[147]:
```
1.1944196917171608
```

[148]:
```
MAPE
```

[148]:
```
0.3468014838950562
```

[149]:
```
len(array_true_ex_test)
```

[149]:
```
747427
```

[150]:
```
ratings_test=[]
for i in range(len(array_true_ex_test)):
  movie=array_true_ex_test[i][0]
  user=array_true_ex_test[i][1]
  ratings_test.append(df2_explicit_dict[user][movie])
len(ratings_test)
```

[150]:
```
747427
```

[151]:
```
link_features_test_ex = link_examples_to_features(
        array_true_ex_test, embedding_test, operator_concat)
predicted_concat_test_ex = clf_concat_ex.predict(np.
 ↪array(link_features_test_ex))
predicted_concat_test_ex=predicted_concat_test_ex.flatten()
```

```
23358/23358 [==============================] - 27s 1ms/step
```

[152]:
```
predicted_concat_test_ex
```

[152]:
```
array([2.3273487, 4.325609 , 4.3286285, …, 3.8423944, 3.3535705,
       3.9380345], dtype=float32)
```

[153]:
```
loss_concat_test_ex=np.sum((ratings_test-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex)
loss_concat_test_ex
```

[153]:
```
1.4266383999617176
```

[154]:
```
RMSE=np.sqrt(np.sum((np.array(ratings_test)-predicted_concat_test_ex)**2)/
 ↪len(predicted_concat_test_ex))
RMSE
```

[154]:
```
1.1944196917171608
```

[155]:
```
MAPE=np.sum(np.abs((np.array(ratings_test)-predicted_concat_test_ex))/
 ↪ratings_test)/len(predicted_concat_test_ex)
MAPE
```

[155]:
```
0.3468014838950562
```