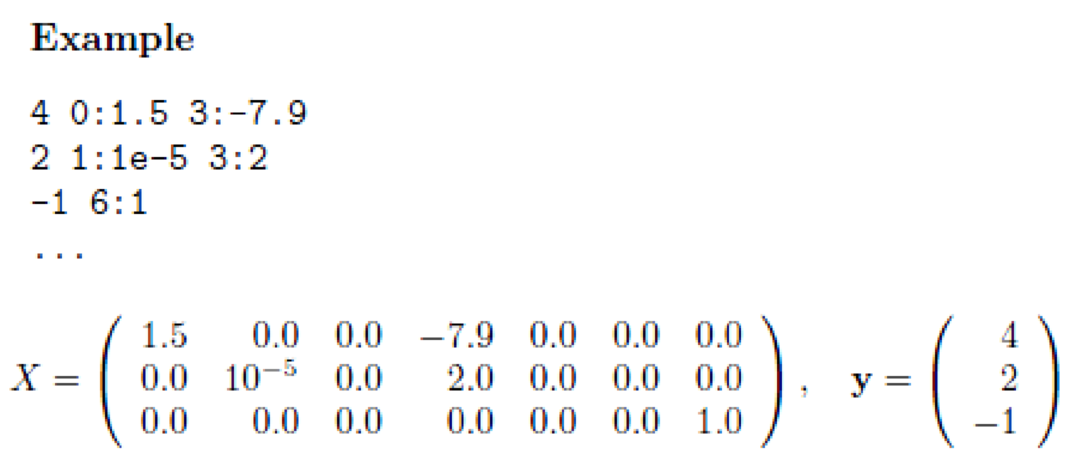
convert 를 하면 나오는 data의 형식

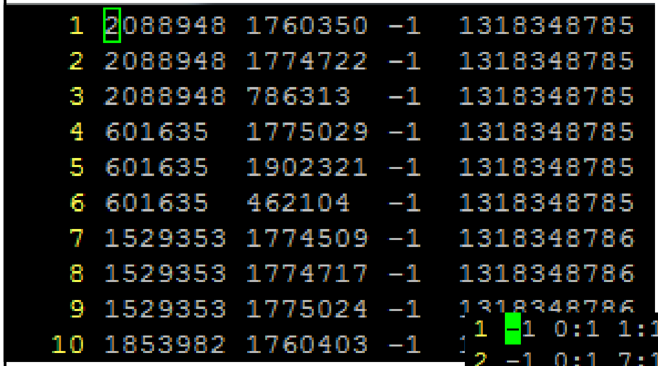


Example

```
4 0:1.5 3:-7.9
2 1:1e-5 3:2
-1 6:1
...
```

$$X = \begin{pmatrix} 1.5 & 0.0 & 0.0 & -7.9 & 0.0 & 0.0 & 0.0 \\ 0.0 & 10^{-5} & 0.0 & 2.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} 4 \\ 2 \\ -1 \end{pmatrix}$$

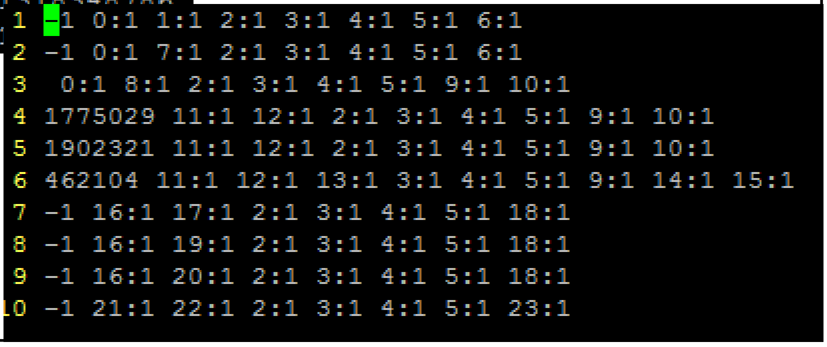
기존 rec_log_train.txt



```
2088948 1760350 -1  1318348785
2088948 1774722 -1  1318348785
2088948 786313  -1  1318348785
601635  1775029 -1  1318348785
601635  1902321 -1  1318348785
601635  462104  -1  1318348785
1529353 1774509 -1  1318348786
1529353 1774717 -1  1318348786
1529353 1775024 -1  1318348786
1853982 1760403 -1
```

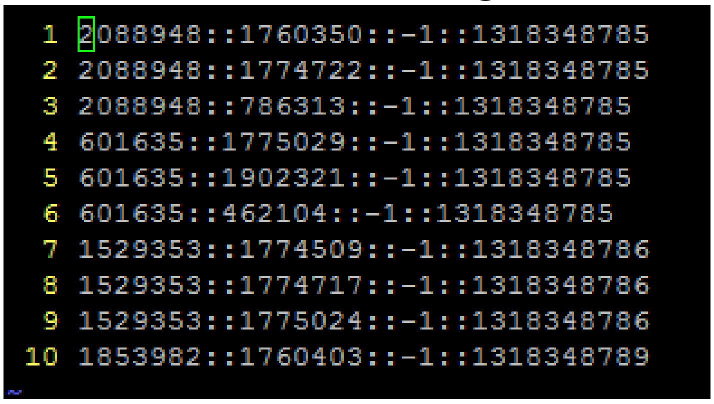
convert

변환이 잘못됨



```
-1 0:1 1:1 2:1 3:1 4:1 5:1 6:1
-1 0:1 7:1 2:1 3:1 4:1 5:1 6:1
 0:1 8:1 2:1 3:1 4:1 5:1 9:1 10:1
1775029 11:1 12:1 2:1 3:1 4:1 5:1 9:1 10:1
1902321 11:1 12:1 2:1 3:1 4:1 5:1 9:1 10:1
462104 11:1 12:1 13:1 3:1 4:1 5:1 9:1 14:1 15:1
-1 16:1 17:1 2:1 3:1 4:1 5:1 18:1
-1 16:1 19:1 2:1 3:1 4:1 5:1 18:1
-1 16:1 20:1 2:1 3:1 4:1 5:1 18:1
-1 21:1 22:1 2:1 3:1 4:1 5:1 23:1
```

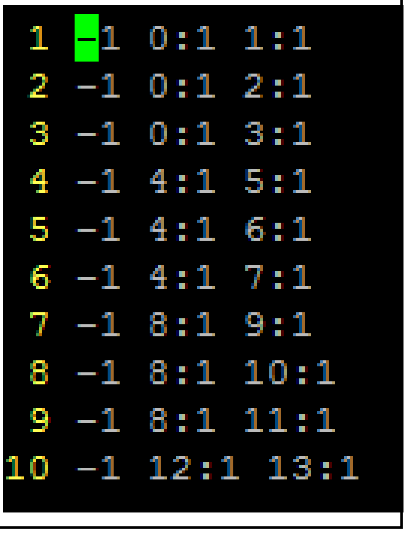
delimiter를 넣은 rec_log_train.txt



```
2088948::1760350::-1::1318348785
2088948::1774722::-1::1318348785
2088948::786313::-1::1318348785
601635::1775029::-1::1318348785
601635::1902321::-1::1318348785
601635::462104::-1::1318348785
1529353::1774509::-1::1318348786
1529353::1774717::-1::1318348786
1529353::1775024::-1::1318348786
1853982::1760403::-1::1318348789
~
```

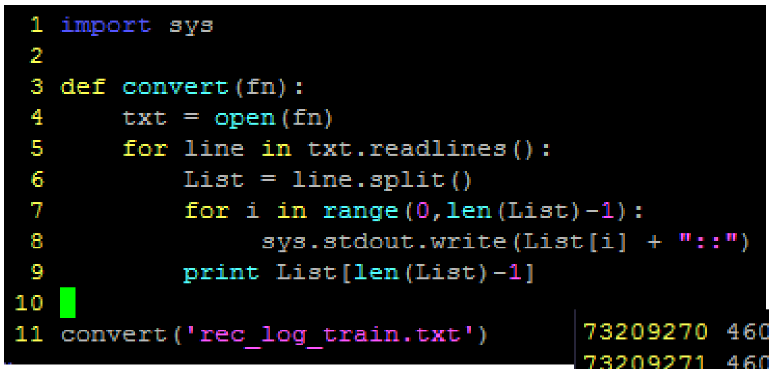
convert

변환이 잘됨



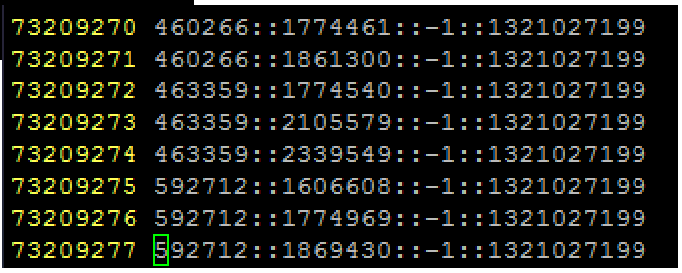
```
-1 0:1 1:1
-1 0:1 2:1
-1 0:1 3:1
-1 4:1 5:1
-1 4:1 6:1
-1 4:1 7:1
-1 8:1 9:1
-1 8:1 10:1
-1 8:1 11:1
-1 12:1 13:1
```

convert 를 하기위한 converter를 만듦



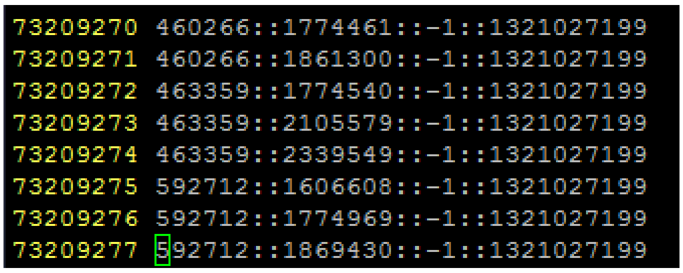
```python
import sys

def convert(fn):
    txt = open(fn)
    for line in txt.readlines():
        List = line.split()
        for i in range(0,len(List)-1):
            sys.stdout.write(List[i] + "::")
        print List[len(List)-1]

convert('rec_log_train.txt')
```

rec_log_train.txt가
끝까지 잘 변환됨



```
73209270 460266::1774461::-1::1321027199
73209271 460266::1861300::-1::1321027199
73209272 463359::1774540::-1::1321027199
73209273 463359::2105579::-1::1321027199
73209274 463359::2339549::-1::1321027199
73209275 592712::1606608::-1::1321027199
73209276 592712::1774969::-1::1321027199
73209277 592712::1869430::-1::1321027199
```

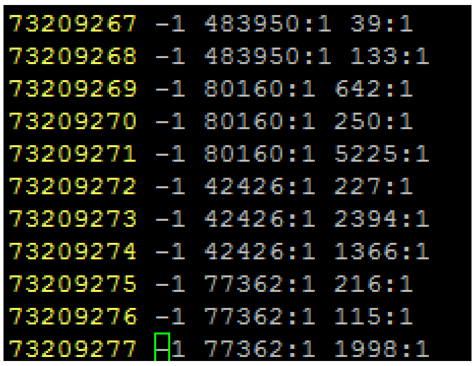
기존 rec_log_train.txt



```
73209270 460266::1774461::-1::1321027199
73209271 460266::1861300::-1::1321027199
73209272 463359::1774540::-1::1321027199
73209273 463359::2105579::-1::1321027199
73209274 463359::2339549::-1::1321027199
73209275 592712::1606608::-1::1321027199
73209276 592712::1774969::-1::1321027199
73209277 592712::1869430::-1::1321027199
```

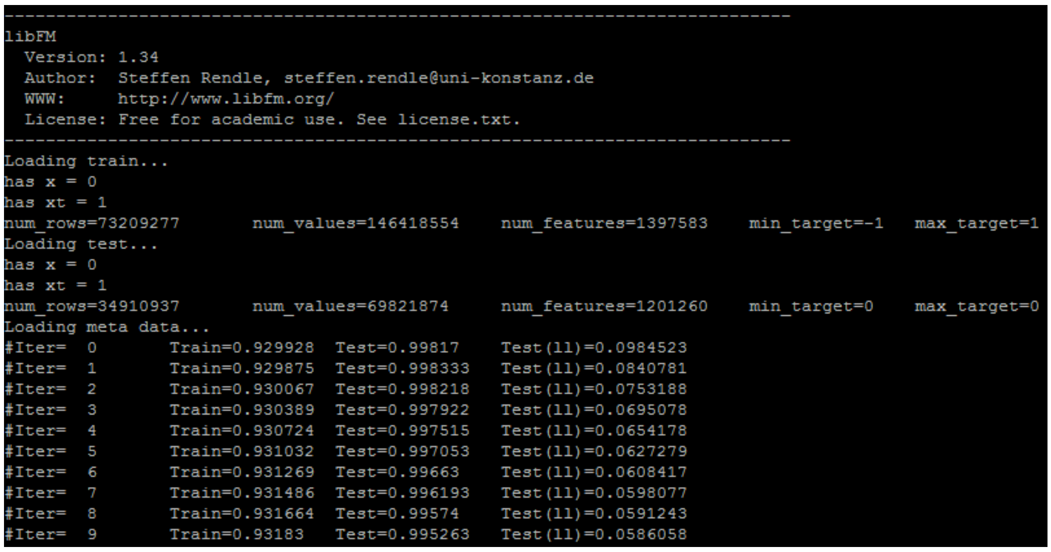
convert

변환이 잘됨



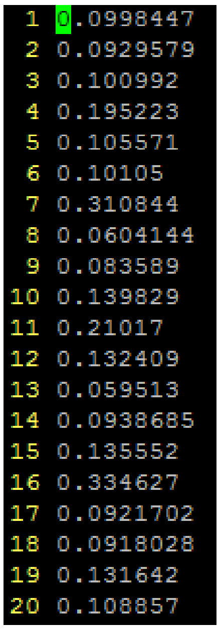
```
73209267 -1 483950:1 39:1
73209268 -1 483950:1 133:1
73209269 -1 80160:1 642:1
73209270 -1 80160:1 250:1
73209271 -1 80160:1 5225:1
73209272 -1 42426:1 227:1
73209273 -1 42426:1 2394:1
73209274 -1 42426:1 1366:1
73209275 -1 77362:1 216:1
73209276 -1 77362:1 115:1
73209277 -1 77362:1 1998:1
```

rec_log_test도 변환한 후 libFM에서 돌려봄



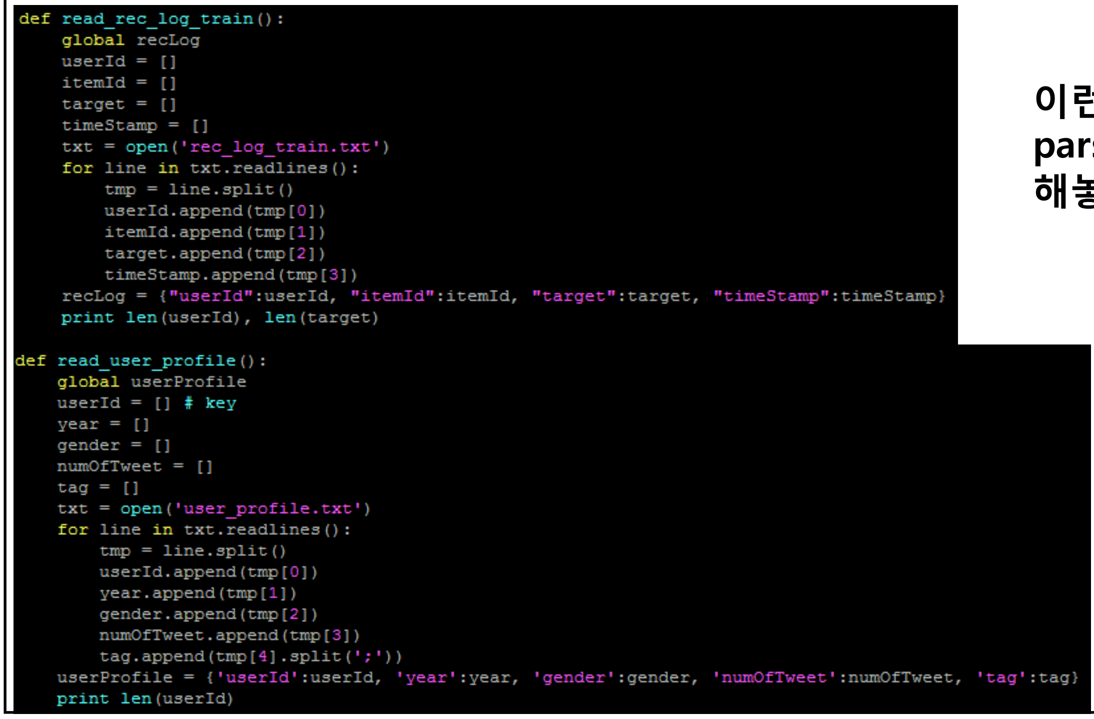
```
----------------------------------------------------------------------------
libFM
  Version: 1.34
  Author:  Steffen Rendle, steffen.rendle@uni-konstanz.de
  WWW:     http://www.libfm.org/
  License: Free for academic use. See license.txt.
----------------------------------------------------------------------------
Loading train...
has x = 0
has xt = 1
num_rows=73209277       num_values=146418554    num_features=1397583    min_target=-1   max_target=1
Loading test...
has x = 0
has xt = 1
num_rows=34910937       num_values=69821874     num_features=1201260    min_target=0    max_target=0
Loading meta data...
#Iter=  0       Train=0.929928  Test=0.99817    Test(ll)=0.0984523
#Iter=  1       Train=0.929875  Test=0.998333   Test(ll)=0.0840781
#Iter=  2       Train=0.930067  Test=0.998218   Test(ll)=0.0753188
#Iter=  3       Train=0.930389  Test=0.997922   Test(ll)=0.0695078
#Iter=  4       Train=0.930724  Test=0.997515   Test(ll)=0.0654178
#Iter=  5       Train=0.931032  Test=0.997053   Test(ll)=0.0627279
#Iter=  6       Train=0.931269  Test=0.99663    Test(ll)=0.0608417
#Iter=  7       Train=0.931486  Test=0.996193   Test(ll)=0.0598077
#Iter=  8       Train=0.931664  Test=0.99574    Test(ll)=0.0591243
#Iter=  9       Train=0.93183   Test=0.995263   Test(ll)=0.0586058
```

결과 file



```
0.0998447
0.0929579
0.100992
0.195223
0.105571
0.10105
0.310844
0.0604144
0.083589
0.139829
0.21017
0.132409
0.059513
0.0938685
0.135552
0.334627
0.0921702
0.0918028
0.131642
0.108857
```

결과 file중 0.978445가 가장 큰 숫자였음



```
def read_rec_log_train():
    global recLog
    userId = []
    itemId = []
    target = []
    timeStamp = []
    txt = open('rec_log_train.txt')
    for line in txt.readlines():
        tmp = line.split()
        userId.append(tmp[0])
        itemId.append(tmp[1])
        target.append(tmp[2])
        timeStamp.append(tmp[3])
    recLog = {"userId":userId, "itemId":itemId, "target":target, "timeStamp":timeStamp}
    print len(userId), len(target)

def read_user_profile():
    global userProfile
    userId = [] # key
    year = []
    gender = []
    numOfTweet = []
    tag = []
    txt = open('user_profile.txt')
    for line in txt.readlines():
        tmp = line.split()
        userId.append(tmp[0])
        year.append(tmp[1])
        gender.append(tmp[2])
        numOfTweet.append(tmp[3])
        tag.append(tmp[4].split(';'))
    userProfile = {'userId':userId, 'year':year, 'gender':gender, 'numOfTweet':numOfTweet, 'tag':tag}
    print len(userId)
```

이런식으로
parsing만
해놓음