Discovering Movie Categories Based on SPHK-means Clustering Algorithm

Dong-Yuan LI[1] and Cai-Feng CAO[1,a]

[1]*Wuyi University, School of Computer, 529020 Jiangmen, China*

Abstract. Basing on SPHK-means, an improved K-means clustering algorithm, we have used dataset provided by MovieLens to design experiment. First, we have reduced dimensions of movie-user scoring matrix. Then, we have multiply sampled movies to conduct agglomerative hierarchical clustering in order to determine the appropriate value of k and initial centers. Finally, according to fixed k and initial centers, we have divided movies into groups through K-means clustering. With evaluation indicators as precision, recall and number of groups found, experiment in this paper has indicated that result of SPHK-means clustering algorithm is better than that of classical K-means clustering algorithm.

1 Introduction

Our anther paper for this conference, *An Improved K-means Clustering Algorithm Applicable to Massive High-dimensional Matrix Datasets*(IST1810), has proposed SPHK-means clustering algorithm, and has analyzed in theory that it may be better than classical K-means clustering algorithm in aspect of classification accuracy and number of categories found, and that it was better than classical agglomerative hierarchical clustering algorithm in aspect of time complexity and space complexity. SPHK-means clustering algorithm further integrates sampling and principal component analysis, which makes it more applicable for massive high-dimensional datasets.

Due to restriction of article length, in this paper, we would not describe the model of SPHK-means clustering algorithm in detail. Instead, we herein have designed experiment using datasets provided by MovieLens [1] to verify that SPHK-means clustering algorithm is better than classical K-means clustering algorithm in aspect of classification accuracy and number of categories found.

2 Introduction of datasets

MovieLens, a movie recommendation system, has recommended MovieLens Latest Datasets for education and development. Our experiment has used MovieLens Latest Datasets (small) updated at January 16th 2016.

Ratings that 668 users scored 10325 movies have been used as training dataset, a fraction of training dataset are showed in Table 1. Style tags of 10325 movies have been used as testing dataset, a fraction of testing dataset are showed in Table 2.

[a] Corresponding author: cfcao@126.com

Table 1. A fraction of training dataset.

userId	movieId	rating
1	16	4
1	24	1.5
2	17	5
2	784	2
⋮	⋮	⋮
668	144976	2.5
668	148626	4.5

Table 2. A fraction of testing dataset.

movieId	genres
5	Comedy
9	Action
14	Drama
⋮	⋮
148238	Comedy
148626	Drama

3 Implementation of SPHK-means clustering algorithm

Experiment environment is Windows 10. We have used python 2.7 and modules including 'csv', 'numpy', 'scipy', 'math' and 'matplotlib'. The process of our experiment is showed in Figure 1.

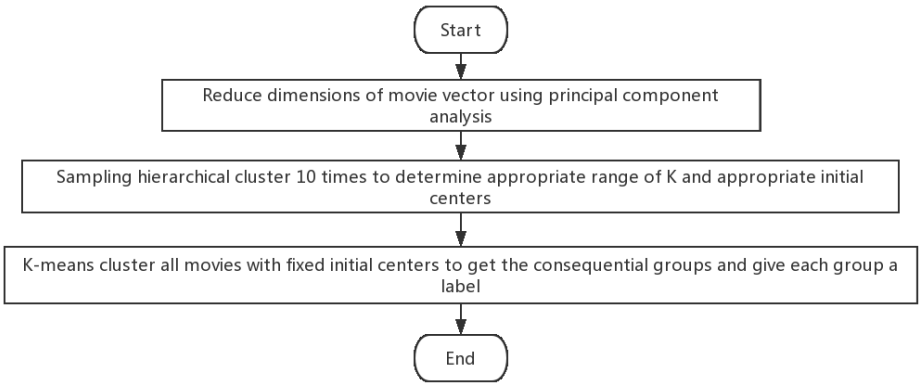


Figure 1. The process of our experiment.

Major portion of functional codes are specifically as follows.

3.1 Reduce dimensions of movie vector using principal component analysis[2]

Function pca(dataMat, topNfeat=999999) realize principal component analysis and reducing dimensions, dataMat is dataset that PCA deals with, topNfeat, number of features extracted, is optional. Its codes are as follows:

```
def pca(dataMat, topNfeat=999999):
meanVals = mean(dataMat, axis=0)
meanRemoved = dataMat - meanVals # Subtract the mean value
covMat = cov(meanRemoved, rowvar=0)
eigVals,eigVects = linalg.eig(mat(covMat))
eigValInd = argsort(eigVals)            # Sort from smallest to largest
eigValInd = eigValInd[:-(topNfeat+1):-1]  # Remove unneeded dimensions
redEigVects = eigVects[:,eigValInd]       # From large to small restructure characteristic vector
lowDDataMat = meanRemoved * redEigVects  # Transform the data into the new space
reconMat = (lowDDataMat * redEigVects.T) + meanVals
return lowDDataMat, reconMat
```

3.2 Sampling hierarchical clustering determines appropriate range of K and appropriate Initial centers

Codes are as follows:

```
sampsize =2000 # sample capacity
samp = random.randint(len(mId1), size=(10,sampsize)) # independently sample 10 times
ncH = []
print '--------- Sampling PCA hierarchical clustering determine the appropriatevalue of K ----------'
for sampi in samp:
    radR = []
    radC = []
    radD = []
    iR = -1
    for j in sampi:
        iR += 1
        for k in range(len(uid)):
            if M[j,k]!=0 :
                radR.append(iR)
                radC.append(k)
                radD.append(M[j,k])
    rR = np.array(radR)
    rC = np.array(radC)
    rD = np.array(radD)
    m = sparse.coo_matrix((rD,(rR,rC)), shape=(sampsize,len(uid))).todense()
    mp = pca(m,100)[0] #PCA, reduce dimensions
    Z = linkage(mp.real,'average','correlation')
    cluH = fcluster(Z+abs(Z.min()), 1.1547)
    print cluH.max()#number of groups of sampling hierarchical clustering
    ncH.append(cluH.max())
print '------------------------------'
k1 = np.mean(ncH)-np.std(ncH)/2
k2 = np.mean(ncH)+np.std(ncH)/2
print ' Upper limit of appropriate range: %d' % k1
print ' Lower limit of appropriate range: %d' % k2
print '------------------------------'
print '----------------results of SPHK-means clustering algorithm ---------------'
k = -1000
while (k<k1)|(k>k2) : # until k in appropriate range
    samp1 = random.randint(len(mId1), size=(1,sampsize))
    radR = []
```

```
    radC = []
    radD = []
    iR = -1
    for i in samp1[0]:
        iR += 1
        for j in range(len(uid)):
            if M[i,j]!=0 :
                radR.append(iR)
                radC.append(j)
                radD.append(M[i,j])
    rR = np.array(radR)
    rC = np.array(radC)
    rD = np.array(radD)
    m = sparse.coo_matrix((rD,(rR,rC)), shape=(sampsize,len(uid))).todense()
    mp = pca(m,100)[0]
    Z = linkage(mp.real,'average','correlation')
    cluH1 = fcluster(Z+abs(Z.min()), 1.1547)
    print cluH1.max()
    k = cluH1.max()
print ' divide all movies into %d categories (repeated): ' % k
code = []
for i in range(k) :
    sumC = np.zeros(len(uid))
    numC = 0
    for j in range(sampsize):
        if cluH1[j] == i+1 :
            sumC = sumC+np.array(m[j].tolist()[0])
            numC += 1
    code.append((sumC/numC).tolist())
code_book = np.array(code) #appropriate initial centers
```

3.3 K-means clustering according to fixed K and initial centers

Function kmeans1(M, k, code_book, iter=10, thresh=1e-05) realize 3) K-means clustering with fixed k and initial centers, M is matrix to deal with, code_book is initial centers. Its codes is as follows:

```
def kmeans1(M, k, code_book, iter=10, thresh=1e-05):
    ii = 0
    while ii<iter :
        kind = vq(M,code_book)[0].tolist() #groups each movie belong to
        kind1 = []
        kind1.extend(kind)
        kindex = [] # index of movies in every group in 'kind'
        for i in range(k) :
            iindex = []
            while i in kind1 :
                iindex.append(kind1.index(i))
                kind1[kind1.index(i)] = k
            kindex.append(iindex)
        code = []
        lastCode = code_book.tolist()
        for i in range(k) :
          sumC = np.zeros(len(uid))
```

```
            numC = 0
            for j in range(M.shape[0]):
                if kind[j] == i :
                    sumC = sumC+np.array(M[j].tolist()[0])
                    numC += 1
            with np.errstate(invalid='ignore'):
                code.append((sumC/numC).tolist())
        for i in range(k) :
            if np.sqrt(((np.array(code[i])-np.array(lastCode[i]))**2).sum()) >= thresh :
                lastCode = code
                break
            if i == k-1 :
                code_book = np.array(code)
                return code_book
        ii += 1
    code_book = np.array(code)
return code_book, kindex
```

4 Experimental results

4.1 Evaluation indexes

In statistics, precision, recall and F-score are typical evaluation indexes for classification problem. If result of classification is as Figure 2[3], precision and recall are defined as formula (1) and formula (2).

$$\text{Precision}=\frac{|\text{true positives}|}{|\text{true positives}|+|\text{false positives}|} \tag{1}$$

$$\text{Recall}=\frac{|\text{true positives}|}{|\text{true positives}|+|\text{false negatives}|} \tag{2}$$

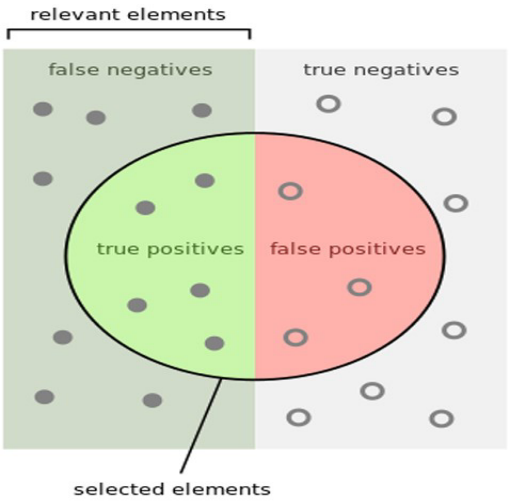


Figure 2[3]. Description precision and recall.

F-score, defined as formula (3), is a measure that combines precision and recall.

$$F=2\cdot\frac{\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}} \tag{3}$$

The higher the evaluation indexes (precision, recall and F-score) are, the better classification accuracy is.

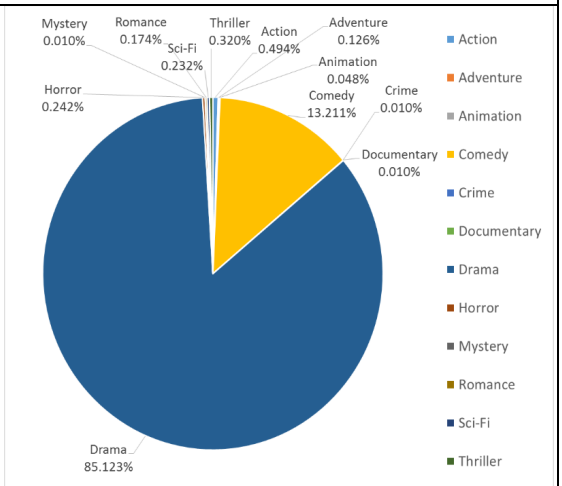
4.2 Data analysis of experimental results

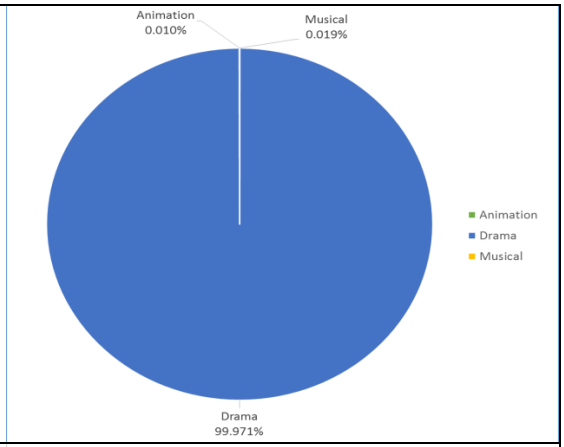
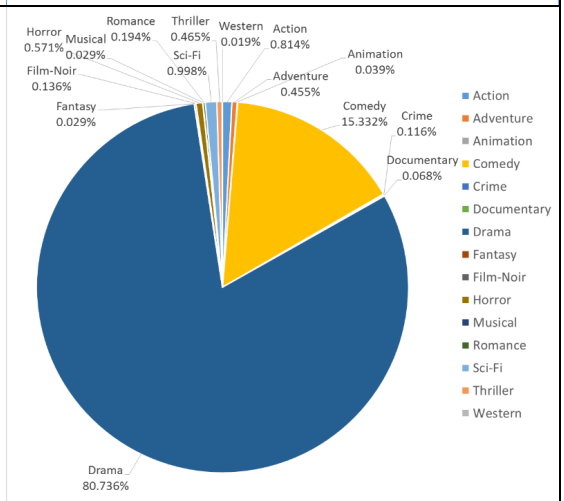
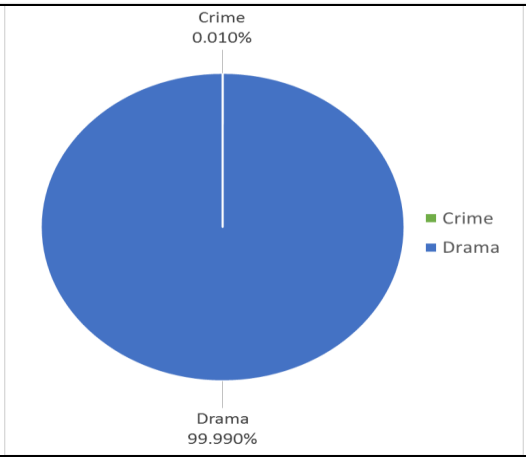
At the same ambient conditions, we have repetitive repeatedly experimented in accordance with SPHK-means clustering algorithm and classical K-means clustering algorithm respectively. Data of experimental results is showed in Table 3 and Table 4.

Table 3. Comparison of evaluation indexes of SPHK-means and classical K-means.

Order	Algorithm	precision	recall	F-score
1	SPHK-means	53.3837%	56.3688%	54.8357%
	Classical K-means	50.5373%	53.3633%	51.9119%
2	SPHK-means	55.0198%	58.0965%	56.5163%
	Classical K-means	50.5373%	53.3633%	51.9119%
3	SPHK-means	54.5842%	57.6365%	56.0688%
	Classical K-means	50.5276%	53.3531%	51.9019%

Table 4. Comparison of movie categories found of SPHK-means and classical K-means.

No.	Algorithm	Movie categories found and amount of each category	Diagram: Pie Chart
1	SPHK-means	12 categories found: Action: 51; Adventure: 13; Animation: 5; Comedy: 1365; Crime: 1; Documentary: 1; Drama: 8789; Horror: 25; Mystery: 1; Romance: 18; Sci-Fi: 24; Thriller: 33.	

	Classical K-means	3 categories found: Animation: 1; Drama: 10322; Musical: 2.	
2	SPHK-means	15 categories found: Action: 84; Adventure: 47; Animation: 4; Comedy: 1583; Crime: 12; Documentary: 7; Drama: 8336; Fantasy: 3; Film-Noir: 14; Horror: 59; Musical: 3; Romance: 20; Sci-Fi: 103; Thriller: 48; Western: 2.	
	Classical K-means	2 categories found: Crime: 1; Drama: 10324.	

In contrast, we can find it from Table 3 that precision and recall of SPHK-means clustering algorithm are both higher than those of K-means clustering algorithm.

In the test data set, movies are actually divided into 19 categories: Action, Adventure, Animation, Children, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, IMAX, Musical, Mystery, Romance, Sci-Fi, Thriller, War and Western. Table 4 has shown that movie categories found by SPHK-means clustering algorithm are greatly more than those found by classical K-means clustering algorithm.

Conclusion

In this paper, through experiment, we have verified that SPHK-means clustering algorithm is better than classical K-means clustering algorithm in aspect of classification accuracy and number of categories found.

Next, in order to adapt SPHK-means clustering algorithm to more scenarios, we will consider situation that one object may belong to two or more categories which require us to integrate membership degree[4] mentioned in fuzzy mathematics into it.

Acknowledgment

This paper is supported by the Features Innovative Project in Guangdong Province “Research and Application of Mass Media Data Mining Technology” (2015KTSCX145).

References

1. http://grouplens.org/datasets/movielens/ (Jan.16th 2016)
2. P. Harrington, Machine Learning (Posts &Telecom Press, Beijing,2013)
3. https://en.wikipedia.org/wiki/Precision_and_recall
4. C. M.Zhang, Sw.G, **15**, 41-43 (2016)