深 圳 大 学 实 验 报 告

课程编号：**2801000049**

课程名称：机器学习

实验项目名称：**Task 4 matrix fact clustering**

学院：电子与信息工程学院

专业：电子信息工程

指导教师：麦晓春

报告人：古炜 学号：**2022280327** 班级：文华班

实验时间：**2024 年 6 月 14 日至 6 月 28 日**

实验报告提交时间：**2024 年 6 月 28 日**

教务部制

1. Experimental Purposes and Requirements

1. Load the dataset seeds.csv.

2. Extract the grain variety and the feature data, and transform the feature data into a numpy array.

3. Run the non-negative matrix decomposition on the data multiple times (rank=3, repeat_times=20).

4. Calculate the Silhouette coefficients for each clustering result, and plot a line graph, labeling the best scores.

5. Transform the clustering results corresponding to the best score into a Dataframe, and make a crosstab of the varieties and the best clustering results pd.crosstab(df1, df2).

2. Experiment Contents and Process

2.1 Experiment 1

(1) Experimental Contents/Introduction

- **NMF (Non-negative Matrix Factorization)**

For any given non-negative matrix V, it can find a non-negative matrix W and a non-negative matrix H, satisfying the condition V=W*H, thus decomposing a non-negative matrix into the product of two non-negative matrices. In this case, each column in the V matrix represents an observation, and each row represents a feature; W matrix is referred to as the basis matrix, while H matrix is referred to as the coefficient matrix or weight matrix. By replacing the original matrix with the coefficient matrix H, it is possible to achieve dimensionality reduction of the original matrix, obtaining a reduced-dimensional matrix of data features, thereby reducing storage space.

$$\mathrm{V}_{\mathrm{m\times n}} \approx \mathrm{W}_{\mathrm{m\times k}} \times \mathrm{H}_{\mathrm{k\times n}} = \hat{\mathrm{V}}_{\mathrm{m\times n}}$$

$$\mathrm{W}_{\mathrm{m\times k}} \geqslant 0$$

$$\mathrm{H}_{\mathrm{k\times n}} \geqslant 0$$

Loss function:

- Square distance:

$$\|\mathrm{A}-\mathrm{B}\|^2 = \sum_{\mathrm{i,j}} (\mathrm{A}_{\mathrm{i,j}} - \mathrm{B}_{\mathrm{i,j}})^2$$

- KL divergence:

$$\mathrm{D}(\mathrm{A}\,\|\,\mathrm{B}) = \sum_{\mathrm{i,j}} \left(A_{i,j} \log \frac{A_{i,j}}{B_{i,j}} - A_{i,j} + B_{i,j} \right)$$

In the definition of KL divergence, D (A || B) $\geqslant$ 0 is obtained if and only if A=B.

After defining the loss function, the problem that needs to be solved becomes the following form, corresponding to different loss functions:

Solve the following minimization problem:

$$\mathrm{minimize}\|\mathrm{V}-\mathrm{WH}\|^2$$

$$\mathrm{s.t.W} \geqslant 0, \mathrm{H} \geqslant 0$$

$$\text{minimize D (V || WH)}$$

$$\text{s.t.W} \geqslant 0, \text{H} \geqslant 0$$

Solving optimization problems:
For the loss function of square distance:

$$\mathrm{W_{i,k}} = \mathrm{W_{i,k}} \frac{(VH^T)_{\mathrm{i,k}}}{(WHH^T)_{\mathrm{i,k}}}$$

$$\mathrm{H_{k,j}} = \mathrm{H_{k,j}} \frac{(W^T V)_{\mathrm{k,j}}}{(W^T WH)_{\mathrm{k,j}}}$$

For the loss function of KL divergence:

$$\mathrm{W_{i,k}} = \mathrm{W_{i,k}} \frac{\sum_{\mathrm{u}} \mathrm{H_{k,u}} \mathrm{V_{i,u}} / (\mathrm{WH})_{\mathrm{i,u}}}{\sum_{\mathrm{v}} \mathrm{H_{k,v}}}$$

$$\mathrm{H_{k,j}} = \mathrm{H_{k,j}} \frac{\sum_{\mathrm{u}} \mathrm{W_{u,k}} \mathrm{V_{u,j}} / (\mathrm{WH})_{\mathrm{u,j}})}{\sum_{\mathrm{v}} \mathrm{W_{v,k}}}$$

The above multiplication rules are mainly designed to ensure non negativity during the calculation process, while in gradient descent based methods, addition and subtraction operations cannot guarantee non negativity. In fact, the above multiplication update rules are equivalent to gradient descent based algorithms. The following uses square distance as the loss function to illustrate the equivalence of the above process:

The square loss function can be written as:

$$\mathrm{l} = \sum_{\mathrm{i}=1}^{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{n}} \left[V_{i,j} - \left(\sum_{k=1}^{r} W_{i,k} \cdot H_{k,j} \right) \right]^2$$

Following the idea of gradient descent method:

$$\mathrm{H_{k,j}} = \mathrm{H_{k,j}} - \eta_{\mathrm{k,j}} \frac{\partial \mathrm{l}}{\partial \mathrm{H_{k,j}}}$$

Mean:

$$\mathrm{H_{k,j}} = \mathrm{H_{k,j}} + \eta_{\mathrm{k,j}} \left[(\mathrm{W^T V})_{\mathrm{k,j}} - (\mathrm{W^T WH})_{\mathrm{k,j}} \right]$$

Let $\eta_{\mathrm{k,j}} = \frac{\mathrm{H_{k,j}}}{(\mathrm{W^T WH})_{\mathrm{k,j}}}$, which gives the form of the multiplication update rule mentioned above.

- Silhouette Coefficient

The Silhouette Coefficient is an indicator used to evaluate the effectiveness of clustering. It can be understood as an indicator that describes the clarity of the contours of each category after clustering. It contains two factors - cohesion and separation.

Cohesion can be understood as reflecting the degree of closeness between a sample point

and its intra class elements.

Separation can be understood as reflecting the degree of closeness between a sample point and elements outside the class.

The formula for Silhouette Coefficient is as follows:

$$S(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Among them, a(i) represents the cohesion of the sample points, calculated as follows:

$$a(i) = \frac{1}{n-1} \sum_{j \neq i}^{n} distance(i,j)$$

Where j represents other sample points in the same class as sample i, and distance represents the distance between i and j. So the more novel a(i) is, the closer it becomes to this category.

The calculation method of b(i) is similar to that of a (i). Just need to traverse other clusters to obtain multiple values {$b_1(i)$, $b_2(i)$, $b_3(i)$, Choose the smallest value from $b_m(i)$} as the final result.

So the original S(i):

$$S(i) = \begin{cases} 1 - \frac{a(i)}{b(i)} & a(i) < b(i) \\ 0 & a(i) = b(i) \\ \frac{b(i)}{a(i)} - 1 & a(i) > b(i) \end{cases}$$

From the above equation, it can be found that:

When a(i)<b(i), i.e. the intra class distance is smaller than the inter class distance, the clustering results are more compact. The value of S will approach 1. The closer it approaches 1, the more obvious the contour becomes.

On the contrary, when a(i)>b(i), the intra class distance is greater than the inter class distance, indicating that the clustering results are very loose. The value of S will approach -1, and the closer it approaches -1, the worse the clustering effect.

From this, it can be concluded that, the range of values for the contour coefficient S is [-1, 1], and the larger the contour coefficient, the better the clustering effect.

(2) Experiment process

This experiment uses Non-Negative Matrix Factorization (NMF) to cluster grain datasets, aiming to evaluate clustering quality by calculating contour coefficients. Firstly, load data from a CSV file, extract grain types and feature data, and convert them into a numpy array. Then, set the parameters of NMF, including decomposition into fractions, number of repeated runs, and maximum number of iterations. Run NMF multiple times, randomly initialize each time, extract decomposition results, and assign clustering based on the highest component weight. Next, calculate the contour coefficients for each clustering and store the results. Draw contour coefficient line graphs for multiple experiments and label the best results. Finally, determine the optimal clustering results and create a cross table between grain types and clustering results to analyze the distribution of different types of grains in each cluster.

Flow chart:

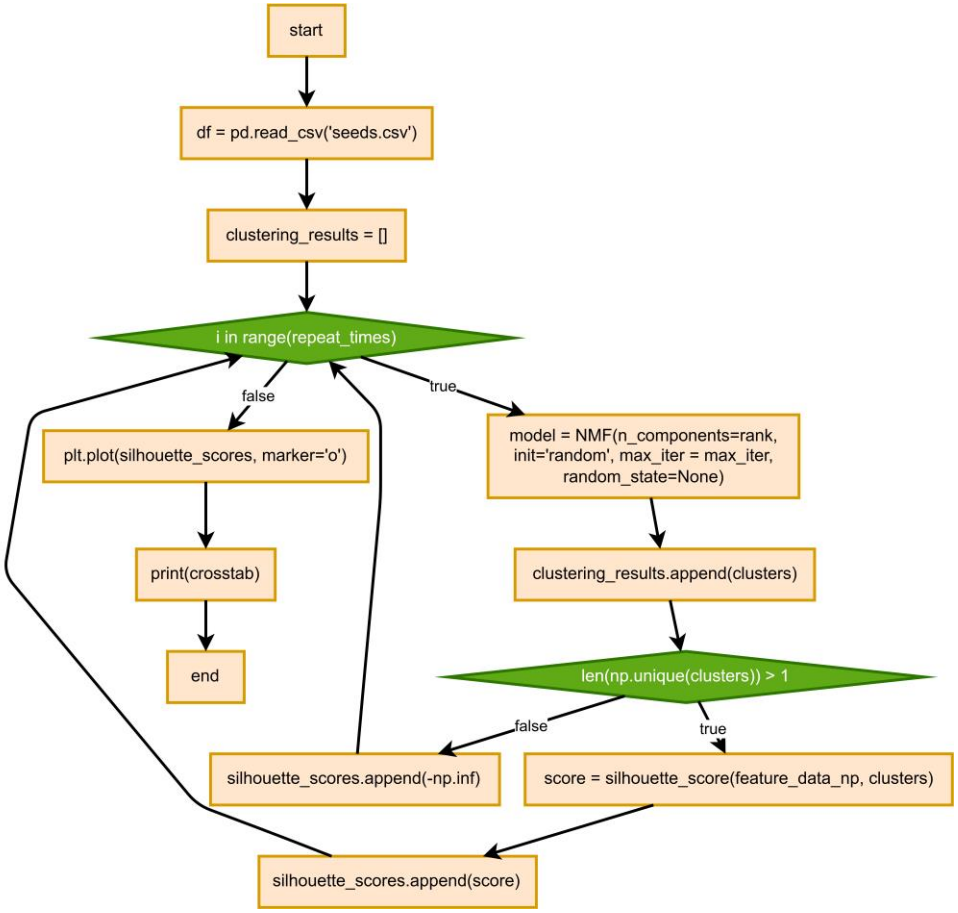

Figure 1 Flow chart

(3) Experimental Results and Analysis

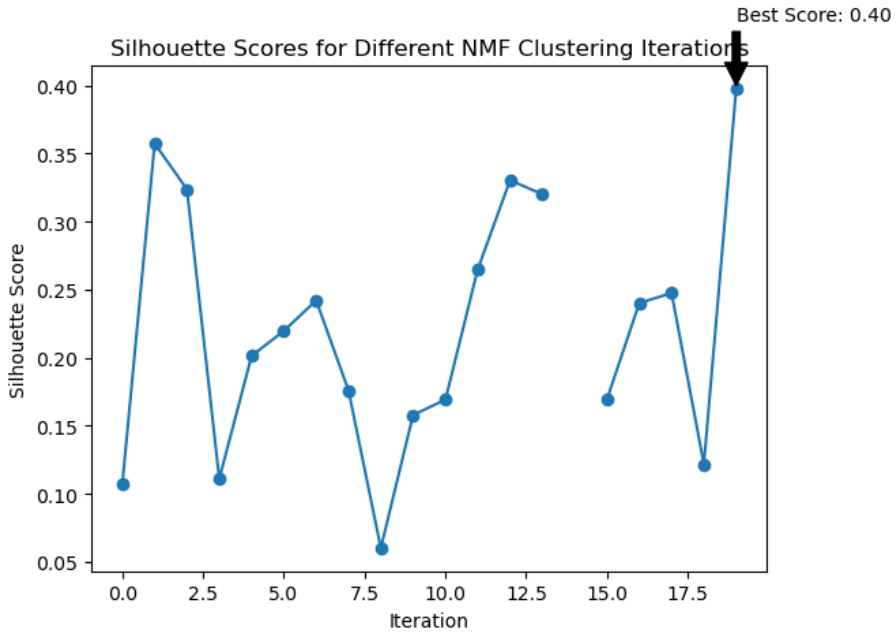

Figure 2 Silhouette Scores for Different NMF Clustering Iterations

This figure shows the contour coefficient values for each iteration after running the NMF algorithm multiple times. Through this graph, the clustering effect of each iteration can be visually observed, and the optimal contour coefficient can be determined.

Table 1 the crosstab of the varieties and the best clustering results

Cluster / grain_variety	0	1	2
Canadian wheat	0	68	2
Kama wheat	33	37	0
Rosa wheat	70	0	0

- Rosa wheat:

All 70 Rosa wheat samples were clustered to cluster 0. This consistency indicates that the features of Rosa wheat are very unique in this dataset and can be clearly recognized and separated by the NMF model.

- Canadian wheat:

68 samples were clustered to cluster 1, 2 samples were clustered to cluster 2, and no samples were clustered to cluster 0. Most Canadian wheat samples are concentrated in cluster 1, indicating that these samples are very close in the feature space. However, a small number of samples were assigned to cluster 2, which may be due to slight differences in certain features between these samples and other samples.

- Kama wheat:

33 samples were clustered to cluster 0, 37 samples were clustered to cluster 1, and no samples were clustered to cluster 2. The Kama wheat samples are distributed between cluster 0 and cluster 1, indicating that these samples have some similarity in the feature space, but there is no clear clustering like Rosa wheat and Canadian wheat. This may mean that Kama wheat's features overlap with Rosa wheat or Canadian wheat in some aspects.

3. Discussion and Conclusions

In this experiment, NMF was used to cluster the grain dataset and calculate the contour coefficient to evaluate the clustering effect. The experimental results showed that Rosa wheat has unique and consistent features that are easy to cluster. Canadian wheats are mainly distributed within a cluster, but there are small sample differences. Kama wheat has diverse characteristics and overlaps with other species. Cross table analysis further revealed the distribution of different grain types in each cluster, verifying the effectiveness and limitations of the NMF model on this dataset.

Through this experiment, I have learned the NMF algorithm and am able to apply it to practical data processing, mastering the relevant libraries and code implementation of the algorithm.

Appendix:

```
import pandas as pd
import numpy as np
from sklearn.decomposition import NMF
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

# 加载数据集
df = pd.read_csv('seeds.csv')

# 提取谷物种类和特征数据
grain_variety = df['grain_variety']
feature_data = df.drop(columns=['grain_variety'])

# 将特征数据转换为 numpy 数组
feature_data_np = feature_data.values

# 定义 NMF 参数
rank = 3
repeat_times = 20
max_iter = 1000

# 用于存储轮廓系数的占位符
silhouette_scores = []
clustering_results = []

# 多次运行 NMF 并计算轮廓系数
for i in range(repeat_times):
    model = NMF(n_components=rank, init='random', max_iter = max_iter, random_state=None)
    W = model.fit_transform(feature_data_np)
    H = model.components_

    # 根据最高成分权重分配聚类
    clusters = np.argmax(W, axis=1)
    clustering_results.append(clusters)

    # 检查聚类结果的唯一标签数
    if len(np.unique(clusters)) > 1:
        # 计算轮廓系数
        score = silhouette_score(feature_data_np, clusters)
        silhouette_scores.append(score)
    else:
```

```
            # 如果只有一个簇，分数为负无穷
            silhouette_scores.append(-np.inf)

# 绘制轮廓系数
plt.plot(silhouette_scores, marker='o')
plt.xlabel('Iteration')
plt.ylabel('Silhouette Score')
plt.title('Silhouette Scores for Different NMF Clustering Iterations')
plt.annotate(f'Best Score: {max(silhouette_scores):.2f}', xy=(np.argmax(silhouette_scores),
max(silhouette_scores)),
                    xytext=(np.argmax(silhouette_scores), max(silhouette_scores)+0.05),
                    arrowprops=dict(facecolor='black', shrink=0.05))
plt.show()

# 确定最佳聚类结果
best_index = np.argmax(silhouette_scores)
best_clusters = clustering_results[best_index]

# 将最佳聚类结果转换为 DataFrame
best_clusters_df = pd.DataFrame(best_clusters, columns=['Cluster'])

# 创建品种和最佳聚类结果的交叉表
crosstab = pd.crosstab(grain_variety, best_clusters_df['Cluster'])
print(crosstab)
```

指导教师批阅意见： 成绩评定： 指导教师签字： 年　　月　　日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。