Capstone Project

NON- PERSONALIZED BOOKS RECOMMENDATION SYSTEM

Group 9:

Vũ Công Duy - 20176737

Trần Công Minh - 20176825

Đào Hồng Quân - 20176850

Lê Đức Long - 20176806

CONTENTS:

I. Introduction

1. Project overview

Recommendation systems greatly boost sales for corporations. However, most systems are user based. Small retailers lack user data for this type of algorithm. Therefore, this project proposes a method and its enhancement for content based recommendation.

The main focus of this project is book recommendation, with book data crawled from the site goodreads.com. The data contains both categorical and numeric fields. Additionally, a book also has “tags”, which are the genres and opinions given by the readers.

The methods in this project, KNN and K-Medoids, are implemented to deal with both types of data.

2. Machine learning methods

- K Nearest Neighbours: K data points with the highest similarities will be returned as recommendation for the input.
- To deal with categorical data, K-Modes and K-Medoids are combined to give the best result.

3. Variables

- Two different tables are used in this project, with one containing the books’ basic information and the other containing each book’s list of tags, in other words, their content.
- There are 9 numeric and 5 categorical variables.

II. Data

1. Data source

3,000,000 books and the first 100 genre tags of each book for this project are crawled by using Scrapy from the website “goodreads.com” and saved in csv files.

After cleaning, only 450,000 books and 12 out of 303 million tags remained.

2. Features

- Numeric:
  - + Number of people tagging
  - + Number of total raters
  - + Number of raters in each star rating (1-5)
  - + Number of reviewers
  - + Number of pages
  - + Publication year
- Categorical:
  - + List of authors
  - + Book's series (if exists)
  - + List of tags
  - + Book's format
  - + Language

	BookID	Title	Author	Rating	5 stars	4 stars	3 stars	2 stars	1 star	Number of reviewers	Pages	Book format	Language	GenreLink	Series	PublishYear
0	3	Harry Potter and the Sorcerer's Stone	J.K. Rowling, Mary GrandPré	4.48	4868903	1704361	631378	148016	127122	118557	309	Hardcover	English	4640799	Harry Potter	1997
1	1	Harry Potter and the Half-Blood Prince	J.K. Rowling	4.57	1738165	611768	175048	29134	13230	41832	652	Paperback	English	41335427	Harry Potter	2005
2	7	The Harry Potter Collection	J.K. Rowling, Mary GrandPré	4.73	25140	4473	1111	230	282	921	318	Paperback	English	21457570	Harry Potter	2005
3	10	Harry Potter Collection	J.K. Rowling	4.73	25140	4473	1111	230	282	921	3342	Hardcover	English	21457570	Harry Potter	2005
4	5	Harry Potter and the Prisoner of Azkaban	J.K. Rowling, Mary GrandPré	4.57	2028235	706012	215277	29265	14292	58998	435	Mass Market Paperback	English	2402163	Harry Potter	1999

III. K-Nearest Neighbours recommendation

1. Problem statement

A quick way to recommend products without user data is by using the similarity in product features.

The input, the book's basic information and its list of tags, will be compared with all points in the dataset. The output are K books with the least dissimilarities.

Multiple calculation methods will be used to get the difference, feature by feature, and summed for the final value with a concluding method.

2. Dissimilarity calculations

2.1 Numeric difference:

For numeric features, we only get the difference by subtraction on the normalized values.

2.2 Categorical difference:

2.2.1 Binary difference:

For single value variables, language, series, and format, returning values 1 if equal and 0 otherwise is enough.

For the series field, if two books have the same value "None", which is having no series, the value should also be 0.

2.2.2 Jaccard dissimilarity:

For fields with multiple values, list comparison should be implemented. The most suitable method is Jaccard similarity, with the formula as follow:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

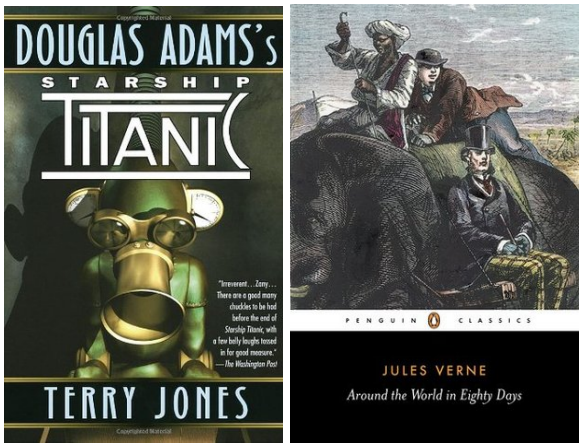
However, since we are trying to get the difference instead of similarity, the complement (1 - J) should be returned instead.

2.3 Euclidean distance:

$$d(\mathbf{p},\mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}.$$

To conclude the result, we use Euclidean distance, which is the square root of the sum of all squared differences, in this case, the values from numeric and categorical differences.

3. Results



The recommendations have similar content, review data, and especially language and format to the input.

The later features have high priority since the recommendations are for products, and clients usually buy books with the same language and format before deciding on the content.

As the size of the data set increases, the system becomes slower, since the computation is performed on all data points.

IV. K-Medoids

1. Problem statement

A book has not only its official genre but is also frequently branded with other tags by online readers, such as “favorite” or “to-read”. There can be a hidden pattern in books with similar tags. The authors and the book’s series are also factors in the readers’ trend.

However, K-Means clustering only works for numerical attributes. Therefore, a different clustering method is required.

2. K-Modes

2.1 Introduction

K-Modes is a modified version of K-Means where the centroid is determined by the mode of each attribute of the data rather than the means.

2.2 Algorithm

2.2.1 Distance measurement

Like in KNN, the series field uses binary difference while authors and genre (tags) use Jaccard dissimilarity.

2.2.2 Mode calculation

In each cluster, there will be a dictionary of frequency for authors, series, and tags.

The author and series with the highest frequency will be selected. The most common length determines the number of tags chosen by descending frequency order.

2.2.3 Algorithm

Step 1: K points are chosen randomly from the dataset and used as the original centroids.

Step 2: For each point in the dataset, the distance to each centroid is calculated. The point is assigned to the cluster of the nearest centroid. The frequency dictionary of the cluster is updated with the new point.

Step 3: The mode of each cluster is calculated based on its frequency dictionary.

Step 4: Calculate the distance between each cluster's old and new centroids. Repeat from step 2 until all distances become zeroes, or a certain number of iterations is reached.

In step 3, to increase efficiency, the centroid of each cluster is defined by the actual data point closest to the mode. This method is K-Medoids.

2.2.4 Inertia

To determine the compactness of a clustering model, the inertia is calculated by adding up the distances of all data points to their respective centroid.

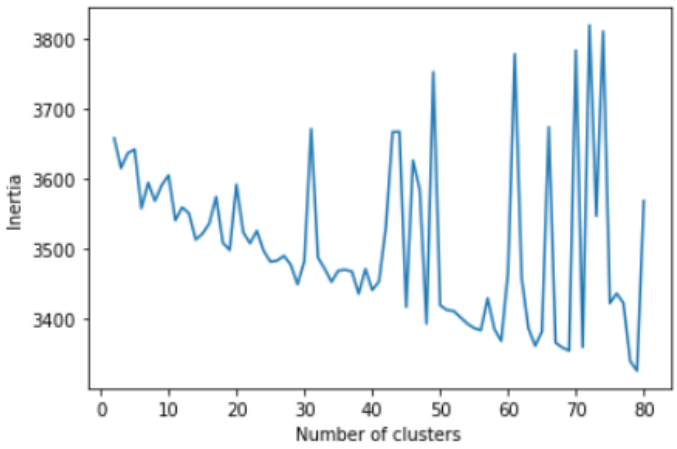
3. Implementation

The model is coded as a class object with the aim to reuse in future projects. Therefore, the functions mentioned beside the algorithm are passed as arguments rather than actually implemented in the class.

4. Results

Since the calculations are performed row by row, the runtime rises greatly with the increase of data points or number of clusters.

It took 16 hours to determine the inertias of models ranging from 2 to 80 clusters.



The result fluctuates with increasing intensity as the number of clusters grows. While the compactness improves with the rise in number, the model also gets higher probability for choosing outliers as original centroids, which damages the result.

The safest and most efficient number of clusters falls between 30 and 40, where the chance for noise interference is still low.

V. Recommendation system with clustering

1. Overview

The program is a combination of the K-Medoids model and KNN recommender.

2. Implementation

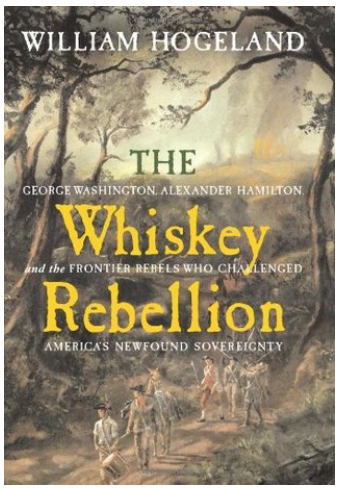
The model is coded as a recommender class containing a K-Medoids model. The functions mentioned in the K-Modes section are also used here.

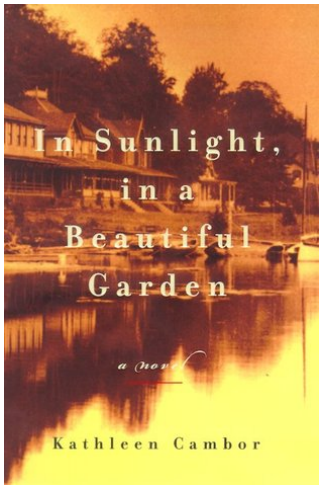
When the data sets are loaded to the system, only a proportion of the set will be used for clustering, the rest will be classified using the K-Medoids model. This helps reduce runtime for data fitting.

The input will be classified with a label and the K nearest points in the same cluster will be the recommendation.

For this system, there is a modification in dissimilarity calculation: the features authors, series, and genre are dropped as they have already been used for clustering.

3. Results





The recommendation now has closer content to the input.

VI. Overview

1. Data

This project is special in that it requires two different tables for calculation. An ID is used to link the two tables.

The models perform operations on pandas dataframes and series, which may cause difficulty in implementation in other projects.

2. K-Medoids Algorithm

This model is still very basic and has high runtime complexity, which is a major drawback.

This algorithm is simple enough for practical purposes without the black box problem.

3. Possible improvement

Each tag of a book also comes with the number of people tagging. This data can be used with the tags to create a form of corpus in the data set, and Jaccard dissimilarity can be replaced with TF-IDF and cosine similarity methods. This may improve runtime and give better insight into these features.

VII. References

Libraries:

- scikit-learn
- numpy
- pandas
- matplotlib