Contents

I Introduction **2**

i Main result 2

ii instruments . . . 2

II About **2**

i Reviewing abstracts and analogues 2

ii Project tasks and objectives 4

III List of methods, algorithms and models **4**

i Overall architecture review 4

ii Creating DataBase 5

iii Setting up API . . 6

iv User Authorization and authentication 6

v K-Means Elbow Method 7

IV Results **7**

Product Recommendational System

Alexandr Mirzak*

NATIONAL RESEARCH UNIVERSITY HIGHER SCHOOL OF ECONOMICS
asmirzak@edu.hse.ru

July 27, 2022

Abstract

Each day more and more products are being manufactured and typical buyer is often struggling with a hard task of choosing the one he needs as manually browsing through all of the is not an easy option. So, recommendational systems based on different algorithms are taking place. In this work I created programing interface, which serves as a way for web services to communicate with each other, in this interface a simple, yet scalable recommendational system was developed, which can be used as a base for an online shop programming interface, once connected to its product database.

I. Introduction

i. Main result

The main result for this project is an attempt to create a web application program interface that allows a user to browse through products and run recommendation programs based on the user's searches to improve the probability of buying certain product by recommending similar ones lying in the same cluster.

ii. instruments

1. FastAPI
2. Python, Anaconda, Jupyter Notebook
3. Numpy, Pandas, Matplotlib, Sklearn, SQLalchemy, Pydantic
4. Raw unfiltered data of Walmart products listings
5. PowerPoint
6. Flourish.studio

II. About

i. Reviewing abstracts and analogues

There are a lot of approaches to recommendations for users (Collaborative filtering, Content-based filtering, Session-based recommender systems, Multicriteria recommender systems, etc.)

*Thanks to YOU for reading this!

However all of these approaches are hard to implement and require lots of resources, so the clustering approach with k-Means clustering method was selected

Algorithm of KMeans

The Kmeans algorithm is an iterative technique that attempts to split a dataset into K separate non-overlapping subgroups (clusters), each of which contains only one data point, the number of groups is completely predetermined by user, so it is crucial to choose the optimal number of groups. It attempts to make intra-cluster data points as comparable as possible while maintaining clusters as distinct (far) as possible. It distributes data points to clusters in such a way that the sum of the squared distances between them and the cluster's centroid (the mean of all data points in that cluster) is as small as possible. ($\mathbf{S}\arg\min \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \mu_i\|^2 = \mathbf{S}\arg\min \sum_{i=1}^{k} |S_i| \, Var S_i$)

Within clusters, the less variation there is, the more homogenous or similar the data points are.[3] This method of clustering is relatively good scaling to bigger datasets, the shapes and sizes of clusters may be of different sizes, allowing for more flexibility. However, this method has some disadvantages, which are needed to be aware of. The main of them are: Relatively bad clustering for outliers, as they may vary sum of squares significantly and shift the centroids, scaling with the number of dimensions is also a major problem of this method, thus the number of dimensions must be reduced.

The more precise description of the algorithms work is as follows: Given an initial set of k means $m_1^{(1)}, \ldots, m_k^{(1)}$ (see below), the algorithm proceeds by alternating between two steps:

Assignment step: Assign each observation to the cluster with the nearest mean: that with the least squared Euclidean (istance. this means partitioning the observations according to the Voronoi diagram generated by the means.)

$$S_i^{(t)} = \left\{ x_p : \left\| x_p - m_i^{(t)} \right\|^2 \leq \left\| x_p - m_j^{(t)} \right\|^2 \right\} \tag{1}$$

where each x_p is assigned to exactly one $S^{(t)}$, even if it could be assigned to two or more of them.

Update step: Recalculate means (centroids) for observations assigned to each cluster.

$$m_i^{(t+1)} = \frac{1}{\left|S_i^{(t)}\right|} \sum_{x_j \in S_i^{(t)}} x_j \qquad (2)$$

The algorithm has converged when the assignments no longer change. The algorithm is not guaranteed to find the optimum.

ii. Project tasks and objectives

The projects tasks were as follows:

- Create an API that can be used to obtain information about products
- Create a database to store information about all operations and products
- Implement user authorization for operations
- Develop a clusterization system for product

Technical requirements are:

- Authorization
  1. User authentication
  2. password hashing
  3. token validation
- Getting a product information by its ID
- Searching by a word
  1. implementing Full-Text-search
  2. text vectorization
  3. some other strange requirement
- function starting the recommendation algorithm

III. List of methods, algorithms and models

i. Overall architecture review

The application has been divided into several modules 1 for a clear and precise separation of logic as well as ease of development. API modules stand for describing and implementing path operations, dataprocess stands for preprocessing raw data from Walmart listings.

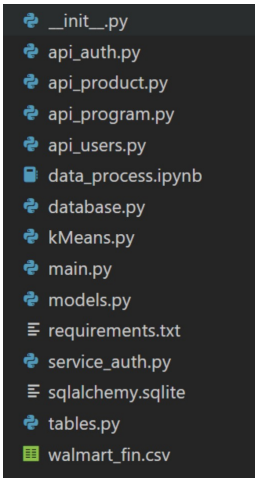

Figure 1: *Modules overview*

database includes all necessary operations for creating dataset and pushing data into it. kMeans includes the program for user to run. Models consist of all the necessary models for data to be sent and received correctly. serviceauth consist of all the necessary "service" functions for authorization and tables stand for multiple tables inside the database that reveal its structure. These modules will be further explained below.

ii. Creating DataBase

To store information about products, users, etc., you will need a table system, in other words, a database. I used SQLAlchemy plugin because of its unique approach to the database itself: it considers the database to be a relational algebra engine, not just a collection of tables. The following tables were needed to fully achieve the tasks set: products, users, experiments and recommendations.

Here is a structure of a database:

The 'Category' from products is a Foreign Key to the Categories.Title, which means that only products of existing categories can be present in the table. In the Experiments table two columns represent start and end of the work of the recommendation algorithm. After the end of the work, information about the obtained clusterization is put into the recommendations table.

Table 1: *Database stucture*

Products	Categories	Experiments
1.P$_i$*d* 1.Exp$_i$*d*	1.C$_i$*d* 1.Exp$_i$*d*	1.U$_i$*d*
2.Title	2.Title	2.Username
3.Description	3.Description	3.Email
4.Category		4.P$_h$*ash*

After creating these tables I created the function that opens session to enable data read and write inside the path operation functions and closes it when the path operation function stops:

```
$def new_session() -> Session:
 session = Session()
 try :
  yield session
 finally:
  session.close()$
```

iii. Setting up API

I used FastAPI library to create an instance of FastAPI() class, which is the app itself. Then I added the router which allowed me to add default prefix to each of the "ApiX" folders (For example, path operations on ApiProduct.py module now all start with prefix '/products'), which allows for more clean way of representing paths.

```
router = APIRouter( prefix='/products')
```

iv. User Authorization and authentication

Since this program can be used by several users, it is necessary to delimit access for more accurate and secure use. For this purpose, the possibility of authorization and authentication has been created, so the user can receive product recommendations based on their search.

A simple approach to storing passwords is to create a table in our database that maps a username with a password. When a user logs in, the server receives an authentication request with a payload containing the username and password. It is incredibly dangerous to store passwords as a plain text in a table. A much safer way is to hash the password when the user signs up, then store the hashed password and for login compare not the password, but the hashed version of it.

For this purpose, a new class AuthService was created, which allows for registration of new users as well as authentication existing ones and authorizing them. It uses JWT (JSON Web Token)[4] as a response model for signing up and signing in, which features a signature that can be validated using the "secret key" – a predetermined string.[2] This is an example of decoding 2 a token

To authorize user for each operation that requires it (running a program, getting information about a product or finding similar ones) I wrote a function GetCurrentUser, which validates the received token and gives a ModelUser as output and denies access in the case when the user

in not logged in

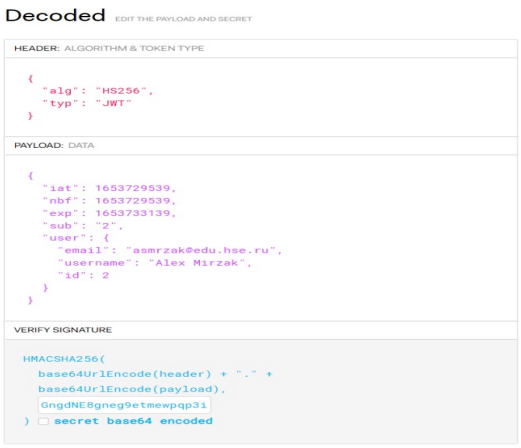

Figure 2: *decoded token*

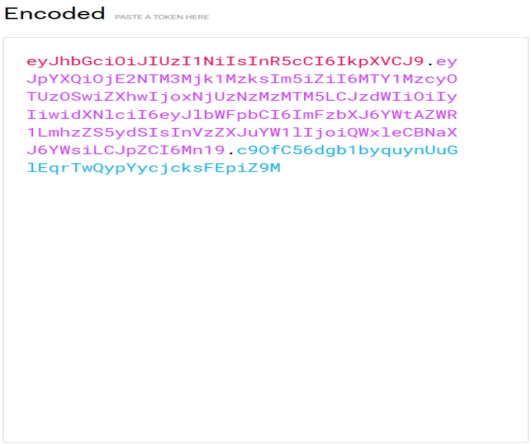

Figure 3: *encoded token*

v. K-Means Elbow Method

The k-Means algorithm is an unsupervised machine learning algorithm that divides data into k clusters. The number of clusters is set by the user, and the algorithm will attempt to group the data even if this number is not optimal in this circumstance. When the centers are plotted and the pattern seems to be an inverted triangle, shaped like an elbow, the value at "breaking point" is the most optimal, as after it no considerable reduction in sum of squared distances happen. After fitting model in a range of clusters I obtained this graph 4

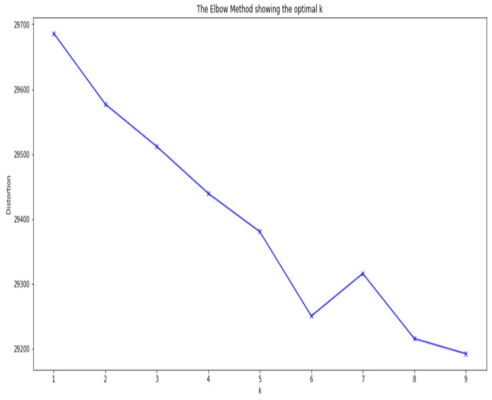

Figure 4: *Elbow method*

IV. Results

From this graph it can be seen that 6 clusters are an optimal size. From here we are able to view top-10 words from each cluster, thus those which are at the center of it and represent a certain group of products.

At the end a would like to re-

Table 2: *Clusters obtained*

CLuster 0	Cluster 1	Cluster 2	Cluster 3	Cluster 4
body	golf	sports	bike	count
oil	mens	football	bicycle	coffee
organic	balls	soccer	cycling	protein
butter	glove	basketball	road	tablets
lotion	footjoy	tennis	mtb	roast

produce the Schrödinger equation, just because I could not find any more formulas for this topic:

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = \left[-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x,t)\right]\Psi(x,t)[1] \tag{3}$$

References

[1] Feliks Aleksandrovich Berezin and Mikhail Shubin. *The Schrödinger Equation*, volume 66. Springer Science & Business Media, 2012.

[2] Rajdeep Bhanot and Rahul Hans. A review and comparative analysis of various encryption algorithms. *International Journal of Security and Its Applications*, 9(4):289–306, 2015.

[3] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[4] Michael Jones, Brain Campbell, and Chuck Mortimore. Json web token (jwt) profile for oauth 2.0 client authentication and authorization grants. Technical report, 2015.