

### PROJECT BASED LAB REPORT

**On**

**RESTAURANT RECOMMENDATION SYSTEM**

**Submitted in partial fulfilment of the Requirements for the award of the Degree of Bachelor of Technology**

**In**

**Computer science and Engineering Under the esteemed guidance of**

**Dr. S. Sagar Imambi**

**By**

|  |  |
| --- | --- |
| **ID.No** | **Student Name** |
| 170030472 | J.Lakshmi Sai Pushkala |
| 170030561 | Kushwanth Kapa |
| 170031125 | R.Naga Swetha Anogini |

**(DST-FIST Sponsored Department) K L EDUCATION FOUNDATION**

**Green Fields, Vaddeswaram, Guntur District-522 502 2019-2020**

**K L EDUCATION FOUNDATION DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**(DST-FIST Sponsored Department)**



**CERTIFICATE**

This is to certify that this project based lab report entitled **“RESTAURANT RECOMMENDATION SYSTEM”** is a bonafide work done by J.Lakshmi Sai Pushkala(170030472), Kushwanth Kapa(170030561), R.Naga Swetha Anogini(170031125) in the course **Deep Learning**, **17CS3074** in partial fulfilment of the requirements for the award of Degree in Bachelor of Technology in **COMPUTER SCIENCEAND ENGNEERING** during the Even Semester of Academic year 2019-2020.

### Faculty in Charge Head of the Department

Dr. S. Sagar Imambi Mr. V. Hari Kiran

**K L EDUCATION FOUNDATION**

**DEPT OF COMPUTER SCINCE AND ENGINEERING**

**(DST-FIST Sponsored Department)**



**DECLARATION**

We hereby declare that this project based lab report entitled **“RESTAURANT RECOMMENDATION SYSTEM”** has been prepared by us in the course **Deep Learning**, **17CS3074** in partial fulfilment of the requirement for the award of degree bachelor of technology in **COMPUTER SCIENCE AND ENGINEERING** during the Even Semester of the academic year 2019-2020.We also declare that this project-based lab report is of our own effort and it has not been submitted to any other university for the award of any degree.

### Date: 20/04/2020

### Place: Vaddeswarm

|  |  |
| --- | --- |
| **ID.NO** | **Student Name** |
| 170030472 | J.Lakshmi Sai Pushkala |
| 170030561 | Kushwanth Kapa |
| 170031125 | R.Naga Swetha Anogini |

**ACKNOWLEDGEMENT**

Our sincere thanks to ***Dr. S. Sagar Imambi*** in the Lab for her outstanding support throughout the project for the successful completion of the work.

We express our gratitude to ***Dr. S. Sagar Imambi*** the instructor and **Dr Pragnyaban Mishra** Course Co-ordinator for the course **Deep Learning**, **17CS3074** in the Computer Science and Engineering Department for providing us with adequate planning and support and means by which we can complete this project-based Lab.

We express our gratitude to **Mr. V. HARIKIRAN,** Head of the Department for computer science and Engineering for providing us with adequate facilities, ways and means by which we can complete this project-based Lab.

We would like to place on record the deep sense of gratitude to the honourable Vice Chancellor, K L University for providing the necessary facilities to carry the project-based Lab.

Last but not the least, we thank all Teaching and Non-Teaching Staff of our department and especially our classmates and our friends for their support in the completion of our project-based Lab.

|  |  |
| --- | --- |
| **ID.NO** | **Student Name** |
| 170030472 | J.Lakshmi Sai Pushkala |
| 170030561 | Kushwanth Kapa |
| 170031125 | R.Naga Swetha Anogini |

**TABLE OF CONTENTS**

**CHAPTERS PAGE NO**

ABSTRACT

CHAPTER 1: INTRODUCTION 1

* 1. INTRODUCTION 1
  2. PROBLEM DEFINITION 1
  3. SCOPE 2
  4. PURPOSE 2
  5. PROBLEM AND EXISTINGTECHNOLOGY 2
  6. PROPOSED SYSTEM 3

CHAPTER 2: REQIUREMENTS & ANALYSIS 4

* 1. PLATFORM REQUIREMENTS 4
  2. MODULE DESCRIPTION 4

CHAPTER 3: DESIGN & IMPLEMENTATION 7

* 1. ALGORITHMS 7
  2. PSEUDO CODE 7

CHAPTER 4: SCREENSHOTS 10

CHAPTER 5: CONCLUSION 17

CHAPTER 6: REFERENCES 18

**CHAPTER 1 : Introduction**

**1.1 Introduction**

A Recommendation System is an information filtering system that seeks to predict the rating a user would give for the item (in this case a restaurant). We can break down the large matrix of ratings from users and items into two smaller matrixes of user-feature and item-feature.

Recommender Systems or Recommendation Systems are simple algorithms that aim to provide the most relevant and accurate items (products, movies, events, articles, food, restaurants)to the user (customers, visitors, app users, readers) by filtering useful stuff from a huge pool of information base. Recommendation engines discover data patterns in the data set by learning consumers’ choices and produces the outcomes that co-relates to their needs and interests. Our project is to develop similar recommendation system related to restaurants.

**1.2 Problem Definition**

Restaurant Recommendation System is a recommender system for restaurants based on user preferences. When user or customer requests for a specific type of restaurant, the system automatically recommends the restaurants that meet their requirements.

Our aim is to build a restaurant recommendation system that provides personalized restaurant recommendations to users. Since different people have different food preferences and dietary restrictions, we perform careful feature selection to take advantage of the information reflected in a user’s reviews.

Furthermore, since dining is frequently a communal activity, we generated recommendations not just for an individual user, but also for a group of users. This system needs to consider the information of all individuals in a group and recommend restaurants that satisfies the group of users according to some criterion.

**1.3 Scope**

Restaurant Recommendation Systems has a wide scope in the future. At present, there are few food delivery applications which take help of the restaurant recommendation systems. There is a necessary scope in increasing the efficiency of the system.

The scope of the application is as that: The restaurants and hotels within the valley will be listed in the application.

**1.4 Purpose**

The main important purpose of the restaurant recommendation system is to help the users to visit and enjoy their favorite food in the best restaurants. For example, In the case of picking a restaurant to have lunch or dinner when you are traveling to other countries or cities, commonly you will ask your friend that lives in that country or city what is the best restaurant in town. The problem is you don’t have any friends that lived in that town. Your personal recommendations can be generated by an artificial friend: the recommender system.

This is the underlying purpose of the system to help the users in picking the best restaurants.

**1.5 Problem And Existing Technology**

While many existing recommender systems mainly target individuals, there is a remarkable increase of recommender systems which generate suggestions for groups. Some early systems were developed in a variety of domains, such as, group web page recommendation (Lieberman et al. [1999](https://link.springer.com/article/10.1007/s10844-017-0469-0#ref-CR39)), *tour packages* for groups of tourists (Ardissono et al. [2003](https://link.springer.com/article/10.1007/s10844-017-0469-0#ref-CR4)), *music tracks* and *playlists* for large groups of many listeners (Crossen et al. [2002](https://link.springer.com/article/10.1007/s10844-017-0469-0#ref-CR16)), *movies* and *TV programs* for friends and family (O’Connor et al. [2001](https://link.springer.com/article/10.1007/s10844-017-0469-0#ref-CR47); Yu et al. [2006](https://link.springer.com/article/10.1007/s10844-017-0469-0#ref-CR62)). Group scenarios are especially popular in the *food domain* in which a group of family members, friends or colleagues wants to make a party or simply have a meal together. However, the complexity significantly increases when food recommender systems need to take into account the preferences of all group members and strategies for achieving the consensus within group members.

**1.6 Proposed System**

In this system, we develop a restaurant recommendation system using the Latent Factor Collaborative Filtering Optimization. This system recommends restaurants for users or group of users based on their preferences such as beautiful ambience, good food, tasteful desserts and so.Our system provides personalized restaurant recommendations to users.

**CHAPTER 2 : REQIUREMENTS & ANALYSIS**

**2.1 Platform Requirements**

|  |  |  |
| --- | --- | --- |
| Hardware/Software | Hardware/Software element | Specification Version |
| Hardware | Processor | Intel core to duo |
| RAM | 1 GB |
| Hard Disk | 100 GB |
| Software | OS | Windows 10 |
| Python and Jupyter Notebook |

**2.2 Module Description**

Recommender Systems or Recommendation Systems are simple algorithms that aim to provide the most relevant and accurate items (products, movies, events, articles) to the user (customers, visitors, app users, readers) by filtering useful stuff from a huge pool of information base.

Recommendation engines discover data patterns in the data set by learning consumers’ choices and produces the outcomes that co-relates to their needs and interests.

Unlike offline stores, online stores have no sales people, they have huge number of products on their websites and users on other hand have limited time and patience to navigate to the items that they are looking for. Recommendation systems solve these kind of problems by exploiting the user preferences and prioritize the items based on all other users past behaviour.

**There are two types of Recommendation Systems**

1. **Content-Based Filtering**  
   Content-based filtering refers to such methods that provide recommendations by comparing representations of content describing an item to representations of content that interest the user pairs. This systems recommends based on comparison between the descriptors of the items and a user profile.
2. **Collaborative Filtering**  
   Unlike content-based filtering, this system doesn’t require description of the data hence it recommends without knowing anything about the products.

Collaborative filtering is the type of recommendation algorithm that bases its predictions and recommendations on the rating or behavior of other users in the system. The fundamental idea of collaborative filtering is to find other users in the community that share opinions.

There are two popular approaches of collaborative filtering:

1. **User-based approach**

Food Recommendation System uses the user ratings of other users with similar preferences to recommend a food item to a certain user. User-based recommendation algorithms firstly identify the k most similar users to the active user in which each user is treated as a vector and the similarities between the active user and other users are computed.

1. **Item-based approach**

Though user- based approach is useful, it suffers from the scalability problem as the user base grows. Searching from the neighbors of a user becomes time-consuming. To extend collaborative filtering to the large user base, a more scalable version of collaborative filtering, the i.e. item based approach was introduced.

The overall structure of this approach seems to be similar to that of content based approach to recommendation and personalization, but item similarity is deduced from user preference patterns rather than extracted from the item data.

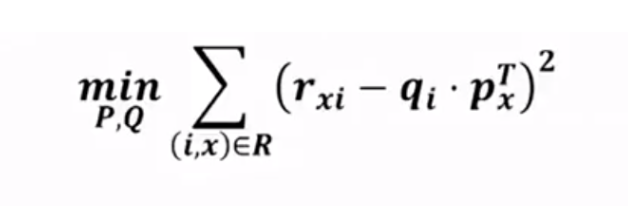
**Matrix Factorization or Latent Factor Collaborative Filtering**

Matrix factorization is a class of collaborative filtering algorithms used in recommender systems. Matrix factorization algorithms work by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices.

A Recommendation System is an information filtering system that seeks to predict the rating a user would give for the item (in this case a restaurant). We can break down the large matrix of ratings from users and items into two smaller matrixes of user-feature and item-feature.

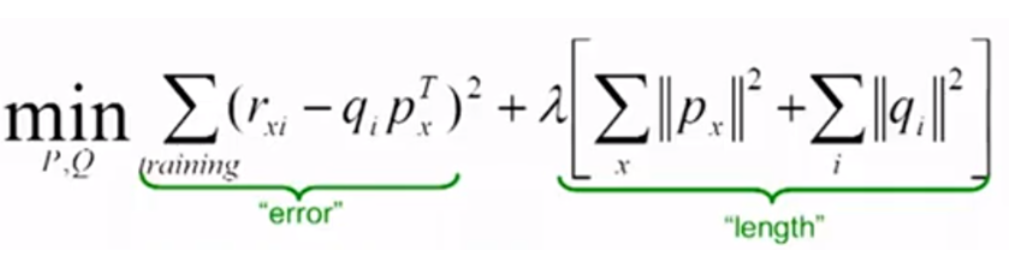
We have two matrixes (user-features, business-features) that we can multiply to predict the ratings that a user gives to a restaurant. Later, we need to update the values in the features of our two matrixes according to the Error.

To optimize the predictions we need to calculate the error using the function below.



Given P is the users-features matrix and Q is the business-features matrix. If we subtract the real ratings (r) with the predicted ratings (P.Q) and we square it, we get the LSE(Least Square Error).

To avoid overfitting we have to add regularization to our LSE formula and it will become the formula written below.



We apply the equation to minimize the error using Gradient Decent to update the values of each feature in matrix P and matrix Q.

**CHAPTER 3 : DESIGN & IMPLEMENTATION**

**3.1 Algorithms**

Matrix factorization algorithms work by decomposing the original matrix into two matrices one is the upper triangle ( U ), and the other is the lower triangle ( L ).

Algorithm:

Step 1 : Take the input matrix as U and an empty diagonal matrix as L

Step 2: n be length of matrix U

Step 3: for col in range(1,n):

1. for row in range(2,n):

a-1. if row > col

mul = U[[row,col]]/U[[col,col]]

L[row,col] = mul

U[row,] = U[row,] – mul \* U[col,]

return U

|  |
| --- |
|  |
|  |

**3.2 Pseudo Code**

def text\_process(mess):

    """

    Takes in a string of text, then performs the following:

    1. Remove all punctuation

    2. Remove all stopwords

    3. Returns a list of the cleaned text

    """

    # Check characters to see if they are in punctuation

    nopunc = [char for char in mess if char not in string.punctuation]

    # Join the characters again to form the string.

    nopunc = ''.join(nopunc)

    # Now just remove any stopwords

    return " ".join([word for word in nopunc.split() if word.lower() not in stop])

yelp\_data['text'] = yelp\_data['text'].apply(text\_process)

#userid vectorizer

userid\_vectorizer = TfidfVectorizer(tokenizer = WordPunctTokenizer().tokenize, max\_features=5000)

userid\_vectors = userid\_vectorizer.fit\_transform(userid\_df['text'])

userid\_vectors.shape

#Business id vectorizer

businessid\_vectorizer = TfidfVectorizer(tokenizer = WordPunctTokenizer().tokenize, max\_features=5000)

businessid\_vectors = businessid\_vectorizer.fit\_transform(business\_df['text'])

businessid\_vectors.shape

#Matrix Factorization

def matrix\_factorization(R, P, Q, steps=25, gamma=0.001,lamda=0.02):

    for step in range(steps):

        for i in R.index:

            for j in R.columns:

                if R.loc[i,j]>0:

                    eij=R.loc[i,j]-np.dot(P.loc[i],Q.loc[j])

                    P.loc[i]=P.loc[i]+gamma\*(eij\*Q.loc[j]-lamda\*P.loc[i])

                    Q.loc[j]=Q.loc[j]+gamma\*(eij\*P.loc[i]-lamda\*Q.loc[j])

        e=0

        for i in R.index:

            for j in R.columns:

                if R.loc[i,j]>0:

                    e= e + pow(R.loc[i,j]-np.dot(P.loc[i],Q.loc[j]),2)+lamda\*(pow(np.linalg.norm(P.loc[i]),2)+pow(np.linalg.norm(Q.loc[j]),2))

        if e<0.001:

            break

    return P,Q

#static input

words = "i want to have dinner with excellent dessert"

test\_df= pd.DataFrame([words], columns=['text'])

test\_df['text'] = test\_df['text'].apply(text\_process)

test\_vectors = userid\_vectorizer.transform(test\_df['text'])

test\_v\_df = pd.DataFrame(test\_vectors.toarray(), index=test\_df.index, columns=userid\_vectorizer.get\_feature\_names())

predictItemRating=pd.DataFrame(np.dot(test\_v\_df.loc[0],Q.T),index=Q.index,columns=['Rating'])

topRecommendations=pd.DataFrame.sort\_values(predictItemRating,['Rating'],ascending=[0])[:7]

for i in topRecommendations.index:

    print(df\_business[df\_business['business\_id']==i]['name'].iloc[0])

    print(df\_business[df\_business['business\_id']==i]['categories'].iloc[0])

    print(str(df\_business[df\_business['business\_id']==i]['stars'].iloc[0])+ ' '+str(df\_business[df\_business['business\_id']==i]['review\_count'].iloc[0]))

    print('')

#dynamic input

words = input()

test\_df= pd.DataFrame([words], columns=['text'])

test\_df['text'] = test\_df['text'].apply(text\_process)

test\_vectors = userid\_vectorizer.transform(test\_df['text'])

test\_v\_df = pd.DataFrame(test\_vectors.toarray(), index=test\_df.index, columns=userid\_vectorizer.get\_feature\_names())

predictItemRating=pd.DataFrame(np.dot(test\_v\_df.loc[0],Q.T),index=Q.index,columns=['Rating'])

topRecommendations=pd.DataFrame.sort\_values(predictItemRating,['Rating'],ascending=[0])[:7]

for i in topRecommendations.index:

    print(df\_business[df\_business['business\_id']==i]['name'].iloc[0])

    print(df\_business[df\_business['business\_id']==i]['categories'].iloc[0])

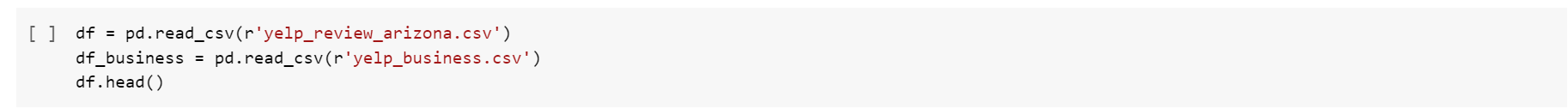
    print(str(df\_business[df\_business['business\_id']==i]['stars'].iloc[0])+ ' '+str(df\_business[df\_business['business\_id']==i]['review\_count'].iloc[0]))

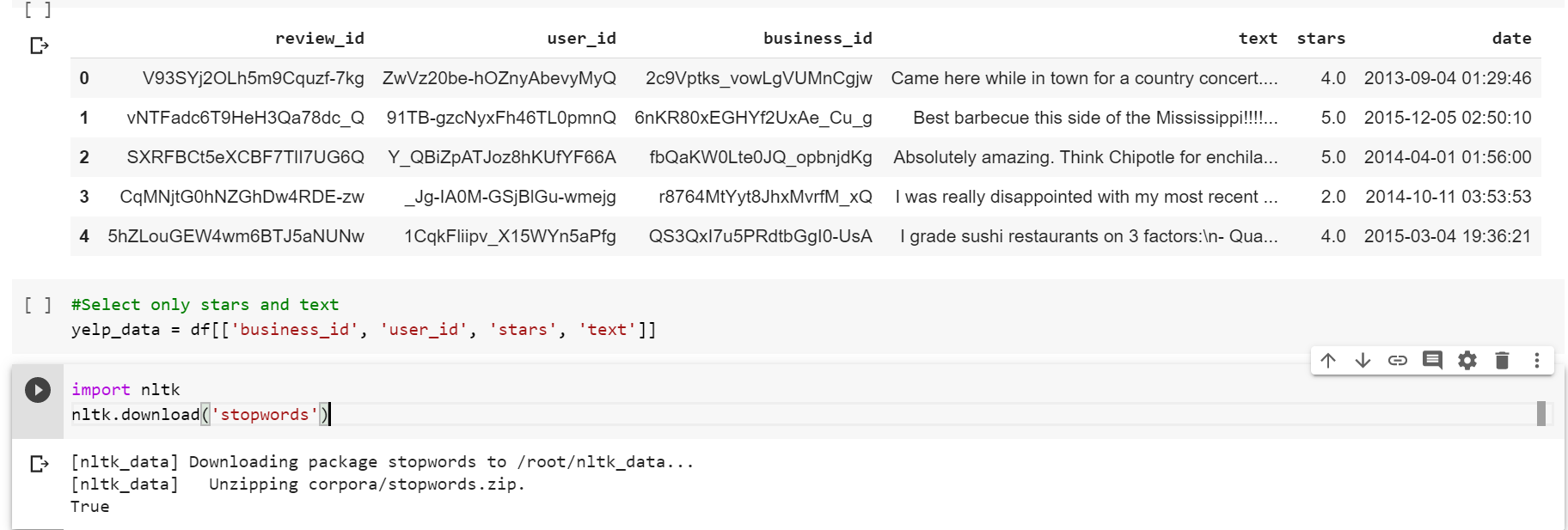
    print('')

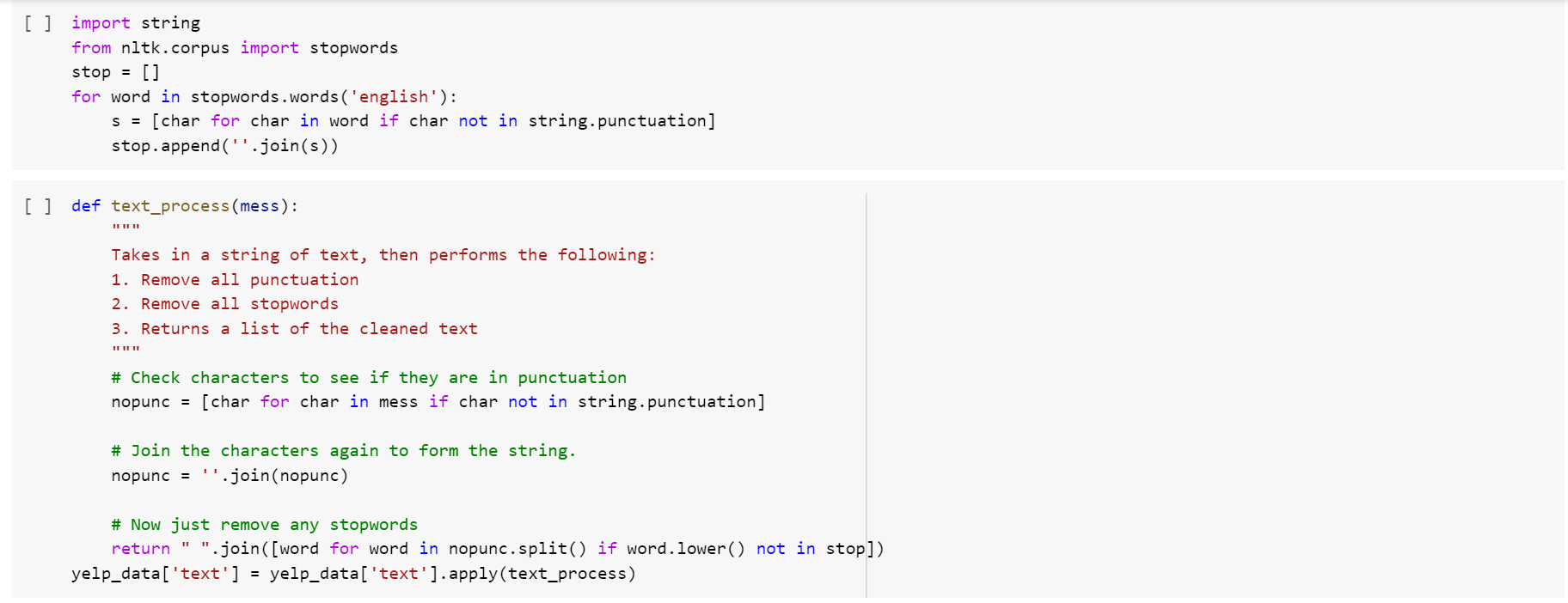
**CHAPTER 4 : SCREENSHOTS**

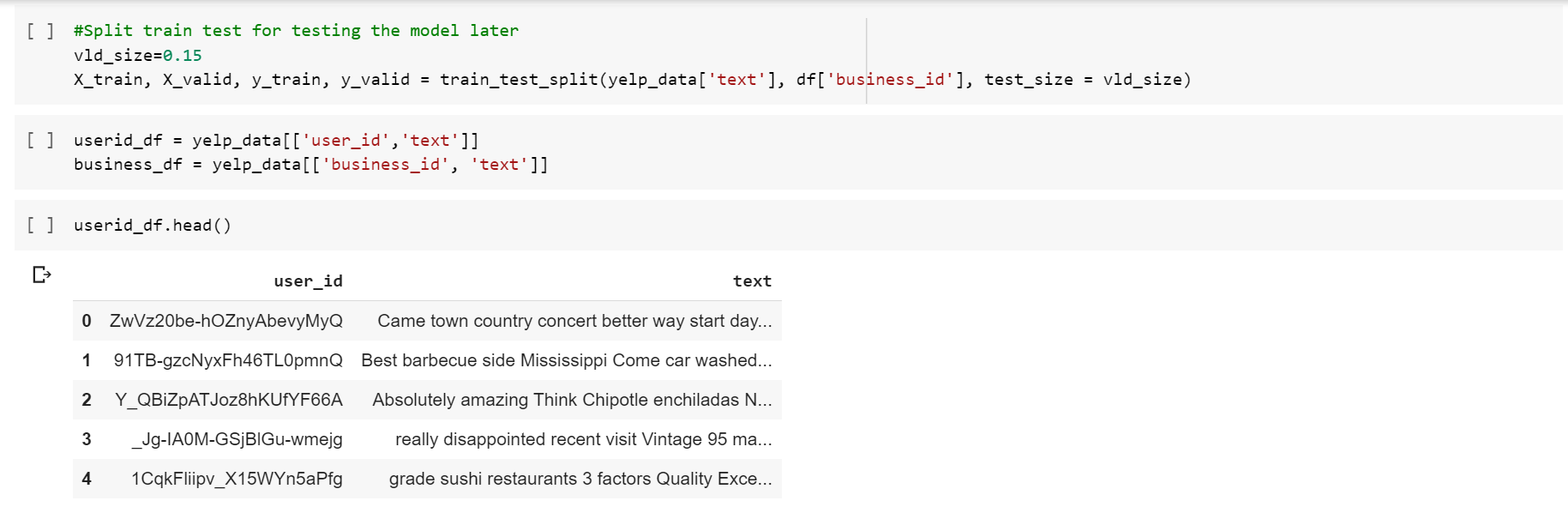


**Import Data**

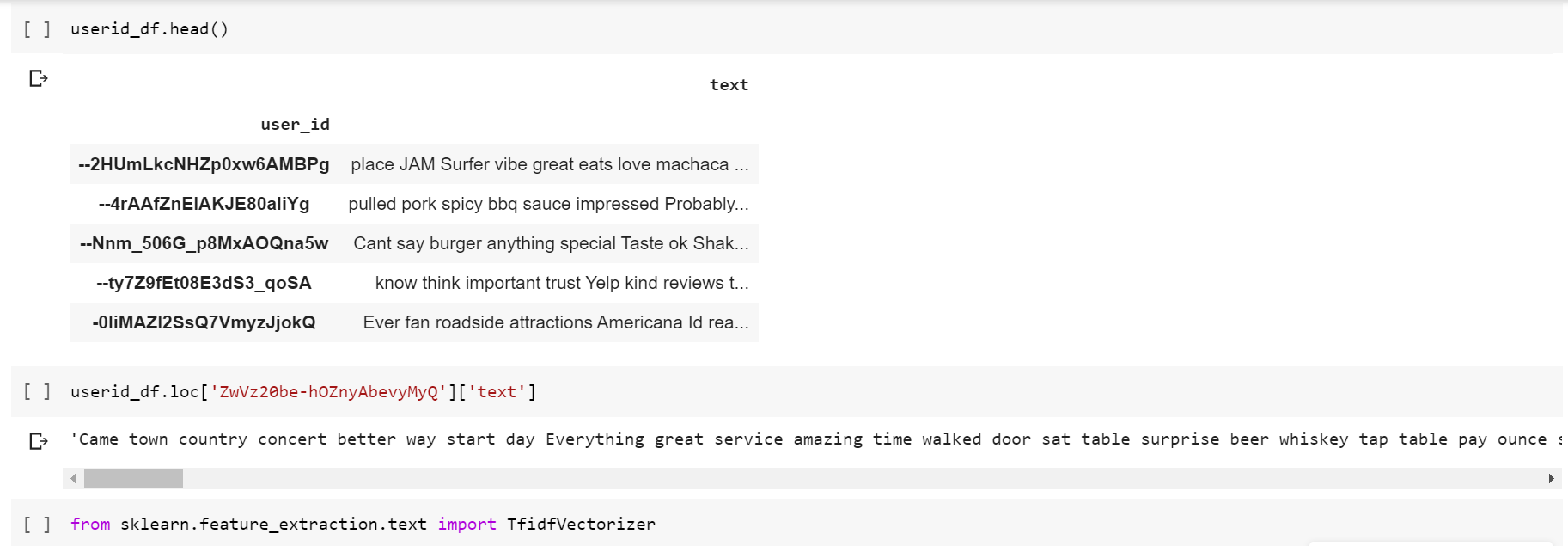






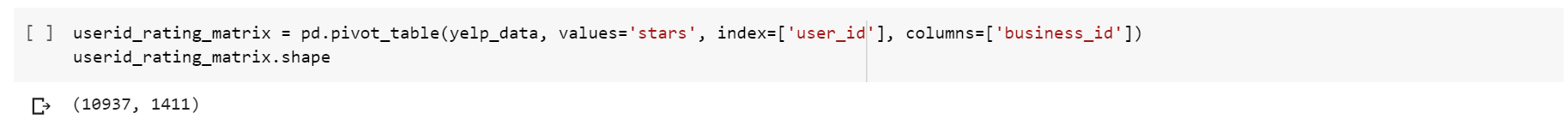


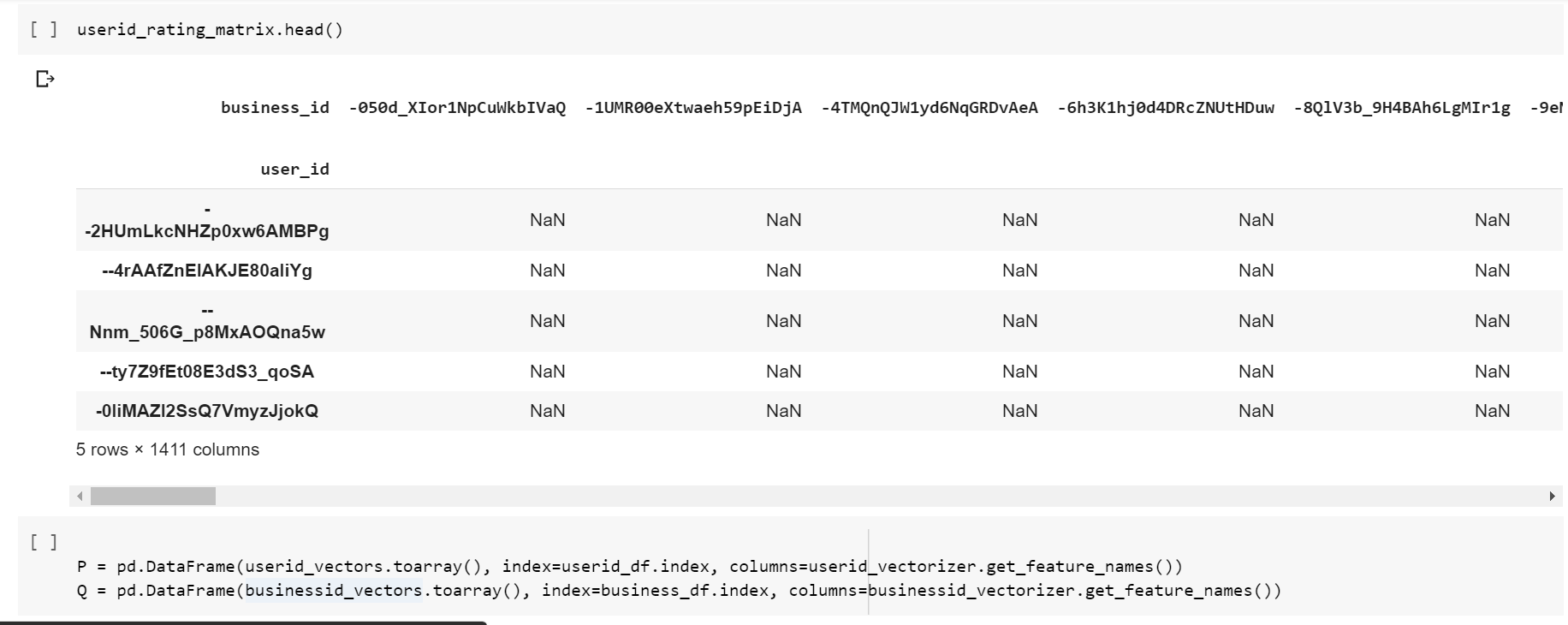


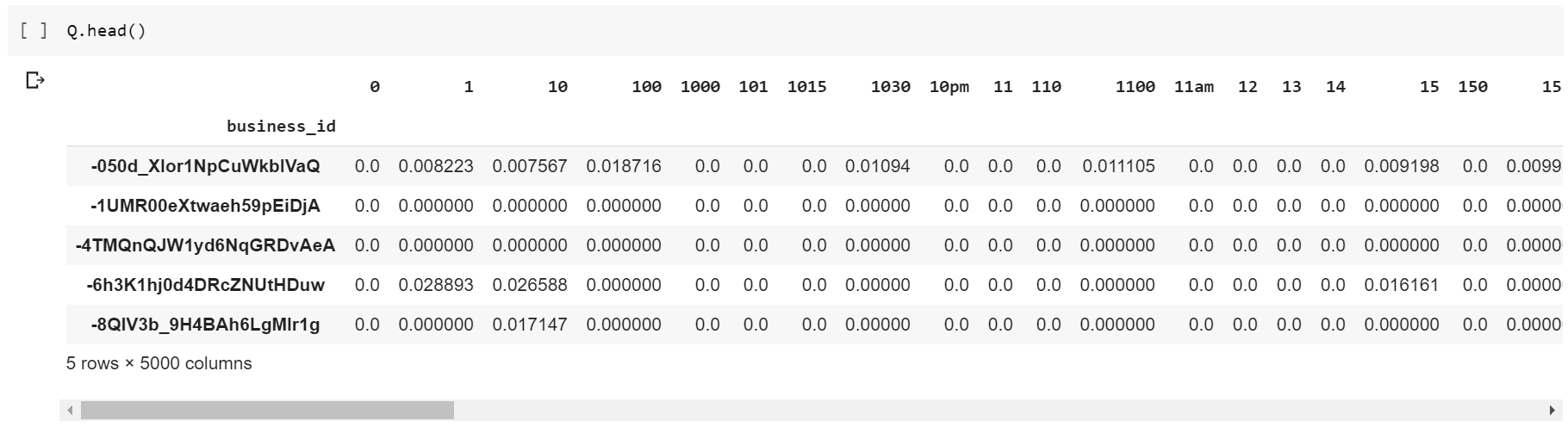




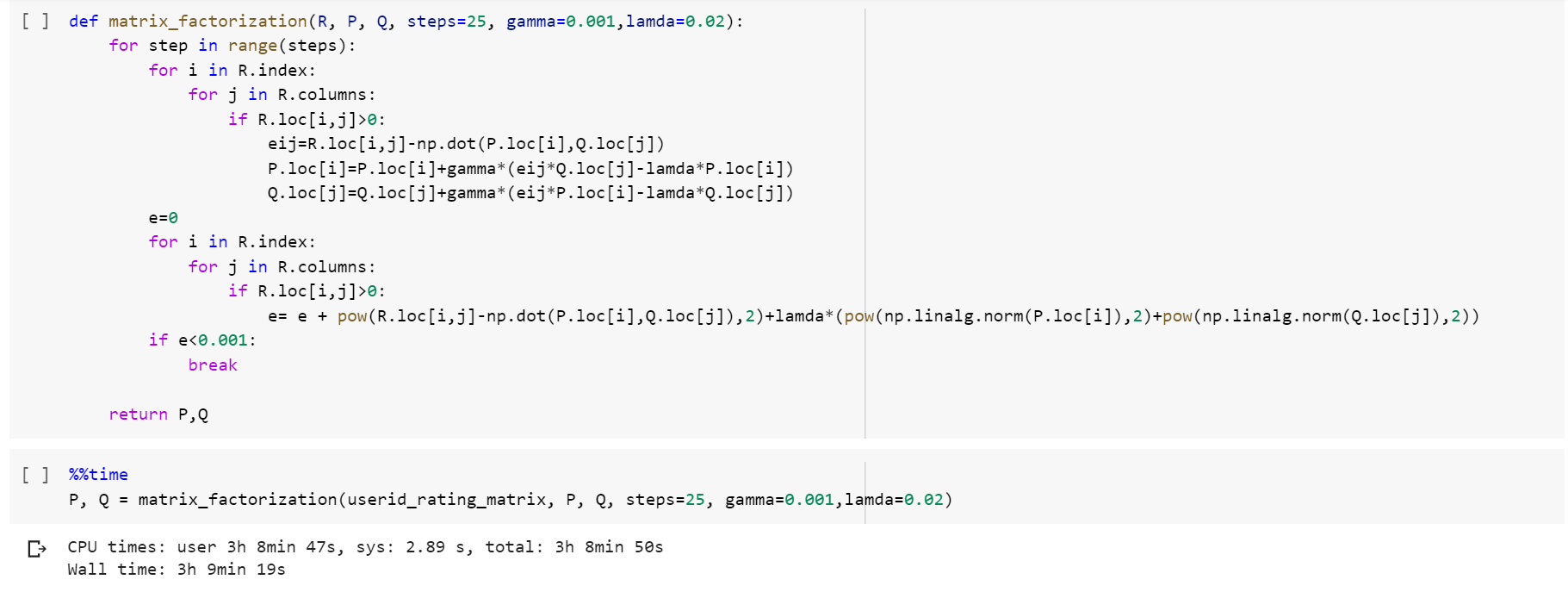
**Matrix Factorization**

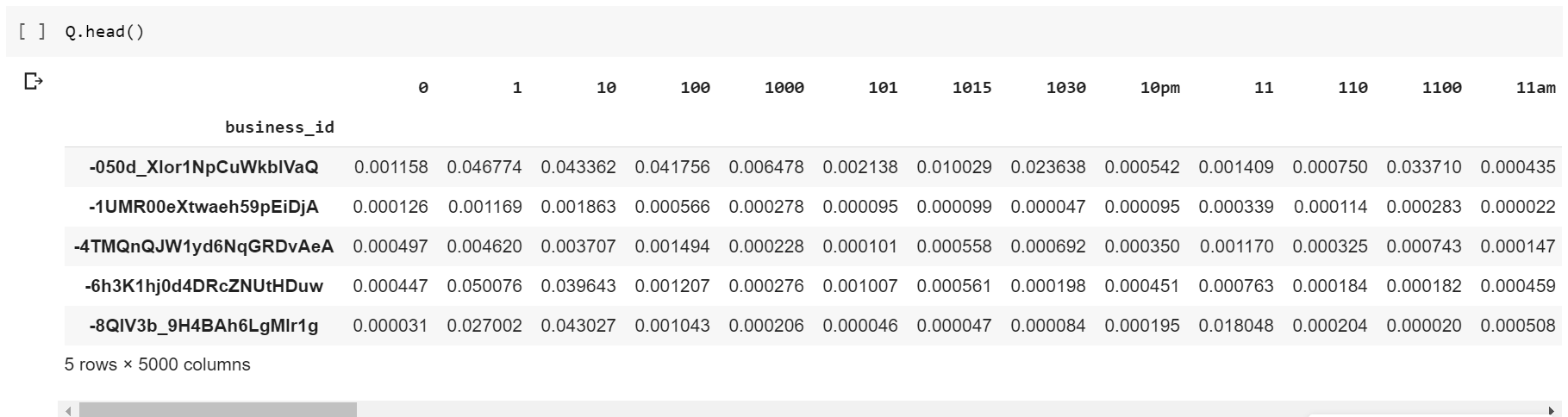






## **Gradient Decent Optimization**

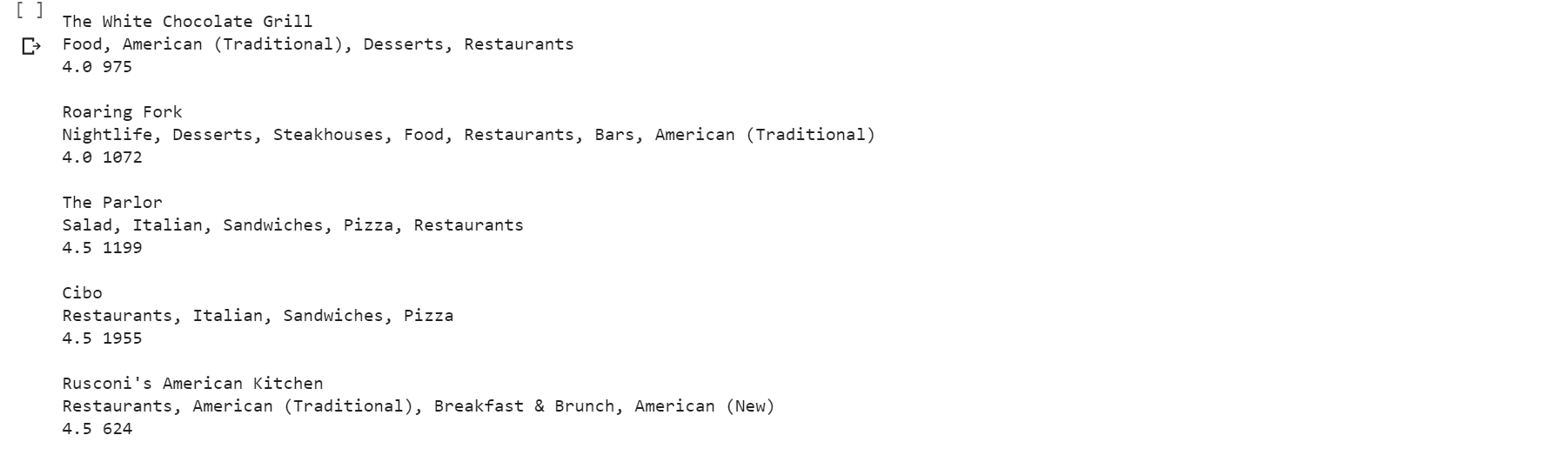






## **Prediction for input text**









**CHAPTER 5 : CONCLUSION**

The project Restaurant Recommendation System was successfully completed by using latent factor collaborative filtering or matrix factorization.

In this project, we developed a model that could recommend the restaurant based on the choice of your interest.

**CHAPTER 6 : REFERENCES**

* <https://towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0>
* <https://medium.com/@sumith.gannarapu/restaurant-recommendation-system-b52911d1ed0b>
* <https://towardsdatascience.com/recommender-systems-in-practice-cef9033bb23a>