

Anime Recommender System

A PROJECT REPORT

Submitted by

Ashwin Prasad H

191401003

in the partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND BUSINESS SYSTEMS



**RAJALAKSHMI
ENGINEERING COLLEGE**
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

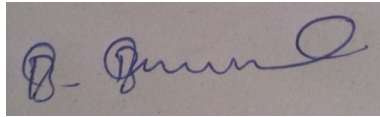
RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI-602105

ANNA UNIVERSITY: CHENNAI-600025

SEPTEMBER 2020

BONAFIDE CERTIFICATE

Certified that this project report “**Anime Recommender System**” is the bonafide work of “**Ashwin Prasad H (191401003)**” who carried out the project work under my supervision.



SIGNATURE

SIGNATURE

Dr.k.Devaki

HEAD OF DEPARTMENT

PROFESSOR

Department of Computer Science and
Business Systems

Rajalakshmi Engineering College
Chennai – 602 105

Mr.B.BHUVANESWARAN, M.E.,

SUPERVISOR

ASSISTANT PROFESSOR (SG)

Department of Computer Science and
Engineering

Rajalakshmi Engineering College
Chennai - 602 105

Submitted to project and viva-voce Examination held on _____

INTERNALEXAMINER

EXTERNALEXAMINER

ACKNOWLEDGEMENT

Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E., F.I.E.**, and our respected Chairperson **Dr. (Mrs) THANGAM MEGANATHAN, Ph.D.**, for providing us with the requisite infrastructure and sincere endeavouring educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.**, our beloved Principal for his kind support and facilities provided to complete our work in time.

We express our sincere thanks to **Dr. N. SANKARRAM, M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Mr. BHUVANESWARAN, M.E.**, Assistant Professor (SG), Department of Computer Science and Engineering, Rajalakshmi Engineering College for her valuable guidance throughout the course of the project.

We are glad to thank our Faculty coordinator, **MRS.. Helen Vijitha** , Associate Professor, Department of Humanity and Sciences for her useful tips during our review to build our project.

Finally, we express our gratitude to our parents and classmates for their moral support and valuable suggestions during the course of the project.

ABSTRACT

In Today's Digital World, It is important for users to have a smooth, easy and personalized experience which makes the searching for the Items that the user prefer easily accessible. If the customer's needs are satisfied and his personal experiences are used to recommend products to customers, it increases the sales of a company and helps reach a broader audience. Each and every user has a different taste and needs. So , it won't be efficient to display the same list of items and recommend the same products to all the users. This is where recommender systems come into play. Recommender systems take into account a user's previous browsing experience and item preference to recommend new products to the user. This way, the probability of the user buying the product is increased significantly which improves both the user's experience and the company's sales. We are already using these recommender systems today in a lot of different fields. For example : Amazon recommends products for customers to buy. Netflix recommends movies for users to watch. Library applications recommend books based on reading history, etc.

Anime is a style of japanese animated film or series that is typically aimed towards both adults and children.

Building an anime recommender system would improve the popularity and reach of the anime culture and helps several anime artists and manga artists.

TABLE OF CONTENTS

CHAPTER NO.	T I T L E	PAGE NO.
	ABSTRACT	
	LIST OF FIGURES	
	LIST OF ABBREVIATION	
1	INTRODUCTION	1-5
	1.1 General	1
	1.2 Objectives	3
	1.3 Existing System	3
	1.3.1 Limitations in Existing System	4 4
	1.4 Proposed System	5
	1.4.1 Advantages Of Proposed System	
2	LITERATURE SURVEY	6-10
3	SYSTEM SPECIFICATIONS	11-13
	3.1 Software Specification	11
	3.1.1 Jupyter Notebook	11
4	SYSTEM DESIGN	14-18
	4.1.2 Development Environment	15

5	PROJECT DESCRIPTION	19-23
	5.1 Description	19
	5.2 Modules	20
	5.2.1 Module Description	20
6	IMPLEMENTATION AND RESULT DISCUSSION	24-34
	6.1 Dataset Collection	24
	6.2 Exploratory Data Analysis	24
	6.3 Data pre-processing	25
	6.4 Model Training	26
	6.5 Model Testing	28
7	CONCLUSION & FUTURE ENHANCEMENT	35
	7.1 Future Enhancement	35
	7.2 conclusion	35
8	REFERENCES	37

LIST OF ABBREVIATIONS

NN	Neural Networks
CF	Collaborative Filtering
EDA	Exploratory Data Analysis
TF	Tensorflow

CHAPTER 1

INTRODUCTION

1.1 GENERAL

In the past few years, the anime culture has been increasing all over the world and piqued people's interest and curiosity. Some Anime films were also became so famous that their box office collections in theaters helped the whole japanese film industries during tough times. But still , the reach of a lot of anime does not extend a lot and high quality anime aren't getting recognized. This is because the number of people watching anime are less compared to live-action movies. But, there are animes which even regular movie watchers watch because of their popularity. There are not a lot of anime recommender systems that personalize for each user based on the previous anime's watches. As a result of this and a small community , the watchers don't watch other not so popular but , deserving anime. To solve all these problems , anime recommender system could be of great help for the both the user and the anime production companies to keep producing quality anime.

Developments

In recent times, Anime has been an huge part of improvement in Japan's economy. A Newly released anime movie called "Kimetsu no yaiba : mugen train" has dominated the box office and reached 100 million \$ sales in a very short period of time, which helped the entire cinema industry of japan amid this covid situation. Things like these led to the acknowledgement of this separate genre and helps improve the overall anime community.

1.2 OBJECTIVES

The main objective is to take into account both the details about the anime and the user preference to produce an effective recommender system that guesses whether a particular user likes a particular anime or not.

1.3 EXISTING SYSTEM

In most of the sites, there is no personalized recommender system for anime. They only suggest anime to watch based on the overall rating and popularity of the anime and recent releases.

1.3.1 LIMITATION IN EXISTING SYSTEM

- The Key limitation of such a system is that the anime with highest rating could have very less number of people who have rated it.

1.4 PROPOSED SYSTEM

Instead of using the overall ratings of the anime for recommendation, the proposed system is to use a vector representation assigned to each user and each anime and enable learning of the values in these vectors using a optimization technique called gradient descent.

This can be done using a neural network with an embedding matrix layer which potentially acts as the matrix where each column acts as the vector for user and anime.

1.4.1 ADVANTAGES OF PROPOSED SYSTEM:

- Unique and personalized recommendations for each user.
- Each anime has it's own unique features learned over time, instead of overall rating.

CHAPTER 2

LITERATURE SURVEY

1. Item2Vec: Neural Item Embedding for Collaborative Filtering Oren Barkan and Noam Koenigstein Microsoft Tel Aviv University

Many Collaborative Filtering (CF) algorithms are item-based in the sense that they analyze item-item relations in order to produce item similarities. Recently, several works in the field of Natural Language Processing suggested to learn a latent representation of words using neural embedding algorithms. Among them, the Skipgram with Negative Sampling (SGNS), also known as Word2Vec, was shown to provide state-of-the-art results on various linguistics tasks. In this paper, we show that item-based CF can be cast in the same framework of neural word embedding. Inspired by SGNS, we describe a method we name Item2Vec for item-based CF that produces embedding for items in a latent space. The method is capable of inferring item-to-item relations even when user information is not available. We present experimental results on large scale datasets that demonstrate the effectiveness of the Item2Vec method and show it is competitive with SVD.

2. Deep Convolutionary Network: Embedding User and Item Features for Recommendation

Making proper recommendation of items to users at the right time is a fundamental task in e-commerce platforms and social service websites. The compatibility between user's interest and item's property is a good predictive factor on whether the user will interact with the item in future. Conversely, the interactions between users and items further drives the evolution of user interests

and item features. As users interact with different items, users' interests and items' features can also co-evolve over time, i.e., their features are intertwined and can influence each other:

- From User to item. In discussion forums such as Reddit, although a group (item) is initially created for statistics topics, users with very different interest profiles can join this group. Hence, the participants can shape the features of the group through their postings. It is likely that this group can finally become one about deep learning if most users discuss about deep learning.
- From Item to user. As the group is evolving towards topics on deep learning, some users may become more interested in such topics, and they may participate in other specialized groups. On the contrary, some users may gradually gain interests in math groups, lose interests in statistics group. Such coevolutionary nature of user and item features raises very interesting and challenging questions: How to model coevolutionary features? How to efficiently train such models on large scale data? There are previous attempts which divide time into epochs, and perform tensor factorization to learn the latent features [7, 25, 38]. These methods are not able to capture the fine-grained temporal dynamics of user-item interactions, and can not answer the query related to time of interaction. Recent, point processes which treat event times as random variables have emerged as a good framework for modeling such temporal feature evolution process [13, 35]. However, these previous work make strong parametric assumptions about the functional form of the generative processes, which may not reflect the reality, and is not accurate enough to capture the complex and nonlinear co-evolution of user and item features in real world.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 SOFTWARE SPECIFICATION

The purpose of the Software Requirement Specification is to produce the specification of the analysis task and also to establish complete information about the requirement, behavior and also the other constraint like functional performance and so on. The main aim of the Software Requirement Specification is to completely specify the technical requirements for the software product in a concise and in unambiguous manner.

3.1.1 Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

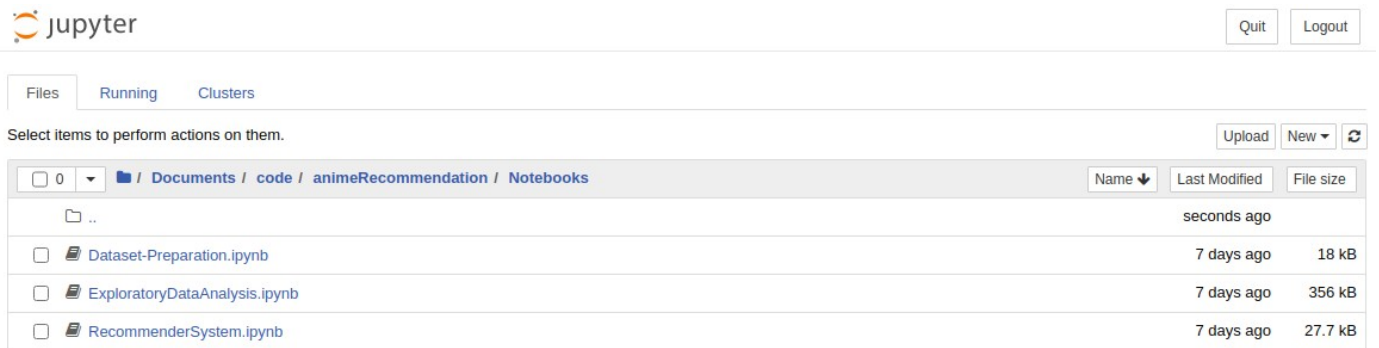


Fig: 3.1.1 Jupyter Notebook

Table : Jupyter Notebook System Requirement Table

RAM	1 GB of free RAM	8 GB of total system RAM
Disk space	1 GB and another 1 GB for caches	SSD drive with at least 5 GB of free space
Monitor resolution	1024x768	1920×1080

CHAPTER 4

SYSTEM DESIGN

4.1 GENERAL

The Development of Anime recommender system consist of a number of steps and the most vital and the first step is to collect the dataset of user ratings for all the anime and pre-processing, cleaning the data. The second process is to train the neural network collaborative filtering model on the dataset to generalize well and increase the accuracy so that inorder to predict whether a user particular user likes a particular anime.

4.1.2 DEVELOPMENT ENVIRONMENT

Hardware Environment

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the systems do and not how it should be implemented.

- Hard disk : 1000GB
- Monitor : DELL
- Ram : 8GB
- Processor : Intel® Core™ i5-4310M CPU @ 2.70GHz × 4
- Processor speed : 2.8GHz

Software Environment

The software requirements are the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team's and tracking the team's progress throughout the development activity.

- Operating system : Ubuntu18.04
- Languages : Python
- IDE : Jupyter Notebook

CHAPTER 5

PROJECT DESCRIPTION

5.1 GENERAL

Anime recommender system would greatly improve the sales of anime as well as manga and attract more customers as it would do a great job at recommending animes that the user would probably like. Neural Network Collaborative Filtering with Learned Embedding. In this process, we create 1 matrix each for anime and user, where each column belongs to a particular anime or user. The values of these matrices are randomly initialized and their values are learned over time with Gradient Descent.

Once these values are learned , we can make the predictions for animes that a particular user likes. For identifying the vector for different users , we use the user id and the process is similar for anime. For a particular user, both these vectors are flattened into a single layer and goes through a single sigmoid activation function , which tells us if the user likes the anime or not (probability).

This can be implemented in a any anime website which can take into user account and compare all the anime vectors with the user's vector. The dot product of this vectors are passed to a few fully connected layers with a sigmoid activation at the end, which returns the probability of the user liking the anime.

All the learnable parameters are learnt through gradient descent repeated in epochs.

5.2 MODULES

1. Dataset Collection Module
2. Exploratory Data Analysis
3. Data Pre-Processing
4. Model Training
5. Model Working

5.2.1 MODULE DESCRIPTION

1. Dataset Collection Module

The Dataset for this project can be collected from the most famous dataset hub for DeepLearning (Kaggle) . This is a dataset of 76000 user recommendations and ratings that was downloaded from kaggle in form of 2 csv files. One containing the list of all anime and it's details and the other one containing the user ratings to anime.

2. Exploratory Data Analysis

Exploratory Data Analysis is the Vital process of analyzing the initial data, summarizing the main characteristics and setting up the hypothesis often by means of visualization.

In this process, we explore some of the properties of the anime available in the dataset such as genre and listing out the top anime based on the calculated weighted averages

3. Data – preprocessing

Data-preprocessing is a process of checking for null values in the dataset, converting implicit ratings into an explicit one and removing the unwanted columns for training the Deep Learning Model.

4. Model Training

Once , the data pre-processing part was completed, the model architecture was prepared and training of the model on this dataset takes place. The training doesn't need to have large number of epochs because of the model's architecture containing

2 embedding layers

5. Model working:

Once the model was trained, it was given a user id and a anime id. The model searches for those vectors and goes through forward propagation. We get a result between 0 and 1. once, the result was rounded off, we know if it is acceptable by the user or not.

CHAPTER 6

IMPLEMENTATION AND RESULT DISCUSSION

6.1 DATASET COLLECTION

The Dataset was collected from the dataset-hub for deep learning projects (Kaggle). This consist of data of all the anime and user ratings to a particular anime.

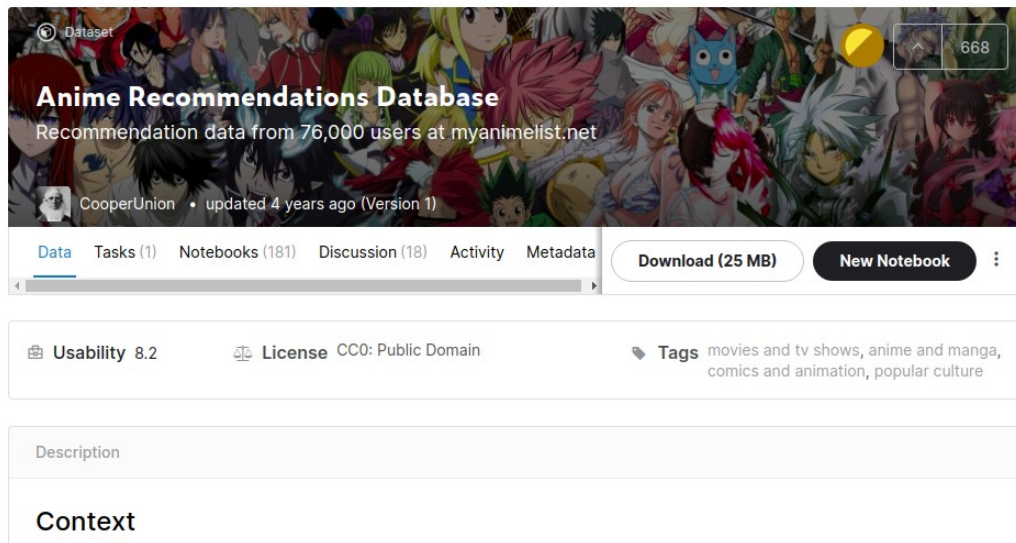


Fig: 6.1. Dataset Collected

6.2 Exploratory Data Analysis

Exploring through the data to gather insights on it and set up initial hypothesis using data visualization techniques and more

```
In [7]: fig1, ax1 = plt.subplots()
ax1.pie(genreValue, labels=keyGenres, autopct='%1.1f%%', shadow=True, startangle=90, radius=6)
plt.show()
```

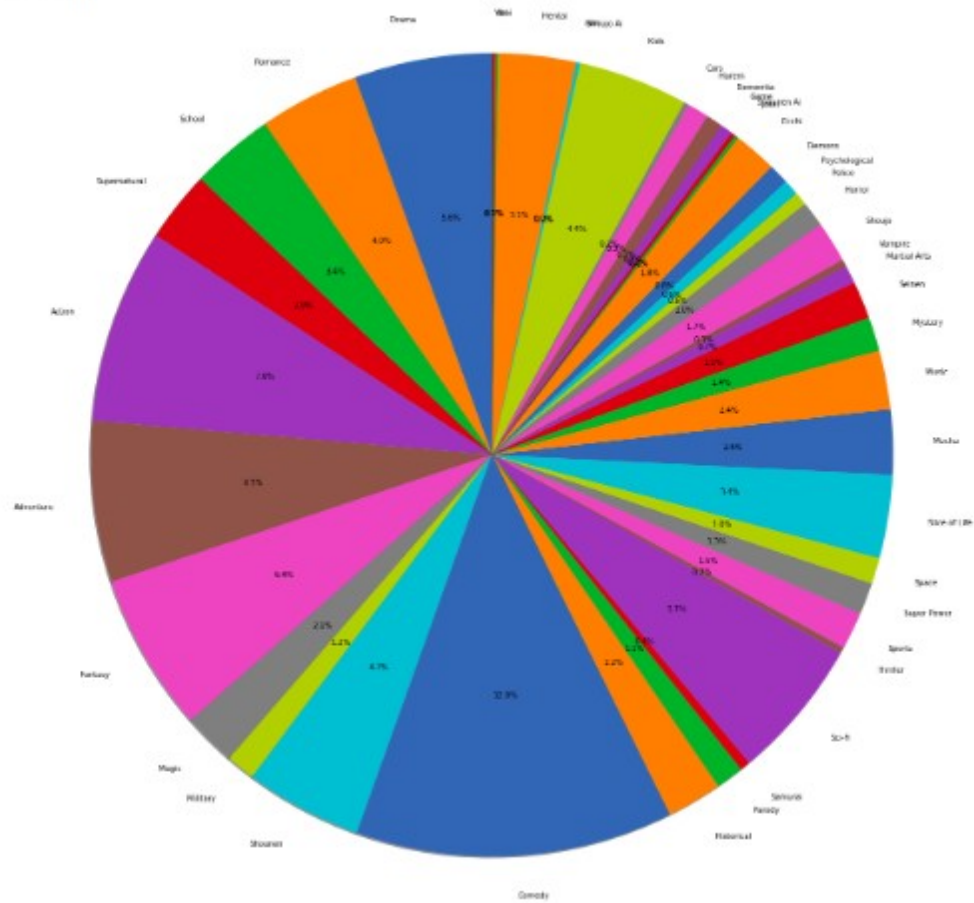


Fig: 6.2. Dataset Seperated

6.3 DATA PRE-PROCESSING

In this process, unnecessary columns are removed and null values are checked and the processed dataset is exported to a csv file for creating a model .

In the below image, recommender dataset is the prepared dataset

Fig: 6.3.1 Prepared dataset

Name ↓ Last Modified File size			
0	Documents / code / animeRecommendation / Datasets		
..	seconds ago		
anime.csv	a year ago	936 kB	
rating.csv	a year ago	111 MB	
recommenderDataset.csv	8 days ago	296 MB	

6.4 Model Training

The model is created based on specific architecture with embedding layers and is trained on the new processed dataset and accuracy and loss of the model is visualized over time.

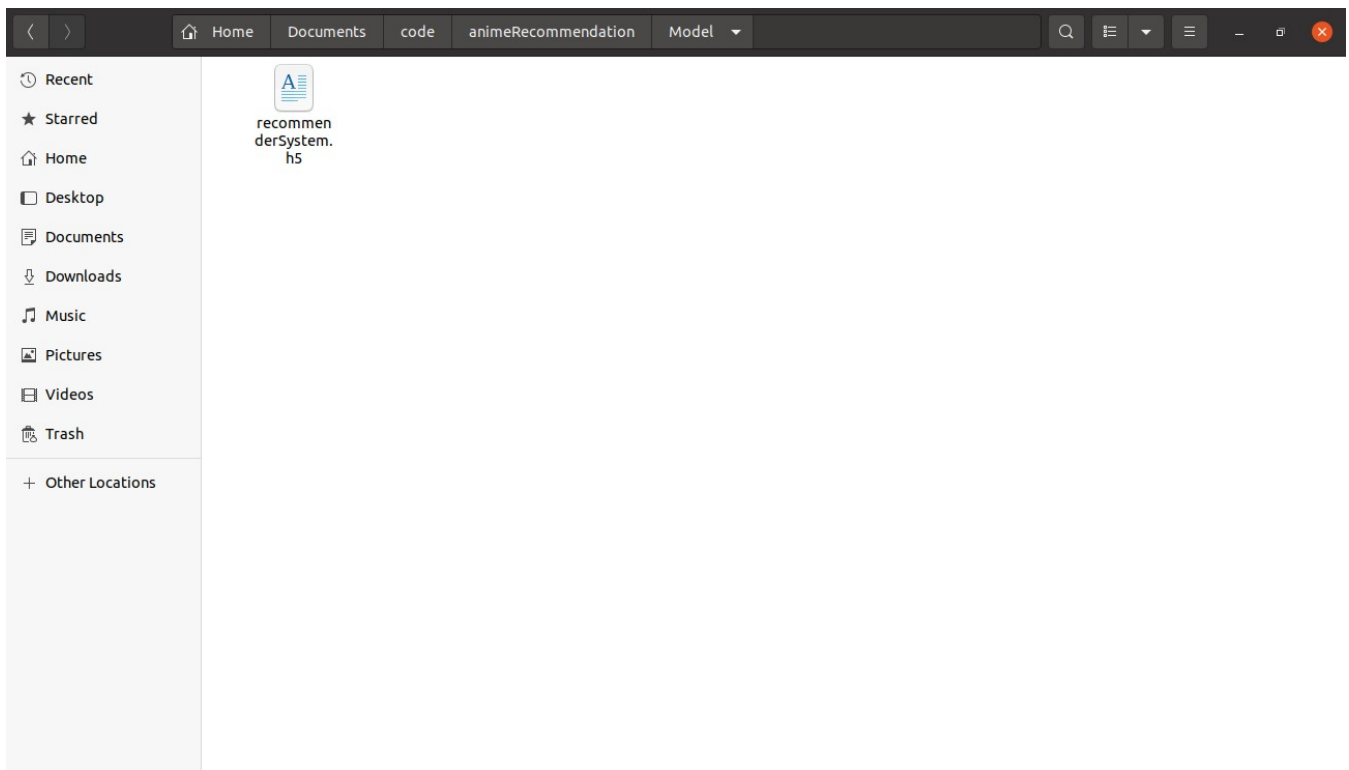


Fig : 6.4.1 The trained model is saved

6.5 Model Working

The trained model is loaded back to the notebook. Inputs of user id and anime id are given to this model to produce a probability of user liking the anime.

```
: # New Predictions
user_id = int(input("Enter the User Id: "))
anime_id = int(input("Enter the Anime Id: "))
y_pred = model.predict([np.array([user_id]),np.array([anime_id])])
if y_pred.round():
    print(f"The User Will probably Like {df[df['anime_id'] == anime_id]['name'].iloc[0]}")
else:
    print(f"The user Will probably Not Like {df[df['anime_id'] == anime_id]['name'].iloc[0]}")
```

```
Enter the User Id: 64178
Enter the Anime Id: 20
The User Will probably Like Naruto
```

6.6 SOURCE CODE

Dataset-preparation.ipynb

```
# coding: utf-8

# # Anime Recommender System
#
# 
#
# <i>Naruto Episode 107</i>
#
# <p><b>Anime : </b> a style of Japanese film and television animation, typically aimed
at adults as well as children. </p>

# ## Dataset Loading

# In[1]:

#importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# In[2]:

#importing the dataset
df1 = pd.read_csv("./anime.csv")
df2 = pd.read_csv("./rating.csv")

# In[3]:

df1.head()

# In[4]:

df2.head()
```

```
# In[5]:
```

```
df = df2.merge(df1,on='anime_id')  
df.head()
```

```
# ## Dataset Preparation
```

```
# In[6]:
```

```
#dropping unnecessary variables  
df.drop(['rating_y','members','genre','episodes','type'],inplace=True,axis=1)
```

```
# In[7]:
```

```
df.head()
```

```
# In[8]:
```

```
#converting to binary values (user likes or doesn't like)  
df['rating_x'] = df['rating_x'].replace([0,1,2,3,4,5],0)  
df['rating_x'] = df['rating_x'].replace([6,7,8,9,10],1)
```

```
# In[9]:
```

```
# removing rows where movie was not rated  
df['rating_x'] = df['rating_x'].replace(-1,np.nan)  
df.dropna(inplace=True)
```

```
# In[10]:
```

```
df.head()
```

```
# In[11]:
```

```
#exporting the dataset
df.to_csv("recommenderDataset.csv")
```

ExploratoryDataAnalysis.ipynb

```
#!/usr/bin/env python
# coding: utf-8
```

```
# # Exploratory Data Analysis of Anime Dataset
```

```
#
```

```
# Exploratory Data Analysis is the Vital process of analyzing the initial data,
summarizing the main characteristics and setting up the hypothesis often by means of
visualization,
```

```
#
```

```
# 
```

```
# In[1]:
```

```
#importing the libraries
```

```
import numpy as np
```

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

```
# In[2]:
```

```
#importing the dataset
```

```
df1 = pd.read_csv("./rating.csv")
```

```
df2 = pd.read_csv("./anime.csv")
```

```
# In[3]:
```

```
df1.head()
```

```
# In[4]:
```

```
df2.head()
```

```

# ## Genre Analysis

# In[5]:

originalGenre = []
indexCount = []

#function to collect each genre and number of genre per anime
def genreSplit(genre):
    if "," in str(genre):
        genrelist = genre.split(", ")
        indexCount.append(0)
        for i in genrelist:
            originalGenre.append(i)
            indexCount[-1]+=1
    else:
        indexCount.append(1)
        originalGenre.append(genre)

df2['genre'].apply(genreSplit)[0]

#duplicating the anime per each genre the anime belongs to
animelist = []
k=0
for i in df2['name']:
    for j in range(indexCount[k]):
        animelist.append(i)
    k+=1

#creating the dataframe with the result
anime_with_genre = pd.DataFrame({
    "Anime":animelist,
    "Genre":originalGenre
})

anime_with_genre.head()

# In[6]:

genreValue = []
keyGenres = []
for i in anime_with_genre['Genre']:

```

```

if i not in keyGenres:
    keyGenres.append(i)
    genreValue.append(1)
else:
    for j in range(len(keyGenres)):
        if keyGenres[j] == i:
            genreValue[j]+=1

genreValue = np.array(genreValue)
keyGenres = np.array(keyGenres)

# In[7]:

fig1, ax1 = plt.subplots()
ax1.pie(genreValue,labels=keyGenres, autopct='%1.1f%%',shadow=True,
startangle=90,radius=6)
plt.show()

# Maximum Number of Anime falls under the category of <b>Comedy</b>, followed by
<b>Action</b>
#
# <b>Comedy , Shounen , Fantasy, Adventure , Action , SciFi</b> and <b>Drama</b>
Alone contribute 45% of the entire Dataset

# ## Top Anime List - Weighted Averages

# In[8]:

#top 20 animes of all time
df2['weighted_rating'] = (df2['rating'] * df2['members']) / df2['members'].sum()
df2.sort_values(by='weighted_rating',ascending=False)[:20] #weighted average

# ## Deathnote Winner , Attack On Titan Runner
# <br>
# 
#

# ## Conclusion

```

```
#
# On analysing the data, It is Understood that The Given Unique User ID and Anime ID
# can be used to create a An User-Item Neural Network Collaborative Filtering
# Recommender System with unique vectors for each unique User as well as Anime whose
# values, the neural network will learn over time
```

RecommenderSystem.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# # Implementation of the NCF Model
#
# Recommender System are simply systems that are designed to recommend stuff to
# users (Movies, Books, etc) in such a way that the sales, views, etc of that particular item
# is maximized.
#
# All the Digital Companies today use these recommender system to recommend stuff to
# us so as to optimize our buying experience and also to maximize their profit
#
# Eg: Netflix recommends movies , Amazon Recommends Products, Facebook Shows
# Ads on the feed based on the user's interests
#
# 

# ## Data Pre-Processing

# In[1]:

# importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf

# In[2]:

# importing the dataset
df = pd.read_csv('recommenderDataset.csv')
```

```
df.head()
```

```
# In[3]:
```

```
#model parameters
user_vocabulary = df['user_id'].max()
anime_vocabulary = df['anime_id'].max()
print(user_vocabulary)
print(anime_vocabulary)
```

```
# In[4]:
```

```
#dataset to train
x1 = np.array(df['user_id']).reshape(-1)
x2 = np.array(df['anime_id']).reshape(-1)
y_train = df['rating_x']
```

```
# ## Building and Training The Network
```

```
# In[5]:
```

```
#creating the model
user = tf.keras.layers.Input(1,)
anime = tf.keras.layers.Input(1,)

#Embedding Layers for user
embedd_x = tf.keras.layers.Embedding(user_vocabulary+1,10)(user)
embedd_y = tf.keras.layers.Embedding(anime_vocabulary+1,10)(anime)

#concatenate the outputs
x = tf.keras.layers.Concatenate()([embedd_x,embedd_y])
x = tf.keras.layers.Dense(24,activation=tf.keras.activations.relu)(x)
x = tf.keras.layers.Dense(1,activation=tf.keras.activations.sigmoid)(x)

model = tf.keras.models.Model(inputs=[user,anime],outputs=x)
```

```
# In[6]:
```

```
#compile and train
```

```
model.compile(loss=tf.keras.losses.binary_crossentropy,optimizer=tf.keras.optimizers.Adam(learning_rate=0.05),metrics=['accuracy'])
train = model.fit([x1,x2],y_train,epochs=10,batch_size=1024)
```

```
# In[7]:
```

```
#save and load
model.save("recommenderSystem.h5")
model = tf.keras.models.load_model('recommenderSystem.h5')
```

```
# ## Network's Performance Analysis
```

```
# In[8]:
```

```
#get the accuracy
y_pred = model.predict([x1,x2])
y_pred = y_pred.reshape(-1)
(y_pred.round() == y_train).mean()
```

```
# In[9]:
```

```
#visualizing the accuracy and the loss
plt.figure(figsize=(8,6))
plt.plot(train.history['accuracy'],label="accuracy")
plt.plot(train.history['loss'],label="loss")
plt.legend()
```

```
# ## Working of the Recommender System
```

```
# In[10]:
```

```
#example anime
df[ df['rating_x'] == 0.0 ]
```

```
# In[11]:
```

```
#Recommender System Working
```

```

user_id = int(input("Enter the User Id: "))
anime_id = int(input("Enter the Anime Id: "))
y_pred = model.predict([np.array([user_id]),np.array([anime_id])])
if y_pred.round():
    print(f"The User Will probably Like {df[df['anime_id'] == anime_id]
['name'].iloc[0]}")
else:
    print(f"The user Will probably Not Like {df[df['anime_id'] == anime_id]
['name'].iloc[0]}")

```

In[12]:

```

# New Predictions
user_id = int(input("Enter the User Id: "))
anime_id = int(input("Enter the Anime Id: "))
y_pred = model.predict([np.array([user_id]),np.array([anime_id])])
if y_pred.round():
    print(f"The User Will probably Like {df[df['anime_id'] == anime_id]
['name'].iloc[0]}")
else:
    print(f"The user Will probably Not Like {df[df['anime_id'] == anime_id]
['name'].iloc[0]}")

```

Conclusion

#

Similarly, A particular User's Id can be matched with a new Anime's Id that the user haven't watched and the model could predict if the user would like the model or not

CHAPTER 7

FUTURE ENHANCEMENT AND CONCLUSION

7.1 FUTURE ENHANCEMENT

The main Future enhancements would be to remove the skewness in the data-set by collecting more data regarding the anime that the user's don't like and improving the complexity of the model to increase the accuracy.

7.2 CONCLUSION

This project is used for recommending anime to user based on their previous watching history and based on the anime's features. This results in a personalized recommendation that keeps the user in continuous interaction with anime.

CHAPTER 8

REFERENCES

- [1] Anime Dataset from Kaggle
- [2] Introductions to Recommender Systems from Towards Data Science
- [3] Hybrid Recommender System with Neural Networks from Towards Data Science
- [4] Collaborative Filtering from Analytics Vidhya
- [5] Pandas Documentation
- [6] Matplotlib Documentation