# ACKNOWLEDGEMENT

We have found this rare opportunity to evince a word of thanks to all those who played a key role in the successful completion of our project. We sincerely thank our Head of Department Dr**.** Parth Shah Sir for giving the chance as well as support for all the time being. And his able guidance and continuous encouragement made us work in all the challenges during project development. We express deep gratitude to assistant Prof. Jalpesh Vasa internal project guide from Faculty of Engineering, CHARUSAT for their valuable suggestions, help and moral support. We also thank to all those who could not find a separate name but have helped directly and indirectly.

Divya Rai (16IT101)

Harita Shah (16IT121)

# ABSTRACT

In today’s digital world where there is an endless variety of content to be consumed like books, videos, articles, movies, etc., finding the content of one’s liking has become an irksome task. This is where recommender system comes into picture where the content providers recommend users the content according to the users’ liking. Recommender systems have become ubiquitous in our lives. Yet, currently, they are far from optimal. In this project, we attempt to understand the item-based collaborative recommendation systems on the MovieLens dataset. We attempt to build a scalable model to perform this analysis.

# Table of Contents

* [Acknowledgement 01](#_ACKNOWLEDGE)
* [Abstract 02](#_TOC_250004)
* Chapter 1 Introduction 05

Project Overview 05

Scope 05

Objective 05

* Chapter 2 System Analysis 06

User Characteristics 06

Tools & Technology…………....………………………………… 06

* [Chapter 3 Implementation…………………………………………………… 0](#_TOC_250001)8

Code 08

Snapshots of output 09

* [Chapter 4 Constraints](#_TOC_250000) 12
* Chapter 5 Conclusion 13
* References 14

**List of figures**

[Figure 1 8](#_Toc528086358)

[Figure 2 8](#_Toc528086359)

[Figure 3 9](#_Toc528086360)

[Figure 4 9](#_Toc528086361)

[Figure 5 10](#_Toc528086362)

[Figure 6 10](#_Toc528086363)

[Figure 7 11](#_Toc528086364)

**CHAPTER 1**

**INTRODUCTION**

* 1. **Project Overview**

A recommendation system is a type of information filtering system which attempts to predict the preferences of a user, and make suggests based on these preferences. There are a wide variety of applications for recommendation systems. These have become increasingly popular over the last few years and are now utilized in most online platforms that we use.

## Scope

### What it can do

* + - Recommends movies to users according to the movies they rated before

### What it can’t do

* + - It cannot give recommendations if the user has not rated any movie
  1. **Objective**
* Applying machine learning in real-time using Collaborative Filtering.
* Parsing data retrieved from a dataset and predicting user preference.
* Evaluating item based collaborative approach of recommender systems
* To recommend movies to its users based on their viewing history and ratings that they provide.

**CHAPTER 2**

**SYSTEM ANALYSIS**

## User Characteristics

User should have given rating according to his/her preferences and watched movies to get desirable recommendation of new movies.

## Tools and Technology

* + 1. **Software requirement**
       1. Operating system : any with python installed in it
       2. Jupyter Notebook in Anaconda Framework

## Programming Language

1. python

## About Collaborative filtering:

Collaborative filtering system recommends items based on similarity measures between users and/or items. The system recommends items preferred by similar users. This is based on the scenario where a person asks his friends, who have similar tastes, to recommend him some movies. Collaborative filtering is based on the assumption that people who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past.

Features of collaborative filtering based systems:

• It is dependent on the relation between users which implies that it is content-independent.

• CF recommender systems can suggest serendipitous items by observing similar-minded people’s behavior.

• They can make real quality assessment of items by considering other peoples experience.

**Datasets**

MovieLens data sets were collected by the GroupLens Research Project

at the University of Minnesota.

This data set consists of:

\* 100,000 ratings (1-5) from 943 users on 1682 movies.

\* Each user has rated at least 20 movies.

\* Simple demographic info for the users (age, gender, occupation, zip)

**Correlation**

Correlation between sets of data is a measure of how well they are related. The most common measure of correlation in stats is the Pearson Correlation. The full name is the Pearson Product Moment Correlation (PPMC). It shows the linear relationship between two sets of data. In simple terms, it answers the question, Can I draw a line graph to represent the data? Two letters are used to represent the Pearson correlation: Greek letter rho (ρ) for a population and the letter “r” for a sample.

The PPMC is not able to tell the difference between dependent variables and [independent variables](http://www.statisticshowto.com/independent-variable-definition/). For example, if you are trying to find the correlation between a high calorie diet and diabetes, you might find a high correlation of .8. However, you could also get the same result with the variables switched around. In other words, you could say that diabetes causes a high calorie diet. That obviously makes no sense. Therefore, as a researcher you have to be aware of the data you are plugging in. In addition, the PPMC will not give you any information about the slope of the line; it only tells you whether there is a relationship.

**CHAPTER 3**

**IMPLEMENTATION**

* 1. **Python Code:**

1. import pandas as pd
2. r\_cols=['user\_id','movie\_id','rating']

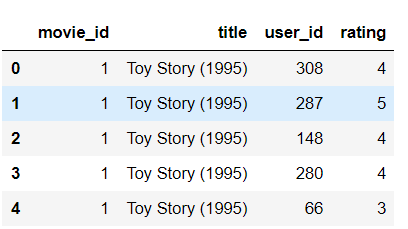
ratings = pd.read\_csv('datasets/movielens/u.data', sep='\t', names=r\_cols, usecols=range(3))

m\_cols=['movie\_id','title']

movies = pd.read\_csv('datasets/movielens/u.item', sep='|', names=m\_cols,usecols=range(2))

ratings=pd.merge(movies,ratings)

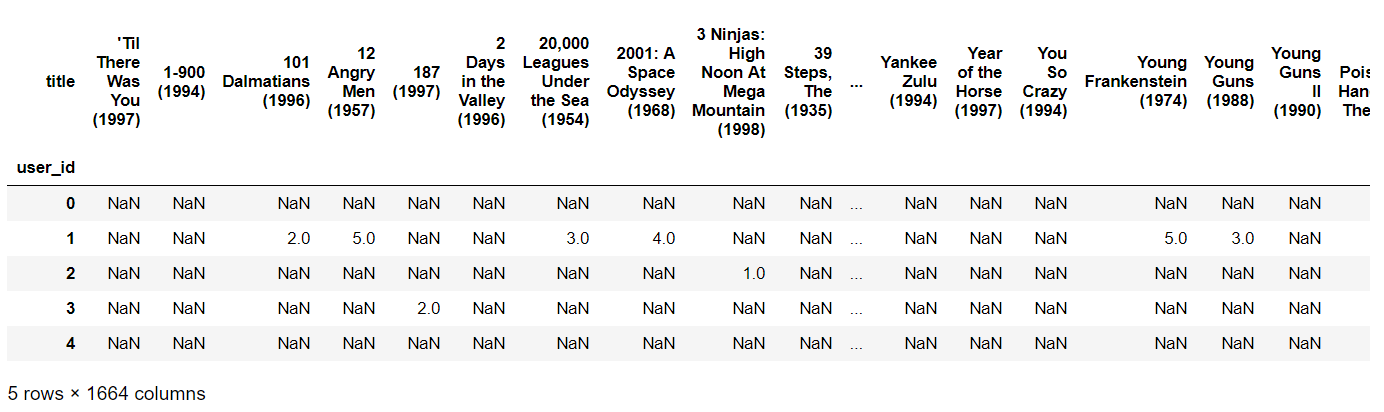
ratings.head()



Figure

1. userRatings=ratings.pivot\_table(index=['user\_id'],columns=['title'],values='rating')

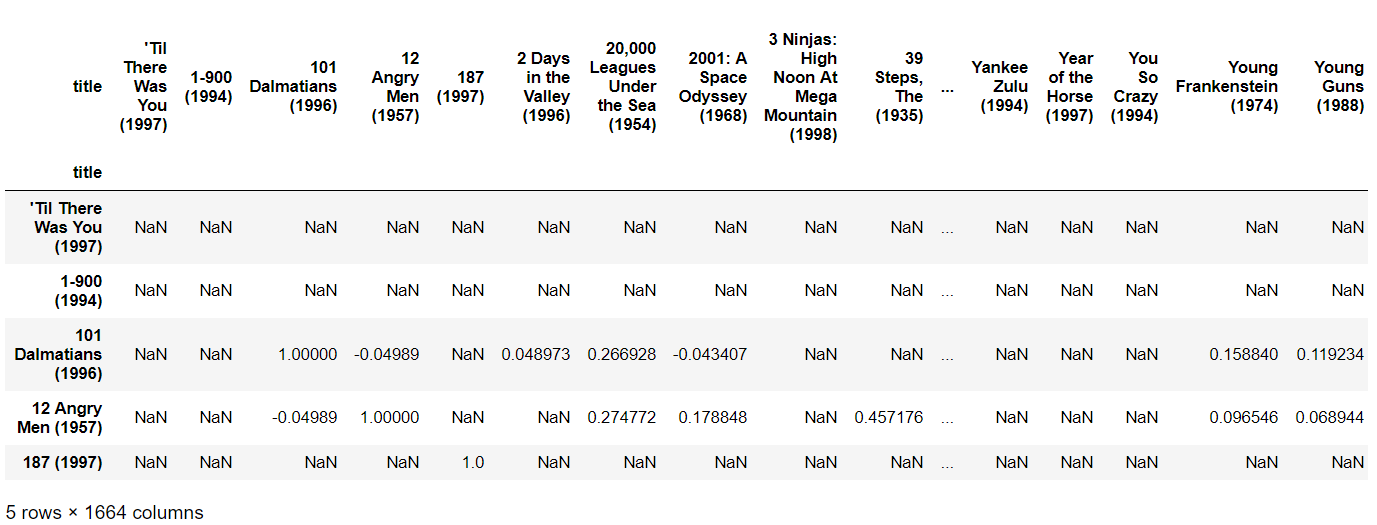
userRatings.head()



Figure

1. corrMatrix=userRatings.corr(method="pearson",min\_periods=20)

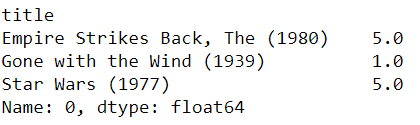
corrMatrix.head()



Figure

1. myRatings=userRatings.loc[0].dropna()

myRatings



Figure

1. simCandidates=pd.Series()

for i in range(0,len(myRatings.index)):

print"Adding sims for "+ myRatings.index[i]+"..."

sims=corrMatrix[myRatings.index[i]].dropna()

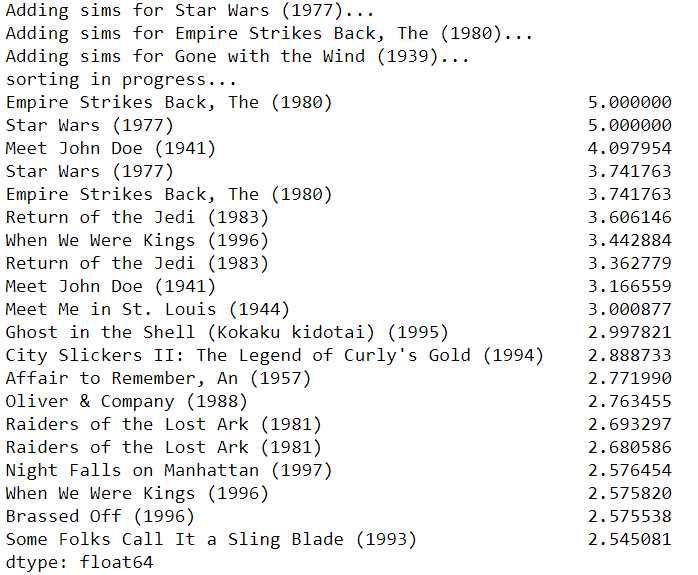
sims=sims.map(lambda x: x\* myRatings[i])

simCandidates=simCandidates.append(sims)

print"sorting in progress..."

simCandidates.sort\_values(inplace=True, ascending=False)

print simCandidates.head(20)



Figure

1. simCandidates = simCandidates.groupby(simCandidates.index).sum()
2. simCandidates.sort\_values(inplace = True, ascending = False)

simCandidates.head(10)



Figure

1. filteredSims = simCandidates.drop(myRatings.index)

filteredSims.head(20)



Figure

**CHAPTER 4**

**CONSTRAINTS**

• Early rater problem: Collaborative filtering systems cannot provide recommendations for new items since there are no user ratings on which to base a prediction.

• Gray sheep: In order for CF based system to work, group with similar characteristics are needed. Even if such groups exist, it will be very difficult to recommend users who do not consistently agree or disagree to these groups.

• Sparsity problem: In most cases, the amount of items exceed the number of users by a great margin which makes it difficult to find items that are rated by enough people.

# CHAPTER 5

**Conclusion**

The work of movie recommendation system is to recommend movies to its users based on their viewing history and ratings that they provide. Personalized recommendation engines help millions of people narrow the universe of potential films to fit their unique tastes. Collaborative filtering is the prime approaches to provide recommendation to users.

# References

* <https://www.sciencedirect.com/science/article/pii/S1110866515000341>
* <https://www.coursera.org/learn/machine-learning/lecture/uG59z/content-based-recommendations>
* <https://www.dataquest.io/blog/pandas-cheat-sheet/>
* <https://www.analyticsvidhya.com/blog/2016/06/quick-guide-build-recommendation-engine-python/>
* <https://pdfs.semanticscholar.org/767e/ed55d61e3aba4e1d0e175d61f65ec0dd6c08.pdf>