Meghnad Saha Institute of Technology

Major Project Report On

Movie Recommendation System

By

Name	University Roll No
Sudhanya Roy	14201017002
Subhadeep Palai	14201017004
Binamrata Prasad	14201017019
Arpan Kumar Charan	14201017020

SEMESTER: VI

MCA

Maulana Abul Kalam Azad University of Technology

PROJECT CERTIFICATE

This is to certify that the project entitled "**Movie Recommendation System**" has been prepared according to the regulations of degree of Masters in Computer Application (MCA) under the University of "Maulana Abul Kalam Azad University of Technology". The project being submitted by -

_____

(STUDENT SIGNATURE)

Students of Masters in Computer Application (MCA), 3rd year 2nd semester of ***Meghnad Saha Institute of Technology*** (affiliated to Maulana Abul Kalam Azad University of Technology) have fulfilled the requirement for submission of this.

The whole procedure has been carried out under my supervision and guidance. I have gone through this project and have seen that it is fulfilling the requirements of Major Project under MAKAUT, WB.

Dated: /07/2020

_____ _____

(INTERNAL PROJECT GUIDE) **(EXTERNAL PROJECT GUIDE)**

_____

(HEAD OF THE DEPARTMENT)

_____

(EXAMINER)

ACKNOWLEDGEMENT

We would take the opportunity to thank **Prof. Aparna Datta**, Head of Department, Meghnad Saha Institute of Technology for allowing us to form a group of four people and for providing us with all the necessary facilities to make our project Work and of worth.

We are also thankful to **Prof. Saubhik Goswami**, our Project Supervisor for constantly supporting and guiding us and also for giving us invaluable insights. His guidance and his words of encouragement motivated us to achieve our goal and impetus to excel.

Last but not the least we thank all our friends for their cooperation and encouragement that they have bestowed on us.

Signature:

_____

(Sudhanya Roy)

Signature:

_____

(Binamrata Prasad)

Signature:

_____

(Arpan Kumar Charan)

Signature:

_____

(Subhadeep Palai)

ABSTRACT

A recommendation system is a system that provides suggestions to users for certain resources like books, movies, songs, etc., based on some data-set. Movie recommendation systems usually predict what movies a user will like based on the attributes present in previously liked movies. We built a recommendation engine for this system. A lot of factors can be considered while designing a movie recommendation system like the genre of the movie, actors present in it or even the director of the movie. The engine recommends a movie based on all content-based attributes.

We have fitted the engine into a web application which can be used by any user. It shows some movies from our data-set. After selecting a movie, it recommends some similar moves from the data-set.

The website http://13.232.227.191 of this application has been showcased for a demo. The project can be clone of fork from Github. https://github.com/rsudhanya/MovieRecommendationEngineContentBasedDjango.git

Contents

Overview of the System

PREFACE

Learning comes from doing. To learn something one has to go through Practical conditions. Recognizing this fact, the university has made it essential for MCA (MATER OF COMPUTER APPLICATION) students to undergo major project Training. During this period, the students learns about the functioning of the organization and the actual business environment. Also this training helps the student how to implement the theoretical knowledge into practical life in our day to day life.
This project was undertaken at **MEGHNAD SAHA INSTITUTE OF TECHNOLOGY, KOLKATA** during industrial training engineering to automate the system .The project is named as **Movie Recommendation System.**

The purpose of this report is to assemble under one cover a sufficient body of knowledge about management and development a successful software engineering project. The following quotes outline the basic idea behind this technical report.

This report is about the adaptation of the techniques of project development and reflects the practice and methods of software engineering project this report is intended for:

- *Project managers*—the report delivers the necessary information of the process a software development project
- *Project coordinators* —the tutorial presents the state of the practice in software development and management techniques.
- *Software engineers, programmers, analysts, and other computer personnel* — the report contains a general description of—and problems in—software engineering project development, plus a number of methodologies and techniques for managing a software development project.

Objective

•Develop in depth understanding of the key technologies in data science and business analytics: data mining, machine learning, visualization techniques, predictive modeling, and statistics.

•Practice problem analysis and decision-making.

•Gain practical, hands-on experience with statistics programming languages and big data tools through coursework and applied research experiences.

•Apply quantitative modeling and data analysis techniques to the solution of real world business problems, communicate findings, and effectively present results using data visualization techniques.

•Demonstrate knowledge of statistical data analysis techniques utilized in business decision making.

•Apply principles of Data Science to the analysis of business problems. Apply algorithms to build machine intelligence.

•Demonstrate use of team work, leadership skills, decision making and organization theory.

Introduction

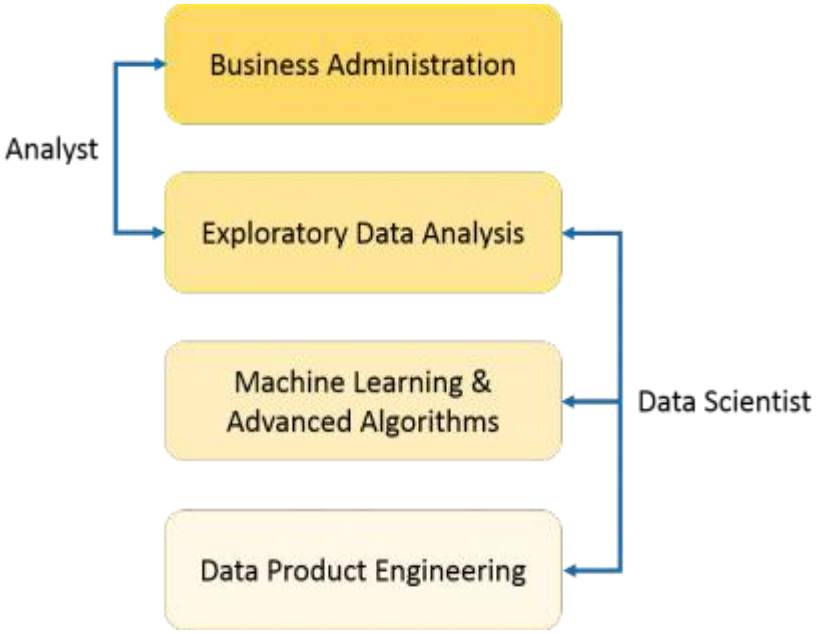
To understand the engine of the Movie Recommendation System several things need to know.

➔ *Data Science*

As the world entered the era of big data, the need for its storage also grew. It was the main challenge and concern for the enterprise industries until 2010. The main focus was on building framework and solutions to store data. Now when Hadoop and other frameworks have successfully solved the problem of storage, the focus has shifted to the processing of this data. Data Science is the secret sauce here. All the ideas which you see in Hollywood sci-fi movies can actually turn into reality by Data Science. Data Science is the future of Artificial Intelligence. Therefore, it is very important to understand what is Data Science and how can it add value to your business.

Use of the term Data Science is increasingly common, but what does it exactly mean? What skills do you need to become Data Scientist? What is the difference between BI and Data Science? How are decisions and predictions made in Data Science? These are some of the questions that will be answered further.

First, let's see what is Data Science. Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data. How is this different from what statisticians have been doing for years?



As you can see from the above image, a Data Analyst usually explains what is going on by processing history of the data. On the other hand, Data Scientist not only does the exploratory analysis to discover insights from it, but also uses various advanced machine learning algorithms to identify the occurrence of a particular event in the future. A Data Scientist will look at the data from many angles, sometimes angles not known earlier.

Data science is an inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data. Data science is related to data mining, deep learning and big data.

Data science is a "concept to unify statistics, data analysis, machine learning, domain knowledge and their related methods" in order to "understand and analyze actual phenomena" with data. It uses techniques and theories drawn from many fields within the context of mathematics, statistics, computer science, domain knowledge and information science. Turing award winner Jim Gray imagined data science as a "fourth paradigm" of science (empirical, theoretical, computational and now data-driven) and asserted that "everything about science is changing because of the impact of information technology" and the data deluge

So, Data Science is primarily used to make decisions and predictions making use of predictive causal analytics, prescriptive analytics (predictive plus decision science) and machine learning.

➔ *Machine Learning*

Machine learning is the science of getting computers to act without being explicitly programmed. In the past decade, machine learning has given us self-driving cars, practical speech recognition, effective web search, and a vastly improved understanding of the human genome. Machine learning is so pervasive today that you probably use it dozens of times a day without knowing it. Many researchers also think it is the best way to make progress towards human-level AI.'

Machine learning (**ML**) is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email

filtering and computer vision, where it is difficult or infeasible to develop conventional algorithms to perform the needed tasks.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

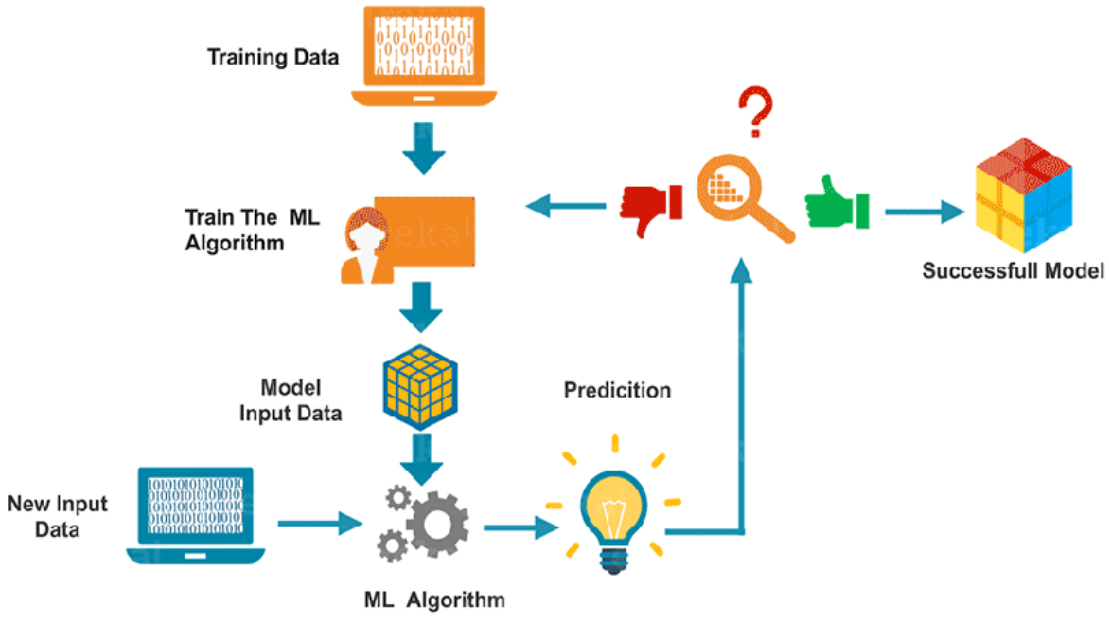
In this class, you will learn about the most effective machine learning techniques, and gain practice implementing them and getting them to work for yourself. More importantly, you'll learn about not only the theoretical underpinnings of learning, but also gain the practical know-how needed to quickly and powerfully apply these techniques to new problems. Finally, you'll learn about some of Silicon Valley's best practices in innovation as it pertains to machine learning and AI.

This course provides a broad introduction to machine learning, data mining, and statistical pattern recognition. Topics include: (i) Supervised learning (parametric/non-parametric algorithms, support vector machines, kernels, neural networks). (ii) Unsupervised learning (clustering, dimensionality reduction, recommendations systems, deep learning). (iii) Best practices in machine learning (bias/variance theory; innovation process in machine learning and AI). The course will also draw from numerous case studies and applications, so that you'll also learn how to apply learning algorithms to building smart robots (perception, control), text understanding (web search, anti-spam), computer vision, medical informatics, audio, database mining, and other areas.

How does Machine Learning Work?

Machine Learning algorithm is trained using a training data set to create a model. When new input data is introduced to the ML algorithm, it makes a prediction on the basis of the model.

The prediction is evaluated for accuracy and if the accuracy is acceptable, the Machine Learning algorithm is deployed. If the accuracy is not acceptable, the Machine Learning algorithm is trained again and again with an augmented training data set.



Types of Machine Learning

Machine learning is sub-categorized to three types:

➢ *Supervised Learning*

Supervised Learning is the one, where you can consider the learning is guided by a teacher. We have a data-set which acts as a teacher and its role is to train the model or the machine. Once the model gets trained it can start making a prediction or decision when new data is given to it.

➢ *Unsupervised Learning*

The model learns through observation and finds structures in the data. Once the model is given a data-set, it automatically finds patterns and relationships in the data-set by creating clusters in it. What it cannot do is add labels to the cluster, like it cannot say this a group of apples or mangoes, but it will separate all the apples from mangoes. suppose we presented images of apples, bananas and mangoes to the model, so what it does, based on some patterns and relationships it creates clusters and divides the dataset into those clusters. Now if a new data is fed to the model, it adds it to one of the created clusters.

➢ *Reinforcement Learning*

It is the ability of an agent to interact with the environment and find out what is the best outcome. It follows the concept of hit and trial method. The agent is rewarded or penalized with a point for a correct or a wrong answer, and on the basis of the positive reward points gained the model trains itself. And again once trained it gets ready to predict the new data presented to it.

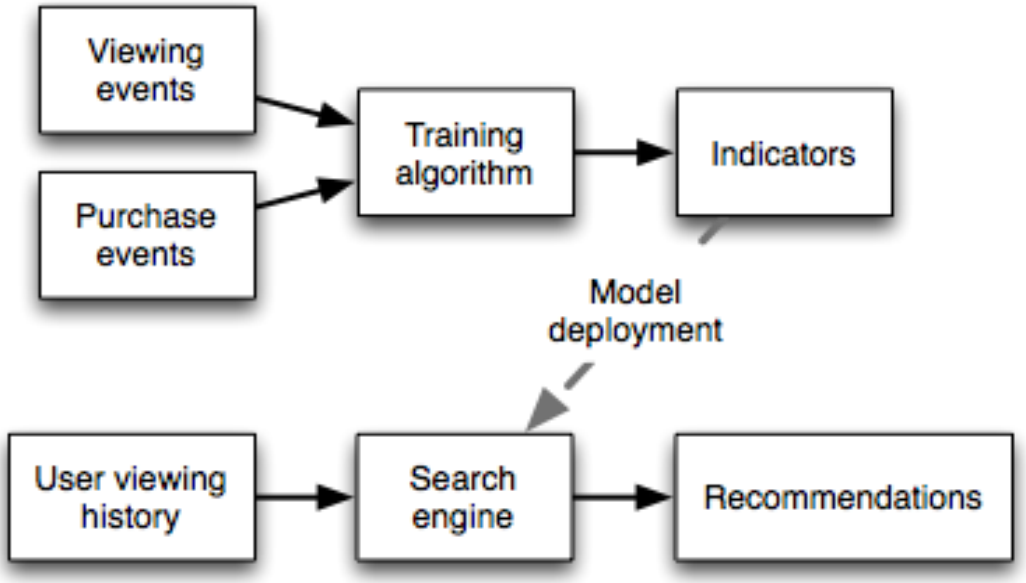
■ Recommendations

How does YouTube know what video you might want to watch next? How does the Google Play Store pick an app just for you? Magic? No, in both cases, an ML-based recommendation model determines how similar videos and apps are to other things you like and then serves up a recommendation.

A recommendation system, or a recommendation system (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. They are primarily used in commercial applications.

Recommendation systems are utilized in a variety of areas and are most commonly recognized as playlist generators for video and music services like Netflix, YouTube and Spotify, product recommendations for services such as Amazon, or content recommendations for social media platforms such as Facebook and Twitter.These systems can operate using a single input, like music, or multiple inputs within and across platforms like news, books, and search queries. There are also popular recommendations systems for specific topics like restaurants and online dating. Recommendation systems have also been developed to explore research articles and experts,collaborators, and financial services.

A recommendation system helps users find compelling content in a large corpora. For example, the Google Play Store provides millions of apps, while YouTube provides billions of videos. More apps and videos are added every day. How can users find new compelling new content? Yes, one can use search to access content. However, a recommendation engine can display items that users might not have thought to search for on their own.



- Approaches

Although machine learning (ML) is commonly used in building recommendation systems, it doesn't mean it's the only solution. There are many ways to build a recommendation system. simpler approaches, for example, we may have very few data, or we may want to build a minimal solution fast etc.

◆ *Popularity based*

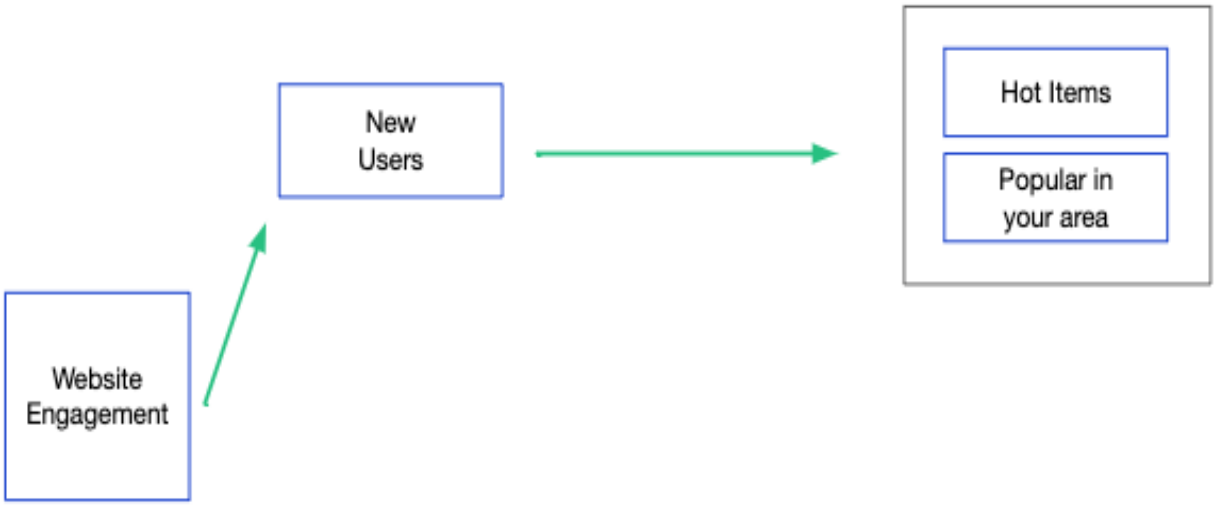
As the name suggests Popularity based recommendation system works with the trend. It basically uses the items which are in trend right now. For example, if any product which is usually bought by every new user then there are chances that it may suggest that item to the user who just signed up.

There are some problems as well with the popularity based recommendation system and it also solves some of the problems with it as well.

The problems with popularity based recommendation system is that the personalization is not available with this method i.e. even though you

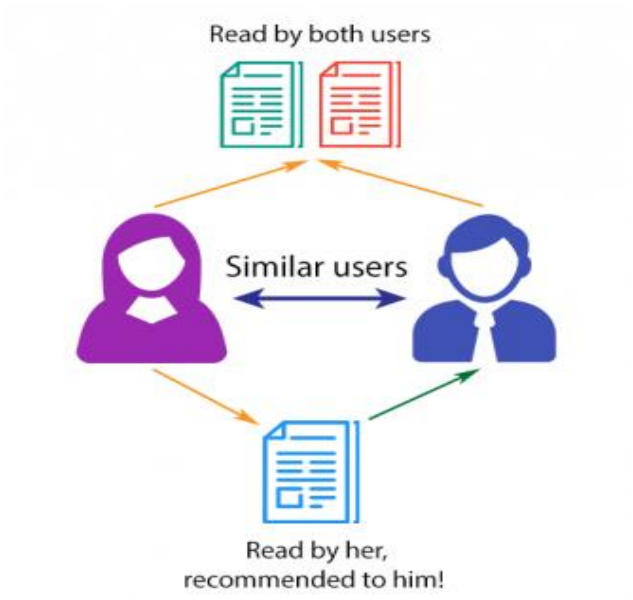
know the behavior of the user you cannot recommend items accordingly.

For example, In shopping store we can suggest popular dresses by purchase count.

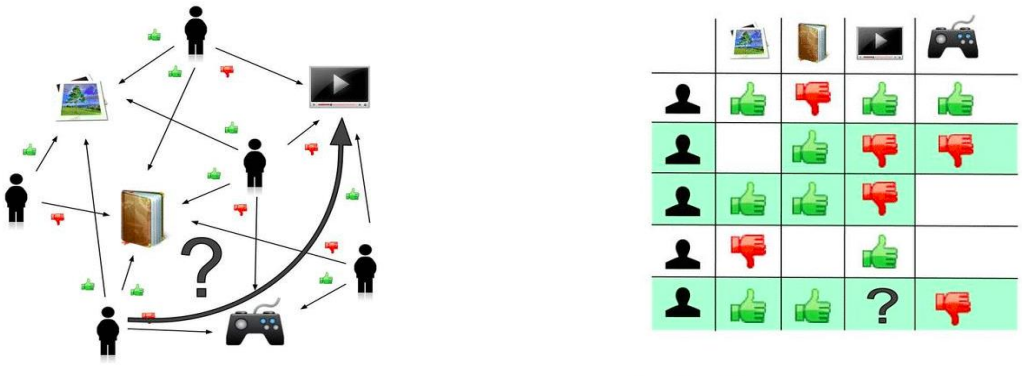


- Collaborative filtering

Collaborative filtering (CF) is a technique used by recommendation systems. Collaborative filtering has two senses, a narrow one and a more general one. In the newer, narrower sense, collaborative filtering is a method of making automatic prediction(filtering) about the interests of a user by collecting preferences or taste information from many users(collaborating). The underlying assumption of the collaborative filtering approach is that if a person *A* has the same opinion as a person *B* on an issue, A is more likely to have B's opinion on a different issue than that of a randomly chosen person. For example, a collaborative filtering recommendation system for television tastes could make predictions about which television show a user should like given a partial list of that user's tastes (likes or dislikes).Note that these predictions are specific to the user, but use information gleaned from many users. This differs from the simpler approach of giving an average(non-specific) score for each item of interest, for example based on its number of votes.

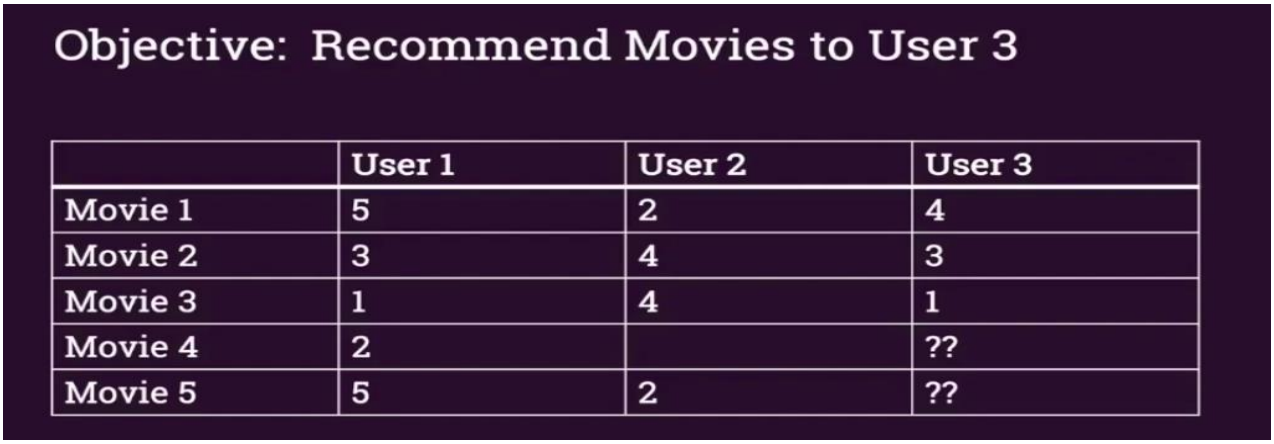


In the more general sense, collaborative filtering is the process of filtering for information or patterns using techniques involving collaboration among multiple agents, viewpoints, data sources, etc.Applications of collaborative filtering typically involve very large data sets. Collaborative filtering methods have been applied to many different kinds of data including: sensing and monitoring data, such as in mineral exploration, environmental sensing over large areas or multiple sensors; financial data, such as financial service institutions that integrate many financial sources; or in electronic commerce and web applications where the focus is on user data, etc. The remainder of this discussion focuses on collaborative filtering for user data, although some of the methods and approaches may apply to the other major applications as well.



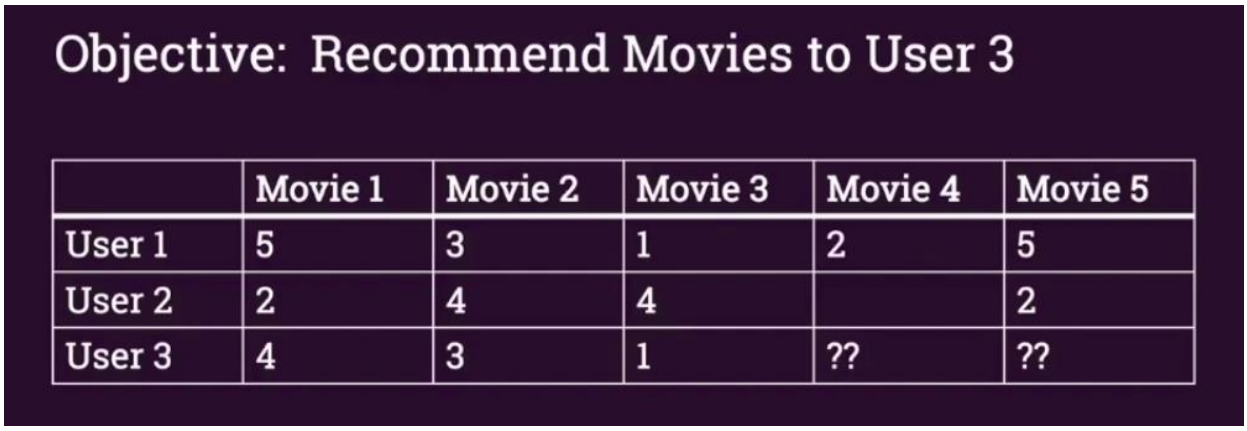
These image shows an example of predicting of the user's rating using collaborative filtering. At first, people rate different items (like videos, images, games). After that, the system is making predictions about user's rating for an item, which the user hasn't rated yet. These predictions are built upon the

existing ratings of other users, who have similar ratings with the active user. For instance, in our case the system has made a prediction, that the active user won't like the video.



User-to-User CF

We can see that User3 is most likely User1. So 4th recommended movie for User3 is obviously Movie5.



Item-to-Item CF

We can see that User3 is most likely User1. So 4th recommended movie for User3 is obviously Movie5.

- Content-based filtering

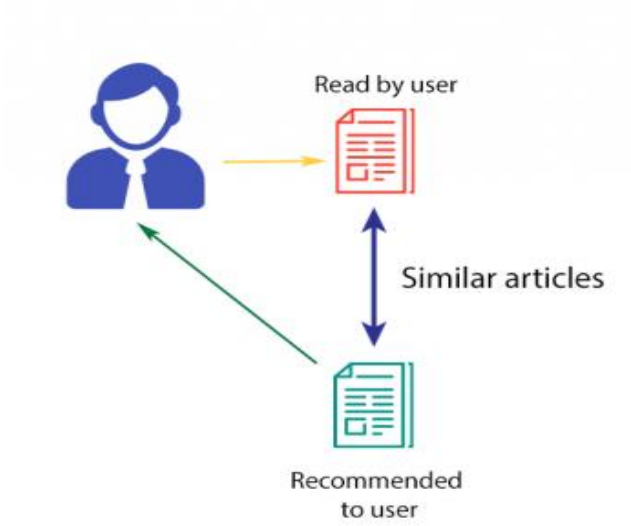
Another common approach when designing recommendation systems is content-based filtering. Content-based filtering methods are based on a description of the item and a profile of the user's preferences.These methods are best suited to situations where there is known data on an item (name, location, description, etc.), but not on the user. Content-based recommendations treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on an item's features.

In this system, keywords are used to describe the items and a user profile is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past, or is examining in the present. It does not rely on a user sign-in mechanism to generate this often temporary profile. In particular, various candidate items are compared with items previously rated by the user and the best-matching items are recommended. This approach has its roots in information retrieval and information filtering research.

To create a user profile, the system mostly focuses on two types of information: I. A model of the user's preference, II. A history of the user's interaction with the recommendation system.

Basically, these methods use an item profile (i.e., a set of discrete attributes and features) characterizing the item within the system. To abstract the features of the items in the system, an item presentation algorithm is applied. A widely used algorithm is the tf-idf representation (also called vector space representation).The system creates a content-based profile of users based on a weighted vector of item features. The weights denote the importance of each feature to the user and can be computed from individually rated content vectors using a variety of techniques. Simple approaches use the average values of the rated item vector while other sophisticated methods use machine learning techniques such as Bayesian Classifiers, cluster analysis,

decision trees, and artificial neural networks in order to estimate the probability that the user is going to like the item.



A key issue with content-based filtering is whether the system is able to learn user preferences from users' actions regarding one content source and use them across other content types. When the system is limited to recommending content of the same type as the user is already using, the value from the recommendation system is significantly less than when other content types from other services can be recommended. For example, recommending news articles based on browsing of news is useful, but would be much more useful when music, videos, products, discussions etc. from different services can be recommended based on news browsing. To overcome this, most content-based recommendation systems now use some form of hybrid system.

Content-based recommendation systems can also include opinion-based recommendation systems. In some cases, users are allowed to leave text review or feedback on the items. These user-generated texts are implicit data for the recommendation system because they are potentially rich resource of both feature/aspects of the item, and users' evaluation/sentiment to the item. Features extracted from the user-generated reviews are improved meta-data

of items, because as they also reflect aspects of the item like meta-data, extracted features are widely concerned by the users.

The content-based filtering is used to build the engine of the "Movie Recommendation System" project.

- *Hybrid recommendation*

Most recommendation systems now use a hybrid approach, combining collaborative filtering, content-based filtering, and other approaches . There is no reason why several different techniques of the same type could not be hybridized. Hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model. Several studies that empirically compare the performance of the hybrid with the pure collaborative and content-based methods and demonstrated that the hybrid methods can provide more accurate recommendations than pure approaches. These methods can also be used to overcome some of the common problems in recommendation systems such as cold start and the sparsity problem, as well as the knowledge engineering bottleneck in knowledge-based approaches.

Netflix s a good example of the use of hybrid recommendation systems.The website makes recommendations by comparing the watching and searching habits of similar users (i.e., collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (content-based filtering).

Project Overview

Project Name:	Movie Recommendation System
Institute:	Meghnad Saha Institute of Technology
Project Type:	Web Application with Data Science & ML
Front End:	HTML/CSS, JavaScript
Back End:	Python
Platform:	Ubuntu Linux 18.04
Host:	AWS (EC2)

Advantages and disadvantages

- Advantages

✓ Understand Customer Behavior

The web applications can also use machine learning algorithms to understand customer behavior and boost customer engagement. An e-commerce application can use machine learning to monitor and understand customer conversation related to a product. It can even use the algorithm to know the features and functionality expected by the customers. Also, an enterprise can use machine learning to communicate with customers more efficiently through contact us forms. The machine learning algorithm can easily analyze the customer queries and companies, and transfer the query to the relevant team. Hence, the sales and customer support teams can easily enhance customer experience by resolving issues faster.

✓ Deliver Personalized Content and Information

Facebook already use machine learning algorithm to personalize newsfeed of each user. The technology used by Facebook combines predictive analytics and statistical analysis to detect patterns based on the user's data. Also, it personalizes the newsfeed of the users based on the detected patterns. The machine learning technology identifies pattern based on the content read and posts liked by the user. Based on the identified pattern, it displays similar posts and content earlier in the feed. While developing web applications, programmers can embed similar machine learning technology to deliver personalized content and information to each user based on his personal choices and preferences.

✓ Speedup Product Discovery

Large companies like Apple, Google, and Microsoft are already using machine learning algorithms to deliver smart search results to each user. While developing e-commerce applications, the programmers can use machine learning algorithm to help customers find products faster. The developers can use specific machine learning algorithm to deliver quality and relevant information to users. Also, they can use the technology to help customers choose products based on their precise needs. The e-commerce portal can further use machine learning to make the customers browse through only relevant products.

- Disadvantages

- Data Acquisition

One of the most painful points in the field of Data Science and Machine Learning is the acquisition of data. Additionally, collecting data comes with a cost. Also, it so happens that when we are collecting data from surveys, it might contain a large volume of bogus and incorrect data. Many times we do face a situation where we find an imbalance in data which leads to poor accuracy of models. These reasons make data acquisition a massive disadvantage.

- Highly error-prone

“Garbage In Garbage Out” is the thing to always remember in this technology. The data we push in the models as training data must be clean and accurate for the problem we are solving. Being easy to automate processes using Machine Learning, it sometimes does happen that data in between is improper. This might cause incorrect results or errors. For example, we might witness a situation where customers may be classified as defaulters or customers are recommended products not related to their search history or patterns.

- Algorithm Selection

A Machine Learning problem can implement various algorithms to find a solution. It is a manual and tedious task to run models with different algorithms and identify the most accurate algorithm based on the results. This is a disadvantage.

- Time-consuming

Machine Learning models are capable of processing huge amounts of data. Larger the volume of data, the time to learn from data and process it also increases. Sometimes it might also mean additional resources for computing.

Scope

- The data-set contains approximately 5000 movie data.
- The recommendation engine recommends movie after selecting a movie from our data-set.
- The application arrange the similar movies best choice to worst choice from all movies of data-set
- The application has a search facility from the data-set.
- There is a reset button to reset recommendation.

Requirements Analysis

Specification	Client	Server
Processor	Any x86 or x64 arch based	Any 2.4GHz, One core, x86 or x64 arch based or above
RAM	512MB or above	1024MB or above
OS	Any GUI based	Ubuntu 18.04 or similar or above
Storage	50MB or above	2048GB or above
Software	Any standard Web Browser supported JAVASCRIPT interpreter(E.G: Google Chrome Version 76.0.3809.132)	Python 3.6 and above, Nginx, Supervisor, Python Packages – pandas, numpy, scikit-learn, django, django-mathfilters , gunicorn

Tools & Technologies

Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisp tradition. It has `filter`, `map`, and `reduce` functions; list comprehensions, dictionaries, sets, and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

The language's core philosophy is summarized in the document *The Zen of Python* (*PEP 20*), which includes aphorisms such as:

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach. Python can serve as a scripting language for web applications, e.g., via mod wsgi for the Apache web server. With Web Server Gateway Interface, a standard API has evolved to facilitate these applications. Web frameworks like Django, Pylons, Pyramid, TurboGears, web2py, Tornado, Flask, Bottle and Zope support developers in the design and maintenance of complex applications. Pyjs and IronPython can be used to develop the client-side of Ajax-based applications. SQLAlchemy can be used as data mapper to a relational database. Twisted is a framework to program communications between computers, and is used (for example) by Dropbox.

Libraries such as NumPy, SciPy and Matplotlib allow the effective use of Python in scientific computing, with specialized libraries such as Biopython and Astropy providing domain-specific functionality. SageMath is a mathematical software with a notebook interface programmable in Python: its library covers many aspects of mathematics, including algebra, combinatorics, numerical mathematics, number theory, and calculus.

Python has been successfully embedded in many software products as a scripting language, including in finite element method software such as Abaqus, 3D parametric modeler like FreeCAD, 3D animation packages such as 3ds Max, Blender, Cinema 4D, Lightwave, Houdini, Maya, modo, MotionBuilder, Softimage, the visual effects compositor Nuke, 2D imaging programs like GIMP, Inkscape, Scribus and Paint Shop Pro, and musical notation programs like scorewriter and capella. GNU Debugger uses Python as a pretty printer to show complex structures such as C++ containers. Esri promotes Python as the best choice for writing scripts in ArcGIS. It has also been used in several video games, and has been adopted as first of the three available programming languages in Google App Engine, the other two being Java and Go.

Python is commonly used in artificial intelligence projects and machine learning projects with the help of libraries like TensorFlow, Keras, Pytorch and Scikit-learn. As a scripting language with modular architecture, simple syntax and rich text processing tools, Python is often used for natural language processing.

Many operating systems include Python as a standard component. It ships with most Linux distributions,[168] AmigaOS 4 (using Python 2.7), FreeBSD (as a package), NetBSD, OpenBSD (as a package) and macOS and can be used

from the command line (terminal). Many Linux distributions use installers written in Python: Ubuntu uses the Ubiquity installer, while Red Hat Linux and Fedora use the Anaconda installer. Gentoo Linux uses Python in its package management system, Portage.

Python is used extensively in the information security industry, including in exploit development.

Most of the Sugar software for the One Laptop per Child XO, now developed at Sugar Labs, is written in Python. The Raspberry Pi single-board computer project has adopted Python as its main user-programming language.

LibreOffice includes Python, and intends to replace Java with Python. Its Python Scripting Provider is a core feature since Version 4.

We have to mention some package for this project.

- Numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops, using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that

provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

The core functionality of NumPy is its "ndarray", for *n*-dimensional array, data structure. These arrays are strided views on memory.[8] In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

Such arrays can also be views into memory buffers allocated by C/C++, Cython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries. This functionality is exploited by the SciPy package, which wraps a number of such libraries (notably BLAS and LAPACK). NumPy has built-in support for memory-mapped ndarrays.

Inserting or appending entries to an array is not as trivially possible as it is with Python's lists. The np.pad(...) routine to extend arrays actually creates new arrays of the desired shape and padding values, copies the given array into the new one and returns it. NumPy's np.concatenate([a1,a2]) operation does not actually link the two arrays but returns a new one, filled with the entries from both given arrays in sequence. Reshaping the dimensionality of an array with np.reshape(...) is only possible as long as the number of elements in the array does not change. These circumstances originate from the fact that NumPy's arrays must be views on contiguous memory buffers. A replacement package called Blaze attempts to overcome this limitation.

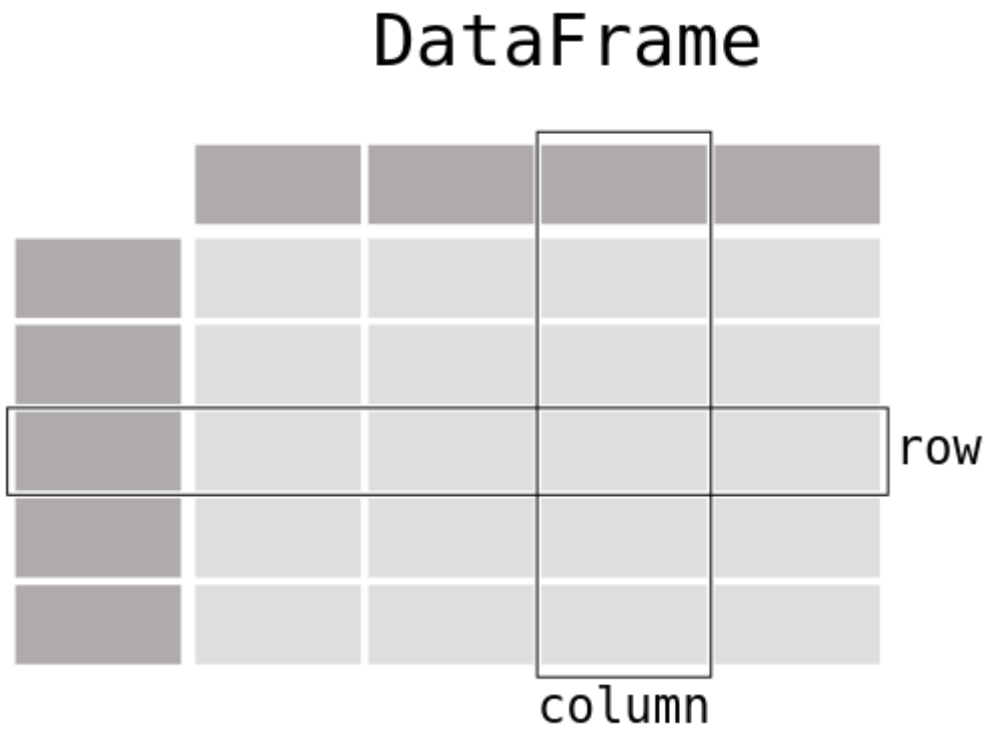
Algorithms that are not expressible as a vectorized operation will typically run slowly because they must be implemented in "pure Python", while vectorization may increase memory complexity of some operations from constant to linear, because temporary arrays must be created that are as large as the inputs. Runtime compilation of numerical code has been implemented by several groups to avoid these problems; open source solutions that interoperate with NumPy include `scipy.weave`, numexpr and Numba. Cython and Pythran are static-compiling alternatives to these.

- Pandas

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

Pandas is mainly used for machine learning in form of dataframes. Pandas allow importing data of various file formats such as csv, excel etc. Pandas allows various data manipulation operations such as groupby, join, merge, melt, concatenation as well as data cleaning features such as filling, replacing or imputing null values.

When working with tabular data, such as data stored in spreadsheets or databases, Pandas is the right tool for you. Pandas will help you to explore, clean and process your data. In Pandas, a data table is called a DataFrame.



Pandas supports the integration with many file formats or data sources out of the box (csv, excel, sql, json, parquet,…). Importing data from each of these data sources is provided by function with the prefix read_*. Similarly, the to_* methods are used to store data.

Selecting or filtering specific rows and/or columns? Filtering the data on a condition? Methods for slicing, selecting, and extracting the data you need are available in Pandas.

pandas aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language.

- DataFrame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging.
- Provides data filtration.

Pandas provides plotting your data out of the box, using the power of Matplotlib. You can pick the plot type (scatter, bar, boxplot,…) corresponding to your data.

A pandas DataFrame can be created using various inputs like −

- Lists
- dict
- Series
- Numpy ndarrays
- Another DataFrame

In the subsequent sections of this chapter, we will see how to create a DataFrame using these inputs.

The DataFrame can be created using a single list or a list of lists.

Example 1

```
import pandas as pd
data = [1,2,3,4,5]
df = pd.DataFrame(data)
print df
```

Its **output** is as follows −

```
   0
0  1
1  2
2  3
3  4
4  5
```

Example 2

```
import pandas as pd
data = [['Alex',10],['Bob',12],['Clarke',13]]
df = pd.DataFrame(data,columns=['Name','Age'])
print df
```

Its **output** is as follows −

```
      Name      Age
0     Alex      10
1     Bob       12
2     Clarke    13
```

Example 3

```
import pandas as pd
data = [['Alex',10],['Bob',12],['Clarke',13]]
df = pd.DataFrame(data,columns=['Name','Age'],dtype=float)
print df
```

Its **output** is as follows −

```
      Name     Age
0     Alex     10.0
1     Bob      12.0
2     Clarke   13.0
```

All the **ndarrays** must be of same length. If index is passed, then the length of the index should equal to the length of the arrays.

If no index is passed, then by default, index will be range(n), where **n** is the array length.

Example 1

```
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],'Age':[28,34,29,42]}
df = pd.DataFrame(data)
```

```
print df
```

Its **output** is as follows −

```
      Age      Name
0     28        Tom
1     34       Jack
2     29      Steve
3     42      Ricky
```

There is no need to loop over all rows of your data table to do calculations. Data manipulations on a column work elementwise. Adding a column to a DataFrame based on existing data in other columns is straightforward.

Basic statistics (mean, median, min, max, counts…) are easily calculable. These or custom aggregations can be applied on the entire data set, a sliding window of the data or grouped by categories. The latter is also known as the split-apply-combine approach.

Change the structure of your data table in multiple ways. You can melt() your data table from wide to long/tidy form or pivot() from long to wide format. With aggregations built-in, a pivot table is created with a single command.

Multiple tables can be concatenated both column wise as row wise and database-like join/merge operations are provided to combine multiple tables of data.

Pandas has great support for time series and has an extensive set of tools for working with dates, times, and time-indexed data.

Data sets do not only contain numerical data. Pandas provides a wide range of functions to cleaning textual data and extract useful information from it.

- scikit-learn

Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits.learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy. The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from the French Institute for Research in Computer

Science and Automation in Rocquencourt, France, took leadership of the project and made the first public release on February the 1st 2010. Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November 2012. Scikit-learn is one of the most popular machine learning libraries on GitHub.

Scikit-learn uses numpy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

The functionality that scikit-learn provides include:

- **Regression**, including Linear and Logistic Regression
- **Classification**, including K-Nearest Neighbors
- **Clustering**, including K-Means and K-Means++
- **Model selection**
- **Pre-processing**, including Min-Max Normalization

Scikit-learn integrates well with many other Python libraries, such as matplotlib and plotly for plotting, numpy for array vectorization, pandas dataframes, scipy, and many more.

Scikit-learn provides dozens of built-in machine learning algorithms and models, called estimators. Each estimator can be fitted to some data using its fit method.

Scikit-learn is largely written in Python, and uses numpy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

Scikit-learn integrates well with many other Python libraries, such as matplotlib and plotly for plotting, numpy for array vectorization, pandas dataframes, scipy, and many more.

In scikit-learn, pre-processors and transformers follow the same API as the estimator objects (they actually all inherit from the same BaseEstimator class). The transformer objects don’t have

a predict method but rather a transform method that outputs a newly transformed sample matrix Scikit-learn provides tools to automatically find the best parameter combinations (via cross-validation).

- Django

Django is a Python-based free and open-source web framework that follows the model-template-view (MVC) architectural pattern. It is maintained by the Django Software Foundation (DSF), an American independent organization established as a 501(c)(3) non-profit.

Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "pluggability" of components, less code, low coupling, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Some well known sites that use Django include PBS, Instagram, Mozilla, The Washington Times, Disqus, Bitbucket, and Nextdoor.

Despite having its own nomenclature, such as naming the callable objects generating the HTTP responses "views",[5] the core Django framework can be seen as an MVC architecture.[6] It consists of an object-relational mapper (ORM) that mediates between data models (defined as Python classes) and a relational database ("Model"), a system for processing HTTP requests with a web templating system ("View"), and a regular-expression-based URL dispatcher ("Controller").

- a lightweight and standalone web server for development and testing
- a form serialization and validation system that can translate between HTML forms and values suitable for storage in the database
- a template system that utilizes the concept of inheritance borrowed from object-oriented programming
- a caching framework that can use any of several cache methods
- support for middleware classes that can intervene at various stages of request processing and carry out custom functions
- an internal dispatcher system that allows components of an application to communicate events to each other via pre-defined signals
- an internationalization system, including translations of Django's own components into a variety of languages
- a serialization system that can produce and read XML and/or JSON representations of Django model instances

- a system for extending the capabilities of the template engine
- an interface to Python's built-in unit test framework
- Django REST framework is a powerful and flexible toolkit for building Web APIs.

Also included in the core framework are: a lightweight and standalone web server for development and testing, a form serialization and validation system that can translate between HTML forms and values suitable for storage in the database, a template system that utilizes the concept of inheritance borrowed from object-oriented programming, a caching framework that can use any of several cache methods, support for middleware classes that can intervene at various stages of request processing and carry out custom functions, an internal dispatcher system that allows components of an application to communicate events to each other via pre-defined signals, an internationalization system, including translations of Django's own components into a variety of languages, a serialization system that can produce and read XML and/or JSON representations of Django model instances, a system for extending the capabilities of the template engine, an interface to Python's built-in unit test framework, Django REST framework is a powerful and flexible toolkit for building Web APIs.

The main Django distribution also bundles a number of applications in its "contrib" package, including: an extensible authentication system, the dynamic administrative interface, tools for generating RSS and Atom syndication feeds, a "Sites" framework that allows one Django installation to run multiple websites, each with their own content and applications, tools for generating Google Sitemaps, built-in mitigation for cross-site request forgery, cross-site scripting, SQL injection, password cracking and other typical web attacks, most of them turned on by default, a framework for creating GIS applications.

HTML/CSS

HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behaviour (JavaScript). "Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as `<**img** />` and `<**input** />` directly introduce content into the page. Other tags such as `<**p**>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML .

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader),

and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.[4]

The name *cascading* comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

Material Design a modern responsive front-end CSS framework based on Material Design. Created and designed by Google, Material Design is a design language that combines the classic principles of successful design along with innovation and technology. Google's goal is to develop a system of design that allows for a unified user experience across all their products on any platform. The metaphor of material defines the relationship between space and motion. The idea is that the technology is inspired by paper and ink and is utilized to facilitate creativity and innovation. Surfaces and edges provide familiar visual cues that allow users to quickly understand the technology beyond the physical world. Elements and components such as grids, typography, color, and imagery are not only visually pleasing, but also create a sense of hierarchy, meaning, and focus. Emphasis on different actions and components create a visual guide for users. Motion allows the user to draw a parallel between what they see on the screen and in real life. By providing both feedback and familiarity, this allows the user to fully immerse him or herself into unfamiliar technology. Motion contains consistency and continuity in addition to giving users additional subconscious information about objects and transformations.

Javascript

JavaScript (often shortened to JS) is a lightweight, interpreted, object-oriented language with first-class functions, and is best known as the scripting language for Web pages, but it's used in many non-browser environments as

well. It is a prototype-based, multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles.

JavaScript runs on the client side of the web, which can be used to design / program how the web pages behave on the occurrence of an event. JavaScript is an easy to learn and also powerful scripting language, widely used for controlling web page behaviour.

Contrary to popular misconception, JavaScript is *not* "Interpreted Java". In a nutshell, JavaScript is a dynamic scripting language supporting prototype based object construction. The basic syntax is intentionally similar to both Java and C++ to reduce the number of new concepts required to learn the language. Language constructs, such as if statements, for and while loops, and switch and try ... catch blocks function the same as in these languages (or nearly so).
JavaScript can function as both a procedural and an object oriented language. Objects are created programmatically in JavaScript, by attaching methods and properties to otherwise empty objects at run time, as opposed to the syntactic class definitions common in compiled languages like C++ and Java. Once an object has been constructed it can be used as a blueprint (or prototype) for creating similar objects.

JavaScript's dynamic capabilities include runtime object construction, variable parameter lists, function variables, dynamic script creation (via eval), object introspection (via for ... in), and source code recovery (JavaScript programs can decompile function bodies back into their source text).

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). However, the language itself does not include any input/output (I/O), such as networking, storage, or graphics facilities, as the host environment (usually a web browser) provides those APIs.

JavaScript engines were originally used only in web browsers, but they are now embedded in some servers, usually via Node.js. They are also embedded in a variety of applications created with frameworks such as Electron and Cordova.

Although there are similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

Git

Git is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows. Git is free and open-source software distributed under the terms of the GNU General Public License version 2. Git stores each newly created object as a separate fill. Although individually compressed, this takes a great deal of space and is inefficient.
Git supports rapid branching and merging, and includes specific tools for visualizing and navigating a non-linear development history. In Git, a core assumption is that a change will be merged more often than it is written, as it is passed around to various reviewers. In Git, branches are very lightweight: a branch is only a reference to one commit. With its parental commits, the full branch structure can be constructed.
Git gives each developer a local copy of the full development history, and changes are copied from one such repository to another. These changes are imported as added development branches and can be merged in the same way as a locally developed branch.
Repositories can be published via Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), or a Git protocol over either a plain socket, or Secure Shell (ssh). Git also has a CVS server emulation, which enables the use of existent CVS clients and IDE plugins to access Git repositories. Subversion repositories can be used directly with git-svn.
Torvalds has described Git as being very fast and scalable, and performance tests done by Mozilla showed that it was an order of magnitude faster than some version-control systems; fetching version history from a locally stored repository can be one hundred times faster than fetching it from the remote server.
The Git history is stored in such a way that the ID of a particular version (a *commit* in Git terms) depends upon the complete development history leading up to that commit. Once it is published, it is not possible to change the old versions without it being noticed. The structure is similar to a Merkle tree, but with added data at the nodes and leaves.

This is Solved by the use of packs that store a large number of objects delta-compressed among themselves in one file (or network byte stream) called a pack file. Packs are compressed using the heuristic that files with the same name are probably similar, but do not depend on it for correctness. A

corresponding index file is created for each pack file, telling the offset of each object in the pack file. Newly created objects
(With newly added history) are still stored as single objects, and periodic repacking is needed to maintain space efficiency. The process of packing the repository can be very computationally costly. By allowing objects to exist in the repository in a loose
But quickly generated format, Git allows the costly pack operation to be deferred until later, when time matters less, e.g., the end of a work day. Git does periodic repacking automatically, but manual repacking is also possible with the git gc command. For data integrity, both the pack file and its index have an SHA-1 checksum inside, and the file name of the pack file also contains an SHA-1 checksum. To check the integrity of a repository, run the git fsck command.

Jupyter Notebook

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web

server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.
A Jupyter Notebook can be converted to a number of open standard output formats(HTML, presentationslides, LaTeX, PDF, ReStructuredText, Markdown , Python) through "Download As" in the web interface, via the nbconvert library or "jupyter nbconvert" command line interface in a shell. To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through "Download As" in the web interface, via the nbconvert library or "jupyter nbconvert" command line interface in a shell. To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

Jupyter Notebook provides a browser-based REPL built upon a number of popular open-source libraries. Jupyter Notebook can connect to many kernels to allow programming in many languages. By default Jupyter Notebook ships with the IPython kernel.

Jupyter Notebook can connect to many kernels to allow programming in many languages. By default Jupyter Notebook ships with the IPython kernel. As of the 2.3 release , there are currently 49 Jupyter-compatible kernels for many programming languages, including Python, R, Julia and Haskell.

The Notebook interface was added to IPython in the 0.12 release (December 2011), renamed to Jupyter notebook in 2015 (IPython 4.0 – Jupyter 1.0). Jupyter Notebook is similar to the notebook interface of other programs such as Maple, Mathematica, and SageMath, a computational interface style that originated with Mathematica in the 1980s. According to *The Atlantic*, Jupyter interest overtook the popularity of the Mathematica notebook interface in early 2018.

Visual Studio Code

Visual Studio Code is a source code editor that can be used with a variety of programming languages. Instead of a project system it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language, contrary to Microsoft Visual Studio which uses the proprietary .sln solution file and project-specific project files. It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many of Visual Studio Code features are not exposed through menus or the user interface, but can be accessed via the command palette. It allows users to set the code page in which the active document is saved, the newline character for Windows/Linux, and the programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language.

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js and C++. It is based on the Electron framework, which is used to develop Node.js web apps that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree

via the settings. Many Visual Studio Code features are not exposed through menus or the user interface, but can be accessed via the command palette.

Visual Studio Code can be extended via extensions, available through a central repository. This includes additions to the editor and language support. A notable feature is the ability to create extensions that add support for new languages, themes, and debuggers, perform static code analysis, and add code linters using the Language Server Protocol.

Visual Studio Code includes multiple extensions for FTP, allowing the software to be used as a free alternative for web development. Code can be synced between the editor and the server, without downloading any extra software.

Visual Studio Code allows users to set the code page in which the active document is saved, the newline character, and the programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language.

AWS/EC2

Amazon web service is a platform that offers flexible, reliable, scalable, easy-to-use and cost-effective cloud computing solutions. AWS is a comprehensive, easy to use computing platform offered Amazon. The platform is developed with a combination of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offerings.

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

Amazon Web Services (AWS) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. In aggregate, these cloud computing web services provide a set of primitive abstract technical infrastructure and distributed computing building blocks and tools. One of these services is Amazon Elastic Compute Cloud (EC2), which allows users to have at their disposal a virtual cluster of computers, available all the time, through the Internet. AWS's version of virtual computers emulates most of the

attributes of a real computer, including hardware central processing units (CPUs) and graphics processing units (GPUs) for processing; local/RAM memory; hard-disk/SSD storage; a choice of operating systems; networking; and pre-loaded application software such as web servers, databases, and customer relationship management (CRM).

The AWS technology is implemented at server farms throughout the world, and maintained by the Amazon subsidiary. Fees are based on a combination of usage (known as a "Pay-as-you-go" model), the hardware/OS/software/networking features chosen by the subscriber, required availability, redundancy, security, and service options. Subscribers can pay for a single virtual AWS computer, a dedicated physical computer, or clusters of either.

The most popular include EC2 and Amazon Simple Storage Service (Amazon S3). Most services are not exposed directly to end users, but instead offer functionality through APIs for developers to use in their applications. Amazon Web Services' offerings are accessed over HTTP, using the REST architectural style and SOAP protocol for older APIs and exclusively JSON for newer ones.

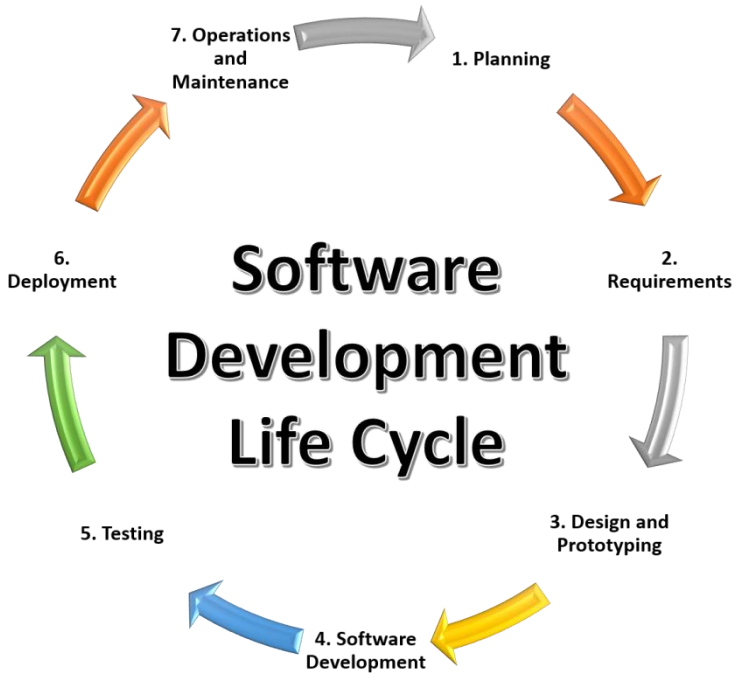
Amazon markets AWS to subscribers as a way of obtaining large scale computing capacity more quickly and cheaply than building an actual physical server farm. All services are billed based on usage, but each service measures usage in varying ways.

System details

Software development life cycle (SDLC)

The Software Development Lifecycle is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce high-quality software which meets customer expectations. The software development should be complete in the pre-defined time frame and cost.

SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC lifecycle has its own process and deliverables that feed into the next phase.



<u>Phase 1: Requirement collection and analysis or Planning:</u>

The requirement is the first stage in the SDLC process. It is conducted by the senior team members with inputs from all the stakeholders and domain experts in the industry. Planning for the quality assurance requirements and reorganization of the risks involved is also done at this stage.

This stage gives a clearer picture of the scope of the entire project and the anticipated issues, opportunities, and directives which triggered the project.

Requirements Gathering stage need teams to get detailed and precise requirements. This helps companies to finalize the necessary timeline to finish the work of that system.

Phase 2: Feasibility study:

Once the requirement analysis phase is completed the next step is to define and document software needs. This process conducted with the help of 'Software Requirement Specification' document also known as 'SRS' document. It includes everything which should be designed and developed during the project life cycle.

There are mainly five types of feasibilities checks:

- Economic: Can we complete the project within the budget or not?
- Legal: Can we handle this project as cyber law and other regulatory framework/compliances?
- Operation feasibility: Can we create operations which is expected by the client?
- Technical: Need to check whether the current computer system can support the software
- Schedule: Decide that the project can be completed within the given schedule or not.

Phase 3: Design**:**

In this third phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture.

This design phase serves as input for the next phase of the model.

There are two kinds of design documents developed in this phase:

High-Level Design (HLD)

- Brief description and name of each module
- An outline about the functionality of every module
- Interface relationship and dependencies between modules
- Database tables identified along with their key elements
- Complete architecture diagrams along with technology details

Low-Level Design(LLD)

- Functional logic of the modules
- Database tables, which include type and size
- Complete detail of the interface
- Addresses all types of dependency issues
- Listing of error messages
- Complete input and outputs for every module

Phase 4: Coding**:**

Once the system design phase is over, the next phase is coding. In this phase, developers start build the entire system by writing code using the chosen programming language. In the coding phase, tasks are divided into units or modules and assigned to the various developers. It is the longest phase of the Software Development Life Cycle process.

In this phase, Developer needs to follow certain predefined coding guidelines. They also need to use programming tools like compiler, interpreters, debugger to generate and implement the code.

Phase 5: Testing**:**

Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality of the entire system. This is done to verify that the entire application works according to the customer requirement.

During this phase, QA and testing team may find some bugs/defects which they communicate to developers. The development team fixes the bug and send back to QA for a re-test. This process continues until the software is bug-free, stable, and working according to the business needs of that system.

Phase 6: Installation/Deployment:

Once the software testing phase is over and no bugs or errors left in the system then the final deployment process starts. Based on the feedback given by the project manager, the final software is released and checked for deployment issues if any.

Phase 7: Maintenance:

Once the system is deployed, and customers start using the developed system, following 3 activities occur

- Bug fixing - bugs are reported because of some scenarios which are not tested at all
- Upgrade - Upgrading the application to the newer versions of the Software
- Enhancement - Adding some new features into the existing software

Project Layout

There is several steps to build the recommendation engine using content-based filtering. Each step is discussed in detail.

Movie Data-set & data pre-processing

&

Selecting features of the dataset

Starting with understanding of data first. There was metadata on over 5,000 movies. Data stored into a CSV (comma separated value) file. To work with data at first we need to load the data into memory. We used pandas dataframe to work with data.

Read data from csv file and visualize the data:

```
In [1]: import pandas as pd
        df = pd.read_csv("movie_dataset.csv")
        df.head()
```

Out[1]:

	index	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	...
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995	culture clash future space war space colony so...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	...
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	ocean drug abuse exotic island east india trad...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	...
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy based on novel secret agent sequel mi6	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	...
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	49026	dc comics crime fighter terrorist secret ident...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	...
4	4	260000000	Action Adventure Science Fiction	http://movies.disney.com/john-carter	49529	based on novel mars medallion space travel pri...	en	John Carter	John Carter is a war-weary, former military ca...	43.926995	...

5 rows × 24 columns

In [2]:
```
df.tail()
```

Out[2]:

	tle	overview	popularity	...	runtime	spoken_languages	status	tagline	title	vote_average	vote_count	cast	crew	directo
	chi	El Mariachi just wants to play his guitar and ...	14.269792	...	81.0	[{"iso_639_1": "es", "name": "Espa\u00f1ol"}]	Released	He didn't come looking for trouble, but troubl...	El Mariachi	6.6	238	Carlos Gallardo Jaime de Hoyos Peter Marquardt...	[{'name': 'Robert Rodriguez', 'gender': 0, 'de...	Rober Rodrigue
	ds	A newlywed couple's honeymoon is upended by th...	0.642552	...	85.0	[]	Released	A newlywed couple's honeymoon is upended by th...	Newlyweds	5.9	5	Edward Burns Kerry Bish\u00e9 Marsha Dietlein ...	[{'name': 'Edward Burns', 'gender': 2, 'depart...	Edwarc Burns
	ıd, ıd, ed	"Signed, Sealed, Delivered" introduces a dedic...	1.444476	...	120.0	[{"iso_639_1": "en", "name": "English"}]	Released	NaN	Signed, Sealed, Delivered	7.0	6	Eric Mabius Kristin Booth Crystal Lowe Geoff G...	[{'name': 'Carla Hetland', 'gender': 0, 'depar...	Scot Smitl
	ıai ıg	When ambitious New York attorney Sam is sent t...	0.857008	...	98.0	[{"iso_639_1": "en", "name": "English"}]	Released	A New Yorker in Shanghai	Shanghai Calling	5.7	7	Daniel Henney Eliza Coupe Bill Paxton Alan Ruc...	[{'name': 'Daniel Hsia', 'gender': 2, 'departm...	Danie Hsia
	ith ew	Ever since the second grade when he first saw ...	1.929883	...	90.0	[{"iso_639_1": "en", "name": "English"}]	Released	NaN	My Date with Drew	6.3	16	Drew Barrymore Brian Herzlinger Corey Feldman ...	[{'name': 'Clark Peterson', 'gender': 2, 'depa...	Brian Herzlinge

```
In [8]: df.columns

Out[8]: Index(['index', 'budget', 'genres', 'homepage', 'id', 'keywords',
               'original_language', 'original_title', 'overview', 'popularity',
               'production_companies', 'production_countries', 'release_date',
               'revenue', 'runtime', 'spoken_languages', 'status', 'tagline', 'title',
               'vote_average', 'vote_count', 'cast', 'crew', 'director'],
              dtype='object')
```

We can see that there is 24 columns in the dataframe. Each column defines a feature of the movie data-set. And also we can see that many there is many Null value, different data format (Not in proper string). So we need to preprocess the data.

- So at first need to know What is data preprocessing and features in detail.

When we talk about data, we usually think of some large datasets with huge number of rows and columns. While that is a likely scenario, it is not always the case — data could be in so many different forms: Structured Tables, Images, Audio files, Videos etc..

Machines don't understand free text, image or video data as it is, they understand 1s and 0s. So it probably won't be good enough if we put on a slideshow of all our images and expect our machine learning model to get trained just by that!

In any Machine Learning process, Data Preprocessing is that step in which the data gets transformed, or Encoded, to bring it to such a state that now the machine can easily parse it. In other words, the features of the data can now be easily interpreted by the algorithm.

- Feature in Machine Learning

A dataset can be viewed as a collection of data objects, which are often also called as a records, points, vectors, patterns, events, cases, samples, observations, or entities.

Data objects are described by a number of features, that capture the basic characteristics of an object, such as the mass of a physical object or the time at which an event occurred, etc.. Features are often called as variables, characteristics, fields, attributes, or dimensions.

A feature is an individual measurable property or characteristic of a phenomenon being observed

For instance, color, mileage and power can be considered as features of a car. There are different types of features that we can come across when we deal with data.

Features can be:

- Categorical : Features whose values are taken from a defined set of values. For instance, days in a week : {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday} is a category because its value is always taken from this set. Another example could be the Boolean set : {True, False}
- Numerical : Features whose values are continuous or integer-valued. They are represented by numbers and possess most of the properties of numbers. For instance, number of steps you walk in a day, or the speed at which you are driving your car at.

Missing values:

It is very much usual to have missing values in your dataset. It may have happened during data collection, or maybe due to some data validation rule, but regardless missing values must be taken into consideration.

- Eliminate rows with missing data :
  Simple and sometimes effective strategy. Fails if many objects have missing values. If a feature has mostly missing values, then that feature itself can also be eliminated.
- Estimate missing values :
  If only a reasonable percentage of values are missing, then we can also run simple interpolation methods to fill in those values. However, most

common method of dealing with missing values is by filling them in with the mean, median or mode value of the respective feature.

Inconsistent values :
We know that data can contain inconsistent values. Most probably we have already faced this issue at some point. For instance, the ‘Address’ field contains the ‘Phone number’. It may be due to human error or maybe the information was misread while being scanned from a handwritten form.

Duplicate values :
A dataset may include data objects which are duplicates of one another. It may happen when say the same person submits a form more than once. The term deduplication is often used to refer to the process of dealing with duplicates.

There is also too many things to describe data preprocessing. Those are not applicable in this project.

So at first select the needed features for learning.

features = ['overview', 'genres', 'keywords', 'production_companies', 'production_countries', 'spoken_languages','tagline', 'cast', 'director']

Now lets starts the pre-processing

```
In [5]: df['spoken_languages']

Out[5]: 0       [{"iso_639_1": "en", "name": "English"}, {"iso...
        1               [{"iso_639_1": "en", "name": "English"}]
        2       [{"iso_639_1": "fr", "name": "Fran\u00e7ais"},...
        3               [{"iso_639_1": "en", "name": "English"}]
        4               [{"iso_639_1": "en", "name": "English"}]
                                      ...
        4798        [{"iso_639_1": "es", "name": "Espa\u00f1ol"}]
        4799                                                  []
        4800            [{"iso_639_1": "en", "name": "English"}]
        4801            [{"iso_639_1": "en", "name": "English"}]
        4802            [{"iso_639_1": "en", "name": "English"}]
        Name: spoken_languages, Length: 4803, dtype: object
```

In [6]:
```
import json
df['spoken_languages'] = df['spoken_languages'].apply(json.loads)
def iso_639_1_name(row):
    spoken_languages = ''
    for i in row:
        try:
            spoken_languages += i['iso_639_1'] + ' '
        except:
            pass
        try:
            spoken_languages += i['name'] + ' '
        except:
            pass
    return spoken_languages
df['spoken_languages'] = df['spoken_languages'].apply(iso_639_1_name)
df['spoken_languages']
```

Out[6]:
```
0                                   en English es Español
1                                              en English
2       fr Français en English es Español it Italiano ...
3                                              en English
4                                              en English
                              ...
4798                                           es Español
4799
4800                                           en English
4801                                           en English
4802                                           en English
Name: spoken_languages, Length: 4803, dtype: object
```

Now lets start the pre-processing.

In [7]:
```
df['production_companies']
```

Out[7]:
```
0       [{"name": "Ingenious Film Partners", "id": 289...
1       [{"name": "Walt Disney Pictures", "id": 2}, {"...
2       [{"name": "Columbia Pictures", "id": 5}, {"nam...
3       [{"name": "Legendary Pictures", "id": 923}, {"...
4            [{"name": "Walt Disney Pictures", "id": 2}]
                              ...
4798            [{"name": "Columbia Pictures", "id": 5}]
4799                                                   []
4800    [{"name": "Front Street Pictures", "id": 3958}...
4801                                                   []
4802    [{"name": "rusty bear entertainment", "id": 87...
Name: production_companies, Length: 4803, dtype: object
```

In [8]:
```
df['production_companies'] = df['production_companies'].apply(json.loads)
def production_companies_name(row):
    production_companies = ''
    for i in row:
        try:
            production_companies += i['name'] + ' '
        except:
            pass
    return production_companies
df['production_companies'] = df['production_companies'].apply(production_companies_name)
```

In [9]:
```
df['production_companies']
```

Out[9]:
```
0       Ingenious Film Partners Twentieth Century Fox ...
1       Walt Disney Pictures Jerry Bruckheimer Films S...
2                          Columbia Pictures Danjaq B24
3       Legendary Pictures Warner Bros. DC Entertainme...
4                                  Walt Disney Pictures
                              ...
4798                               Columbia Pictures
4799
4800    Front Street Pictures Muse Entertainment Enter...
4801
4802           rusty bear entertainment lucky crow films
Name: production_companies, Length: 4803, dtype: object
```

In [10]:

```
df['production_countries'] = df['production_countries'].apply(json.loads)
def iso_3166_1_name(row):
    country = ''
    for i in row:
        try:
            country += i['iso_3166_1'] + ' '
        except:
            pass
        try:
            country += i['name'] + ' '
        except:
            pass
    return country
df['production_countries'] = df['production_countries'].apply(iso_3166_1_name)
df['production_countries']
```

Out[10]:

```
0       US United States of America GB United Kingdom
1                   US United States of America
2       GB United Kingdom US United States of America
3                   US United States of America
4                   US United States of America
                            ...
4798            MX Mexico US United States of America
4799
4800                US United States of America
4801             US United States of America CN China
4802                US United States of America
Name: production_countries, Length: 4803, dtype: object
```

In [11]:

```
features = [
    'overview',
    'genres',
    'keywords',
    'production_companies',
    'production_countries',
    'spoken_languages',
    'tagline',
    'cast',
    'director'
]
for feature in features:
    df[feature] = df[feature].fillna('')
df
```

Out[11]:

overview	popularity	...	runtime	spoken_languages	status	tagline	title	vote_average	vote_count	cast	crew	director
In the 22nd century, a paraplegic Marine is di...	150.437577	...	162.0	en English es Español	Released	Enter the World of Pandora.	Avatar	7.2	11800	Sam Worthington Zoe Saldana Sigourney Weaver S...	[{'name': 'Stephen E. Rivkin', 'gender': 0, 'd...	James Cameron
Captain Barbossa, long believed to be dead, ha...	139.082615	...	169.0	en English	Released	At the end of the world, the adventure begins.	Pirates of the Caribbean: At World's End	6.9	4500	Johnny Depp Orlando Bloom Keira Knightley Stel...	[{'name': 'Dariusz Wolski', 'gender': 2, 'depa...	Gore Verbinski
A cryptic message from Bond's past sends him o...	107.376788	...	148.0	fr Français en English es Español it Italiano ...	Released	A Plan No One Escapes	Spectre	6.3	4466	Daniel Craig Christoph Waltz L\u00e9a Seydoux ...	[{'name': 'Thomas Newman', 'gender': 2, 'depar...	Sam Mendes
Following the death of District Attorney Harve...	112.312950	...	165.0	en English	Released	The Legend Ends	The Dark Knight Rises	7.6	9106	Christian Bale Michael Caine Gary Oldman Anne ...	[{'name': 'Hans Zimmer', 'gender': 2, 'departm...	Christopher Nolan
John Carter is a war-weary, former military ca...	43.926995	...	132.0	en English	Released	Lost in our world, found in another.	John Carter	6.1	2124	Taylor Kitsch Lynn Collins Samantha Morton Wil...	[{'name': 'Andrew Stanton', 'gender': 2, 'depa...	Andrew Stanton
...	...	...	...	...	...	...	...	...	...	...	...	...
El Mariachi just wants to play his guitar and ...	14.269792	...	81.0	es Español	Released	He didn't come looking for trouble, but troubl...	El Mariachi	6.6	238	Carlos Gallardo Jaime de Hoyos Peter Marquardt...	[{'name': 'Robert Rodriguez', 'gender': 0, 'de...	Robert Rodriguez

So our data pre-precessing is finished.

Now We have to combine those features:

```
In [12]:
def combine_features(row):
    try:
        return row['overview'] + ' ' + row['genres'] + ' ' + row['keywords'] + ' ' + row['production_companies']
    except Exception as ex:
        print (ex, row)
df["combined_features"] = df.apply(combine_features,axis=1)
df["combined_features"]
```

```
Out[12]: 0       In the 22nd century, a paraplegic Marine is di...
         1       Captain Barbossa, long believed to be dead, ha...
         2       A cryptic message from Bond’s past sends him o...
         3       Following the death of District Attorney Harve...
         4       John Carter is a war-weary, former military ca...
                                      ...
         4798    El Mariachi just wants to play his guitar and ...
         4799    A newlywed couple's honeymoon is upended by th...
         4800    "Signed, Sealed, Delivered" introduces a dedic...
         4801    When ambitious New York attorney Sam is sent t...
         4802    Ever since the second grade when he first saw ...
         Name: combined_features, Length: 4803, dtype: object
```

Matrix of token counts of selected features

Now we have to create a matrix of number of token of selected features.

We used CountVectorizer of sklearn. Let's dive into deep to understand how it works.

In order to use textual data for predictive modeling, the text must be parsed to remove certain words – this process is called tokenization. These words need to then be encoded as integers, or floating-point values, for use as inputs in machine learning algorithms. This process is called feature extraction (or vectorization).

Scikit-learn's CountVectorizer is used to convert a collection of text documents to a vector of term/token counts. It also enables the pre-processing of text data prior to generating the vector representation. This functionality makes it a highly flexible feature representation module for text.

```
In [1]: from sklearn.feature_extraction.text import CountVectorizer

In [2]: corpus = [
            'This is the first document.',
            'This document is the second document.',
            'And this is the third one.',
            'Is this the first document?',
        ]

In [3]: vectorizer = CountVectorizer()
        X = vectorizer.fit_transform(corpus)
        print(vectorizer.get_feature_names())

        ['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third', 'this']

In [4]: print(X.toarray())

        [[0 1 1 1 0 0 1 0 1]
         [0 2 0 1 0 1 1 0 1]
         [1 0 0 1 1 0 1 1 1]
         [0 1 1 1 0 0 1 0 1]]
```

Now apply it into our data-set.

```
In [15]: from sklearn.feature_extraction.text import CountVectorizer
         cv = CountVectorizer()
         count_matrix = cv.fit_transform(df["combined_features"])
         count_matrix.toarray()

Out[15]: array([[0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0],
                ...,
                [0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0]])

In [17]: print(count_matrix)

  (0, 14655)    1
  (0, 29450)    3
  (0, 250)      1
  (0, 5184)     2
  (0, 21883)    1
  (0, 18488)    1
  (0, 15237)    1
  (0, 8477)     1
  (0, 29761)    1
  (0, 19833)    1
  (0, 21810)    2
  (0, 21267)    1
  (0, 31044)    1
  (0, 19585)    1
  (0, 4486)     1
  (0, 2949)     1
  (0, 29889)    1
  (0, 3253)     1
  (0, 11271)    1
  (0, 21386)    1
  (0, 1388)     1
  (0, 23434)    1
  (0, 1347)     1
```

Similarly Matrix

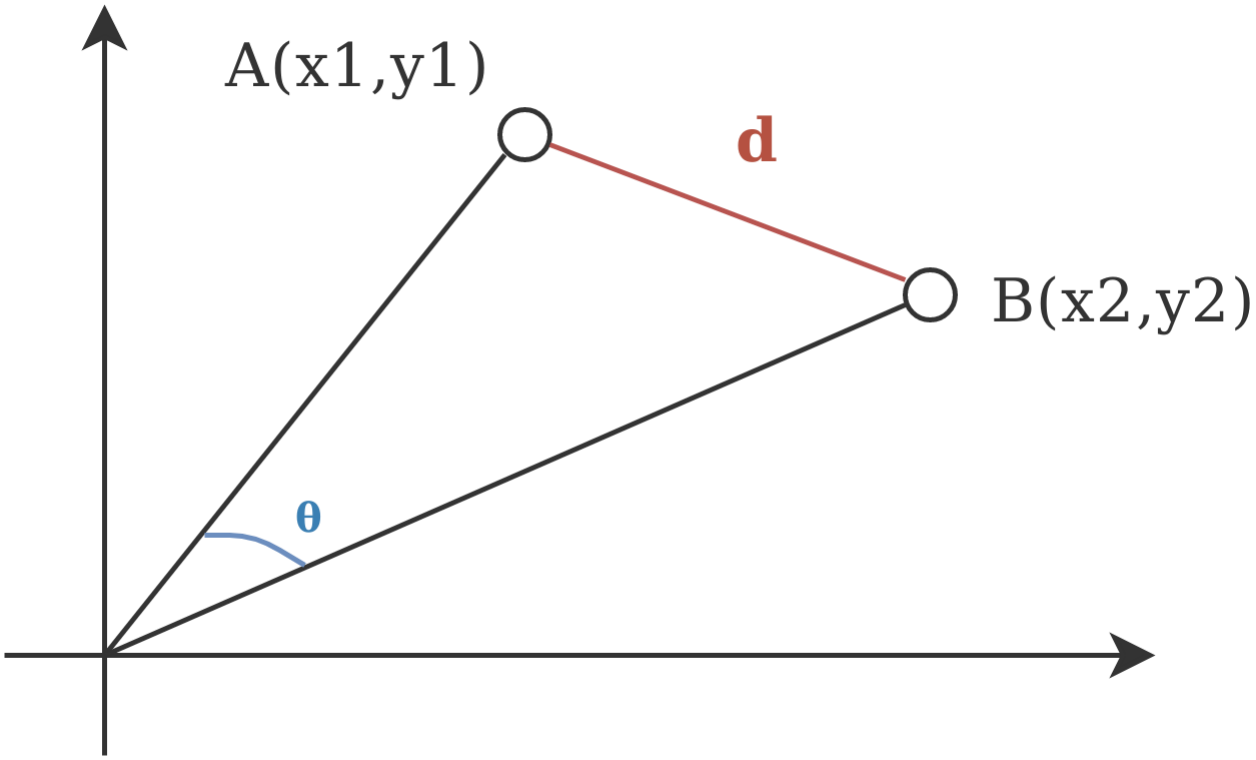
Now we have to find the similarity matrix which is collection of similarity score for each item.

Let's understand the similarity score and similarity matrix.

Example:

Text A: London Paris London
Text B: Paris Paris London

(London, Paris)
A: (2, 1)
B: (1, 2)
X-Axis = London, Y-Axis = Paris



A: (x1, y1) = (2, 1)
B: (x2, y2) = (1, 2)
D = Euclidean Distance and theta is Angular Distance

Now for 2 vector A and B

$$\cos\theta = \frac{A \cdot B}{|A||B|}$$

The value of cos(theta) lies between 0 and 1 for positive direction of A vector and B vector.

Now how much document is similar it depends on their distance of their count vector.

Now let's look at the cosine similarity:

Cosine similarity is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar documents are far

apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.

Cosine similarity is a metric used to determine how similar the documents are irrespective of their size.

Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. In this context, the two vectors we are talking about are arrays containing the word counts of two documents.

As a similarity metric, how does cosine similarity differ from the number of common words?

When plotted on a multi-dimensional space, where each dimension corresponds to a word in the document, the cosine similarity captures the orientation (the angle) of the documents and not the magnitude. If you want the magnitude, compute the Euclidean distance instead.

The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance because of the size (like, the word ‘cricket’ appeared 50 times in one document and 10 times in another) they could still have a smaller angle between them. Smaller the angle, higher the similarity.

Cosine similarity is generally used as a metric for measuring distance when the magnitude of the vectors does not matter. This happens for example when working with text data represented by word counts. We could assume that when a word (e.g. science) occurs more frequent in document 1 than it does in document 2, that document 1 is more related to the topic of science. However, it could also be the case that we are working with documents of uneven lengths (Wikipedia articles for example). Then, science probably occurred more in document 1 just because it was way longer than document 2. Cosine similarity corrects for this.

Text data is the most typical example for when to use this metric. However, you might also want to apply cosine similarity for other cases where some properties of the instances make so that the weights might be larger without meaning anything different. Sensor values that were captured in various lengths (in time) between instances could be such an example.

So we used cosine similarity to calculate similarity score in our engine. Now lets another example how to calculate cosine similarity with 3 text document.

```
In [1]: doc_trump = "Mr. Trump became president after winning the political election. Though he lost the support of some

        doc_election = "President Trump says Putin had no political interference is the election outcome. He says it was

        doc_putin = "Post elections, Vladimir Putin became President of Russia. President Putin had served as the Prime

        documents = [doc_trump, doc_election, doc_putin]

In [2]: from sklearn.feature_extraction.text import CountVectorizer
        import pandas as pd

In [3]: count_vectorizer = CountVectorizer()
        sparse_matrix = count_vectorizer.fit_transform(documents)

In [4]: doc_term_matrix = sparse_matrix.todense()
        df = pd.DataFrame(doc_term_matrix,
                          columns=count_vectorizer.get_feature_names(),
                          index=['doc_trump', 'doc_election', 'doc_putin'])
        df
```

Out[4]:

	after	as	became	by	career	claimed	do	earlier	election	elections	...	the	though	to	trump	vladimir	was	who	winning	witchl
doc_trump	1	0	1	0	0	0	0	0	1	0	...	2	1	0	2	0	0	0	1	
doc_election	0	0	0	1	0	1	1	0	2	0	...	2	0	1	1	0	1	1	0	
doc_putin	0	1	1	0	1	0	0	1	0	1	...	1	0	0	0	1	0	0	0	

3 rows × 48 columns

```
In [5]: from sklearn.metrics.pairwise import cosine_similarity
        cosine_similarity(df, df)
```

```
Out[5]: array([[1.        , 0.51480485, 0.38890873],
               [0.51480485, 1.        , 0.38829014],
               [0.38890873, 0.38829014, 1.        ]])
```

We can see the score according to diagonal is high i.e. 1.

Now apply the cosine similarity in our movie data-set

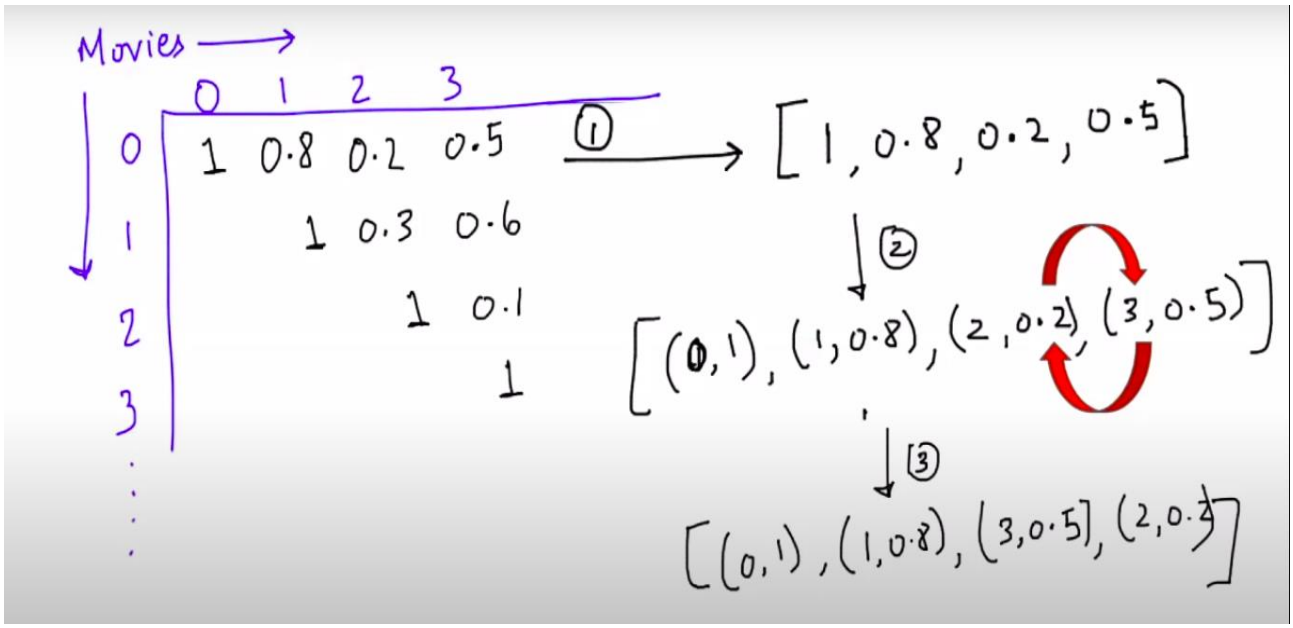
```
In [18]: from sklearn.metrics.pairwise import cosine_similarity
         similarity_score = cosine_similarity(count_matrix)
         similarity_score
```

```
Out[18]: array([[1.        , 0.37008508, 0.32716515, ..., 0.33994485, 0.26363636,
                 0.18397324],
                [0.37008508, 1.        , 0.38095973, ..., 0.43546484, 0.34426519,
                 0.23031121],
                [0.32716515, 0.38095973, 1.        , ..., 0.30429031, 0.27117967,
                 0.20116856],
                ...,
                [0.33994485, 0.43546484, 0.30429031, ..., 1.        , 0.34497574,
                 0.2247765 ],
                [0.26363636, 0.34426519, 0.27117967, ..., 0.34497574, 1.        ,
                 0.2508726 ],
                [0.18397324, 0.23031121, 0.20116856, ..., 0.2247765 , 0.2508726 ,
                 1.        ]])
```

Recommendation

Now let's recommend some movie using our similarity score matrix. So we used 2 method to convert index to title and title to index.

Here we sort a selected movie according to similarity score matrix.



After that we are showing top 10 similar movies.

```
In [19]:
def get_title_from_index(index):
    return df[df.index == index]["title"].values[0]
def get_index_from_title(title):
    return df[df.title == title]["index"].values[0]
```

```
In [ ]:
movie_rec = "Avatar"
```

```
In [20]:
movie_index = get_index_from_title(movie_rec)
similar_movies = list(enumerate(similarity_score[movie_index]))
sorted_similar_movies = sorted(similar_movies, key=lambda x: x[1], reverse=True)
i=0
for element in sorted_similar_movies[ : 10]:
    print(get_title_from_index(element[0]))
```

```
Avatar
Alien
Aliens vs Predator: Requiem
Dragonball Evolution
Prometheus
Aliens
Fantastic 4: Rise of the Silver Surfer
X-Men: Days of Future Past
Independence Day: Resurgence
Sunshine
```

Application with code sample

For application we saved our model into pickle file and we reuse into Django view for each request.

Here are code sample:

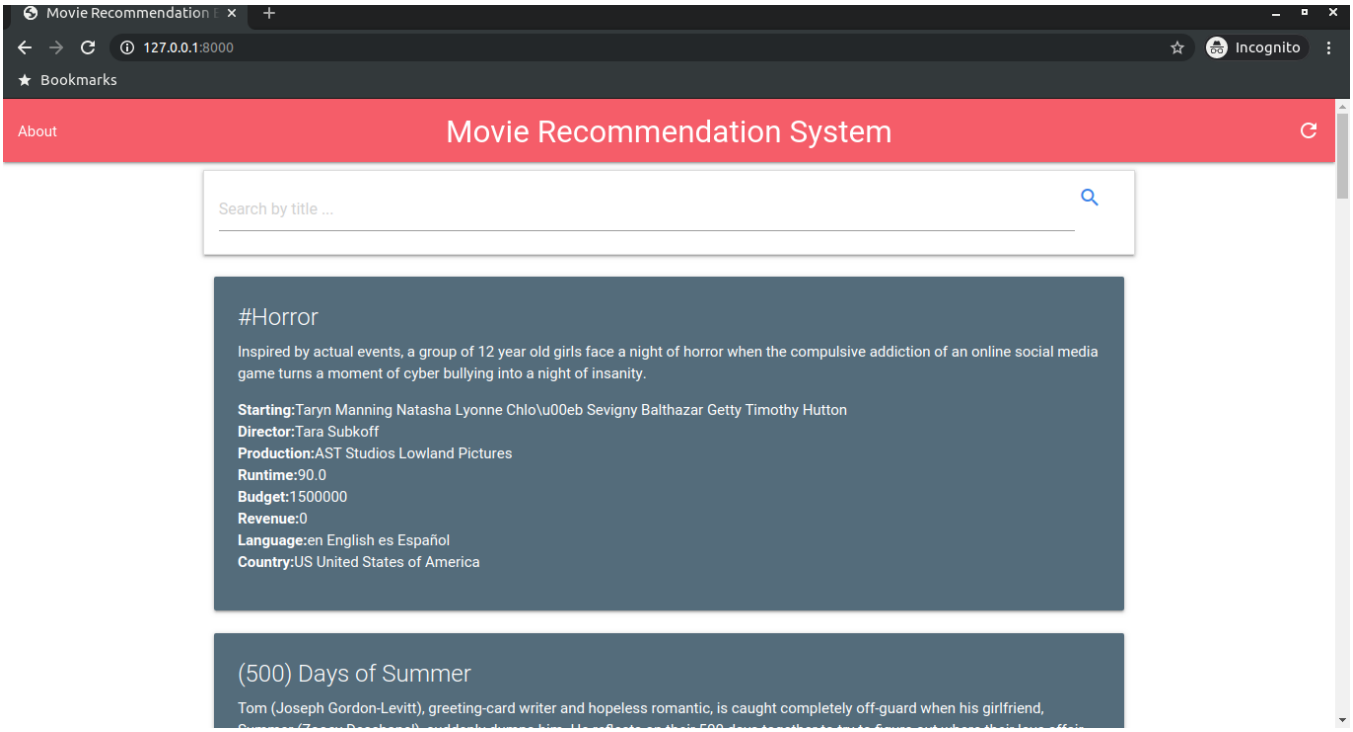
```
import pandas as pd
df = pd.read_csv("movie_dataset.csv")

import json
df['spoken_languages'] = df['spoken_languages'].apply(json.loads)
df['production_companies'] = df['production_companies'].apply(json.loads)
df['production_countries'] = df['production_countries'].apply(json.loads)
```

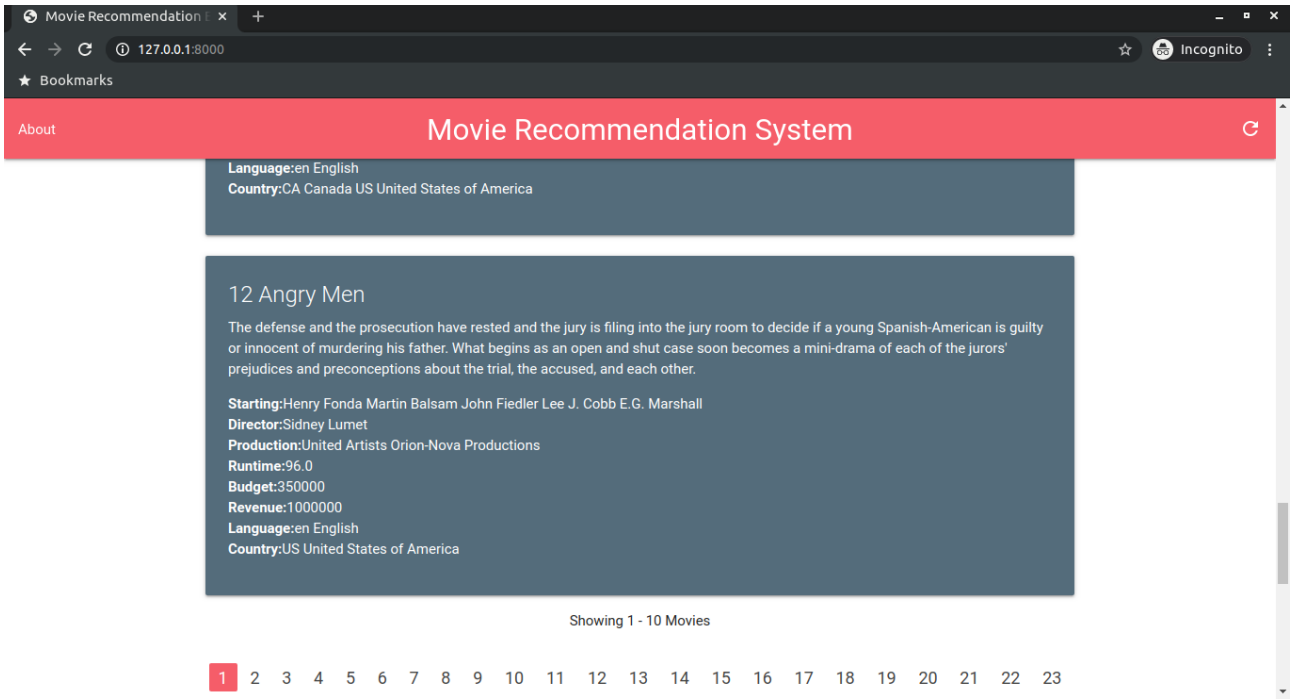
```
def iso_639_1_name(row):
spoken_languages = ''
for i in row:
try:
spoken_languages += i['iso_639_1'] + ''
except:
pass
try:
spoken_languages += i['name'] + ''
except:
pass
return spoken_languages
def production_companies_name(row):
production_companies = ''
for i in row:
try:
production_companies += i['name'] + ''
except:
pass
return production_companies
def iso_3166_1_name(row):
country = ''
for i in row:
try:
country += i['iso_3166_1'] + ''
except:
pass
try:
```

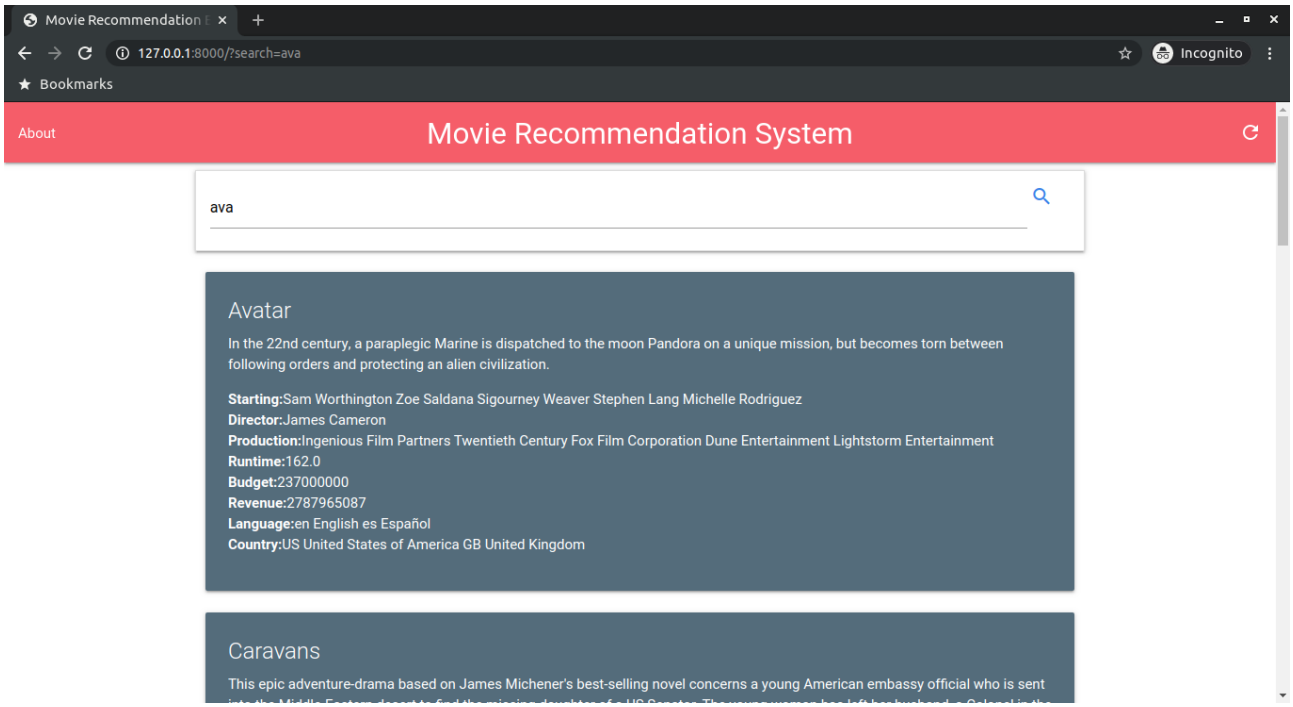
```
country += i['name'] + ''
except:
pass
return country

df['spoken_languages'] = df['spoken_languages'].apply(iso_639_1_name)
df['production_companies'] = df['production_companies'].apply(production_companies_name)
df['production_countries'] = df['production_countries'].apply(iso_3166_1_name)

features = ['overview', 'genres', 'keywords', 'production_companies', 'production_countries', 'spoken_languages','tagline', 'cast', 'director']
for feature in features:
df[feature] = df[feature].fillna('')
def combine_features(row):
try:
return row['overview'] + '' + row['genres'] + '' + row['keywords'] + '' + row['production_companies'] + '' + row['production_countries'] + '' + row['spoken_languages'] + '' + row['tagline'] + '' + row['cast'] + '' + row['director']
except Exception as ex:
print (ex, row)
df["combined_features"] = df.apply(combine_features,axis=1)

from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
count_matrix = cv.fit_transform(df["combined_features"])

from sklearn.metrics.pairwise import cosine_similarity
```

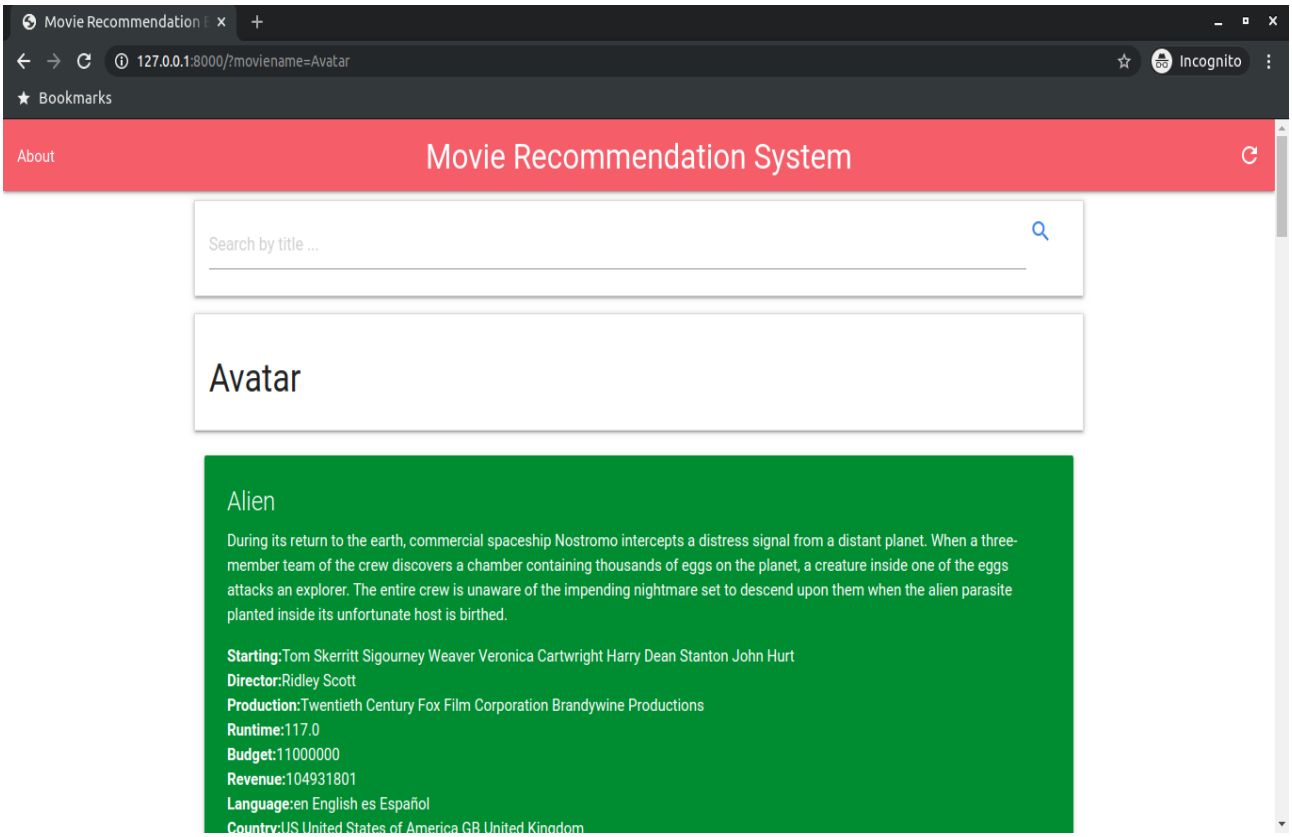
```
similarity_score = cosine_similarity(count_matrix)

df.to_pickle("movie_data_frame.pkl")

import numpy
numpy.save("similarity_score", similarity_score)
```

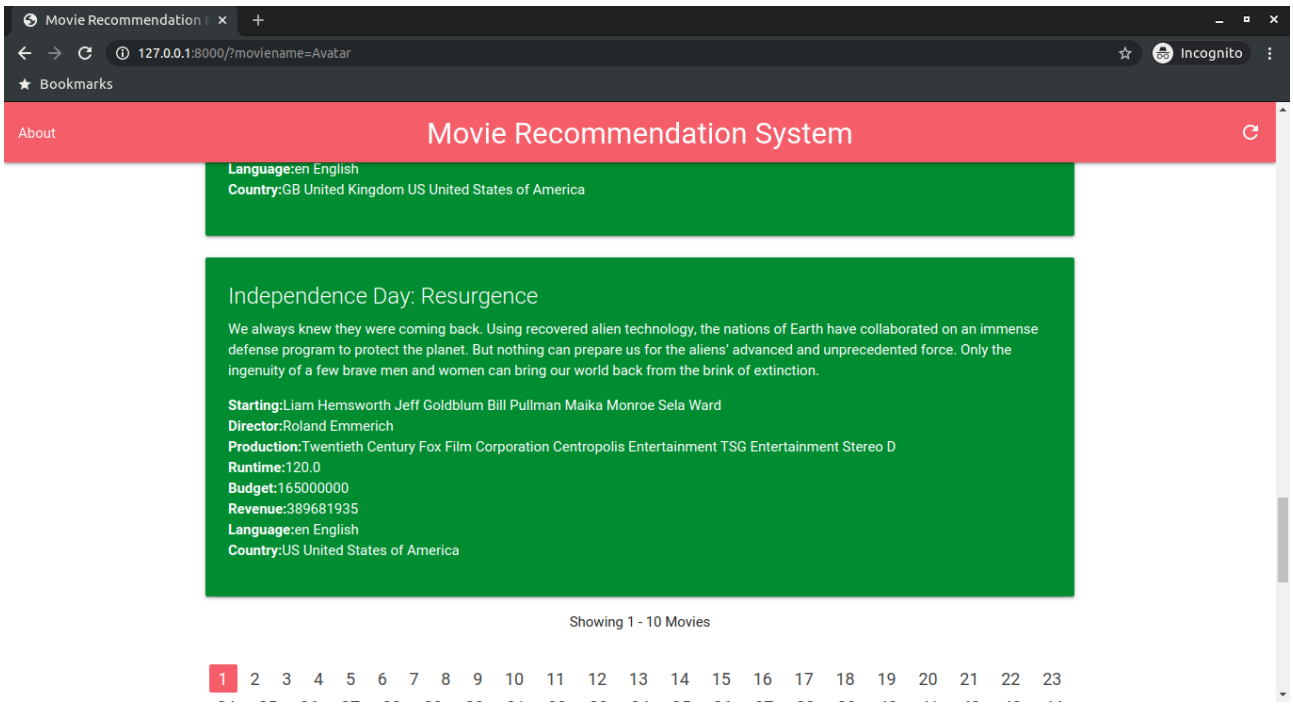
Now let's see some screenshot of our application "Movie Recommendation System"

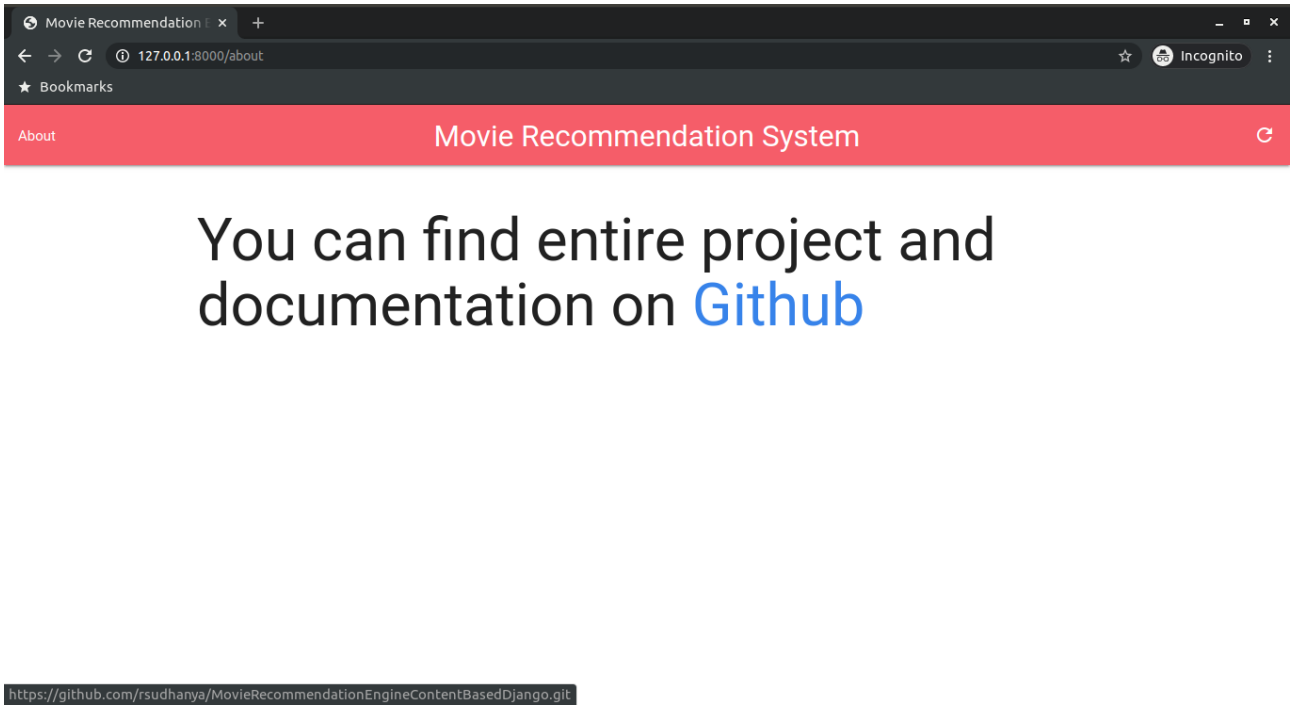



Movie Recommendation
127.0.0.1:8000
Incognito
Bookmarks
About
Movie Recommendation System
Language:en English
Country:CA Canada US United States of America
12 Angry Men
The defense and the prosecution have rested and the jury is filing into the jury room to decide if a young Spanish-American is guilty or innocent of murdering his father. What begins as an open and shut case soon becomes a mini-drama of each of the jurors' prejudices and preconceptions about the trial, the accused, and each other.
Starting:Henry Fonda Martin Balsam John Fiedler Lee J. Cobb E.G. Marshall
Director:Sidney Lumet
Production:United Artists Orion-Nova Productions
Runtime:96.0
Budget:350000
Revenue:1000000
Language:en English
Country:US United States of America
Showing 1 - 10 Movies
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23


Movie Recommendation
127.0.0.1:8000/?search=ava
Incognito
Bookmarks
About
Movie Recommendation System
ava
Avatar
In the 22nd century, a paraplegic Marine is dispatched to the moon Pandora on a unique mission, but becomes torn between following orders and protecting an alien civilization.
Starting:Sam Worthington Zoe Saldana Sigourney Weaver Stephen Lang Michelle Rodriguez
Director:James Cameron
Production:Ingenious Film Partners Twentieth Century Fox Film Corporation Dune Entertainment Lightstorm Entertainment
Runtime:162.0
Budget:237000000
Revenue:2787965087
Language:en English es Español
Country:US United States of America GB United Kingdom
Caravans
This epic adventure-drama based on James Michener's best-selling novel concerns a young American embassy official who is sent

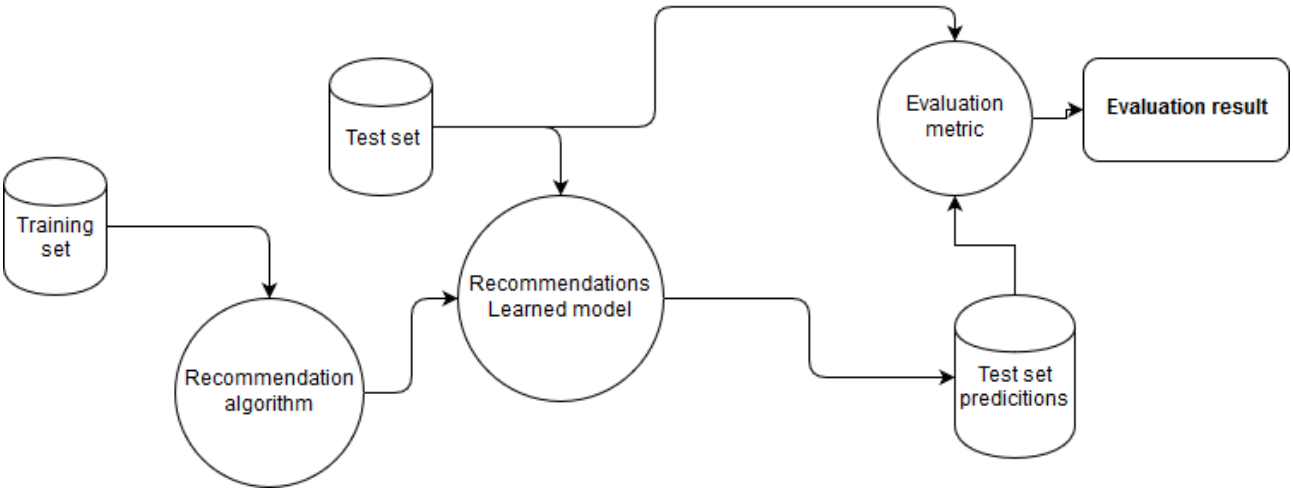

Movie Recommendation
127.0.0.1:8000/?moviename=Avatar
Incognito
Bookmarks
About
Movie Recommendation System
Search by title ...
Avatar
Alien
During its return to the earth, commercial spaceship Nostromo intercepts a distress signal from a distant planet. When a three-member team of the crew discovers a chamber containing thousands of eggs on the planet, a creature inside one of the eggs attacks an explorer. The entire crew is unaware of the impending nightmare set to descend upon them when the alien parasite planted inside its unfortunate host is birthed.
Starting:Tom Skerritt Sigourney Weaver Veronica Cartwright Harry Dean Stanton John Hurt
Director:Ridley Scott
Production:Twentieth Century Fox Film Corporation Brandywine Productions
Runtime:117.0
Budget:11000000
Revenue:104931801
Language:en English es Español
Country:US United States of America GB United Kingdom


Movie Recommendation
127.0.0.1:8000/?moviename=Avatar
Incognito
Bookmarks
About
Movie Recommendation System
Language:en English
Country:GB United Kingdom US United States of America
Independence Day: Resurgence
We always knew they were coming back. Using recovered alien technology, the nations of Earth have collaborated on an immense defense program to protect the planet. But nothing can prepare us for the aliens' advanced and unprecedented force. Only the ingenuity of a few brave men and women can bring our world back from the brink of extinction.
Starting:Liam Hemsworth Jeff Goldblum Bill Pullman Maika Monroe Sela Ward
Director:Roland Emmerich
Production:Twentieth Century Fox Film Corporation Centropolis Entertainment TSG Entertainment Stereo D
Runtime:120.0
Budget:165000000
Revenue:389681935
Language:en English
Country:US United States of America
Showing 1 - 10 Movies
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23


Movie Recommendation
127.0.0.1:8000/about
Incognito
Bookmarks
About
Movie Recommendation System
You can find entire project and documentation on Github
https://github.com/rsudhanya/MovieRecommendationEngineContentBasedDjango.git

Evaluation

Evaluation is a technique of metrics have been “Translated” to help us evaluate recommendation systems. To understand how these metrics work, we need to first understand the workflow of recommendation systems and then how to evaluate them.



The diagram below explains a workflow of recommendation systems. First a training set is fed to a recommendation algorithm which produces a recommendation model that can be used to generate new predictions. To evaluate the model a held out test set is fed to the learned model where predictions are generated for each user-item pair. Predictions with known labels (true value) are then used as an input to the evaluation algorithm to produce evaluation results.

Different type of evaluation metrics:

Once you have built a classification model, you need evaluate how good the predictions made by that model are. So, how do you define 'good' predictions?

There are some performance metrics which help us improve our models. Let us explore the differences between them for a binary classification problem:

Consider the following Confusion Matrix for a classification problem which predicts whether a patient has Cancer or not for 100 patients:

		Predicted	
Actual		Cancer = Yes	Cancer = No
	Cancer = Yes	True Positive (TP) = 25	False Negative (FN) = 5
	Cancer = No	False Positive(FP) = 5	True Negative(TN) = 65

Now, the following are the fundamental metrics for the above data:

Precision: It is implied as the measure of the correctly identified positive cases from all the predicted positive cases. Thus, it is useful when the costs of False Positives is high.

$$\text{Precision} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Positive})} = \frac{25}{(25+5)} = \frac{25}{30} = 0.83$$

The evaluation of Recommender Systems (RS) has been an explicit object of study in the field since its earliest days, and is still an area of active research, where open questions remain. The dominant evaluation methodologies in off-line experimentation have been traditionally error-based. There is however an increasing realization that the quality of the ranking of recommended items can be more important in practice (in terms of the effective utility for users) than the accuracy in predicting specific preference values. As a result,

precision-oriented metrics are being increasingly often considered in the field. Yet there is considerable divergence in the way these metrics are being applied by different authors, as a consequence of which, the results reported in different studies are difficult to put in context and compare. In the typical formulation of the recommendation problem, user interests for items are represented as numeric ratings, some of which are known. Based on this, the task of a recommendation algorithm consists of predicting unknown ratings based on the known ones and, in some methods, some additional available information about items and users. With this formulation, the accuracy of recommendations has been evaluated by measuring the error between predicted and known ratings, by metrics such as.

Recall: It is the measure of the correctly identified positive cases from all the actual positive cases. It is important when the cost of False Negatives is high.

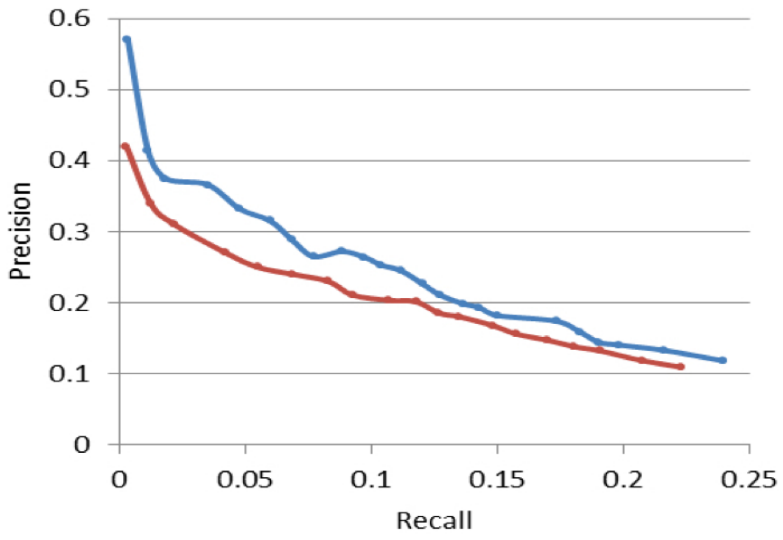
$$\text{Recall} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Negative})} = \frac{25}{(25+5)} = \frac{25}{30} = 0.83$$

Many evaluation metrics have been used to rank recommendation algorithms, some measuring similar features, but some measuring drastically different quantities. For example, methods such as the Root of the Mean Square Error (RMSE) measure the distance between predicted preferences and true preferences over items, while the Recall method computes the portion of favored items that were suggested. Clearly, it is unlikely that a single algorithm would outperform all others over all possible methods. Therefore, we should expect different metrics to provide different rankings of algorithms. As such, selecting the proper evaluation metric to use has a crucial influence on the selection of the recommender system algorithm that will be selected for deployment. This survey reviews existing evaluation metrics, suggesting an approach for deciding which evaluation metric is most appropriate for a given application. We categorize previously suggested recommender systems into three major groups, each corresponding to a different task. The first obvious task is to recommend a set of good (interesting, useful) items to the user. In this task it is assumed that all good items are interchangeable. A second, less discussed, although highly important task is utility optimization. For example, many e-commerce websites use a recommender system,

hoping to increase their revenues. In this case, the task is to present a set of recommendations that will optimize the retailer revenue. Finally, a very common task is the prediction of user opinion (e.g., rating) over a set of items. While this may not be an explicit act of recommendation, much research in recommender systems focuses on this task, and so we address it here. For each such task we review a family of common evaluation metrics that measure the performance of algorithms on that task. We discuss the properties of each such metric, and why it is most appropriate for a given task. In some cases, applying incorrect evaluation metrics may result in selecting an inappropriate algorithm. We demonstrate this by experimenting with a wide collection of data sets, comparing a number of algorithms using various evaluation metrics, showing that the metrics rank the algorithms

differently. We also discuss the proper design of an offline experiment, explaining how the data should be split, which measurements should be taken, how to determine if differences in performance are statistically significant, and so forth. We also describe a few common pitfalls that may produce results that are not statistically sound. When users add movies to their queues in Netflix, the system presents a list of 10 movies that they may like. However, when users choose to see recommendations (by clicking “movies that you will love”) the system presents all the possible recommendations. If there are too many recommended movies to fit a single page, the system allows the user to move to the next page of recommendations. These two different usage scenarios illustrate a fundamental difference between recommendation applications—in the first, the system is allowed to show a small, fixed number of recommendations. In the second, the system provides as many recommendations as it can. Even though the two cases match a single task—the “recommend good items” task—there are several important distinctions that arise. It is important to evaluate the two cases properly. When the system is required to present a list with a small, fixed size, that is known a priori, methods that present curves (precision-recall), or methods that evaluate the entire list (half-lifeutility score), become less appropriate. For example, a system may get a relatively high half-life utility score, only due to items that fall outside the fixed list, while another system that selects all the items in the list correctly, and uninteresting

items elsewhere, might get a lower score. Precisionrecall curves are typically used to help us select the proper list length, where the precision and recall reach desirable values. Another important difference, is that for a small list, the order of items in the list is less important, as we can assume that the user looks at all the items in the list. Moreover, may of these lists are presented in a horizontal direction, which also reduces the importance of properly ordering the items. In these cases, therefore, a more appropriate way to evaluate the recommendation system should focus on the first N movies only. In the “recommend good items” task this can be done, for example, by measuring the precision at N—the number of items that are interesting out of the recommended N items. In the “optimize utility” task, we can do so by measuring the aggregated utility (e.g., sum of utility) of the items that are indeed interesting within the N recommendations. A final case is when we have unlimited recommendation lists in the “recommend good items” scenario, and we wish to evaluate the entire list. In this case, one can use the half-life utility score with a binary utility of 1 when the (hidden) item was indeed selected by the user, and 0 otherwise. In that case, the half-life utility score prefers a recommender system that places interesting items closer to the head of the list, but provides an evaluation for the entire list in a single score.



Precision-Recall curve comparing the recommendation results

Accuracy**:** One of the more obvious metrics, it is the measure of all the correctly identified cases. It is most used when all the classes are equally important.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{(\text{True Positive} + \text{False Positive} + \text{True Negative} + \text{False Negative})}$$

$$= \frac{25 + 65}{(25 + 5 + 65 + 5)} = \frac{90}{100} = 0.90$$

Now for our above example, suppose that there only 30 patients who actually have cancer. What if our model identifies 25 of those as having cancer?

The accuracy in this case is = 90% which is a high enough number for the model to be considered as ‘accurate’. However, there are 5 patients who actually have cancer and the model predicted that they don’t have it.

Obviously, this is too high a cost. Our model should try to minimize these False Negatives.

When we evaluate the quality of recommender systems (RS), most approaches only focus on the predictive accuracy of these systems. Recent works suggest that beyond accuracy there is a variety of other metrics that should be considered when evaluating a RS. In this paper we focus on two crucial metrics in RS evaluation: coverage and serendipity. Based on a literature review, we first discuss both measurement methods as well as the trade-off between good coverage and serendipity. We then analyze the role of coverage and serendipity as indicators of recommendation quality, present novel ways of how they can be measured and discuss how to interpret the obtained measurements. Overall, we argue that our new ways of measuring these concepts reflect the quality impression perceived by the user in a better way than previous metrics thus leading to enhanced user satisfaction.

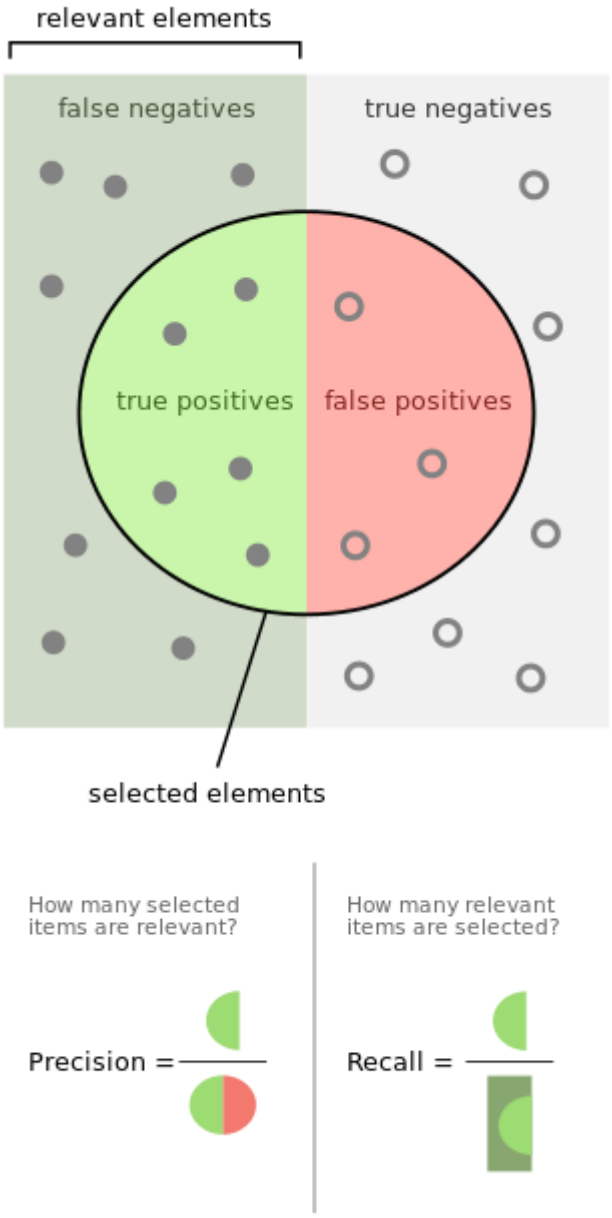
F1-score:This is the harmonic mean of Precision and Recall and gives a better measure of the incorrectly classified cases than the Accuracy Metric.

$$\text{F1-score} = \left(\frac{\text{Recall}^{-1} + \text{Precision}^{-1}}{2}\right)^{-1} = 2 * \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

We use the Harmonic Mean since it penalizes the extreme values.

In statistical analysis of binaryclassification, the F1 score (also F-score or F-measure) is a measure of a test's accuracy. It considers both the precision *p* and the recall *r* of the test to compute the score: *p* is the number of correct positive results divided by the number of all positive results returned by the classifier, and *r* is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive).

The F1 score is the harmonic mean of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall). The F1 score is also known as the Sørensen–Dice coefficient or Dice similarity coefficient (DSC).



F1 score takes values between 0 and 1 just like recall and precision. It becomes 0 when one of precision or recall is 0 and 1 when both precision and recall are 1.

To summarise the differences between the F1-score and the accuracy,

- Accuracy can be used when the class distribution is similar while F1-score is a better metric when there are imbalanced classes as in the above case.

• In most real-life classification problems, imbalanced class distribution exists and thus F1-score is a better metric to evaluate our model on.

For these cases, we use the F1-score.

Now We need to know ground truth data-set.

Ground truth is a term used in various fields to refer to information provided by direct observation (i.e. empirical evidence) as opposed to information provided by inference. "Ground truth" may be seen as a conceptual term relative to the knowledge of the truth concerning a specific question. It is the ideal expected result. This is used in statistical models to prove or disprove research hypotheses. The term "ground truthing" refers to the process of gathering the proper objective (provable) data for this test. Compare with gold standard. For example, suppose we are testing a stereo vision system to see how well it can estimate 3D positions. The "ground truth" might be the positions given by a laser rangefinder which is known to be much more accurate than the camera system.

Bayesian spam filtering is a common example of supervised learning. In this system, the algorithm is manually taught the differences between spam and non-spam. This depends on the *ground truth* of the messages used to train the algorithm – inaccuracies in the ground truth will correlate to inaccuracies in the resulting spam/non-spam verdicts.

The ground truth is what you measured for your target variable for the training and testing examples.

Nearly all the time you can safely treat this the same as the label.

In some cases it is not precisely the same as the label. For instance if you augment your data set, there is a subtle difference between the ground truth(your actual measurements) and how the augmented examples relate to the labels you have assigned. However, this distinction is not usually a problem.

Ground truth can be wrong. It is a measurement, and there can be errors in it. In some ML scenarios it can also be a subjective measurement where it is difficult define an underlying objective truth - e.g. expert opinion or analysis,

which you are hoping to automate. Any ML model you train will be limited by the quality of the ground truth used to train and test it, and that is part of the explanation on the Wikipedia quote. It is also why published articles about ML should include full descriptions of how the data was collected.

We are showing few movies with F1 score from ground truth data-set of approximately 5000 movies for choosing Movie “Avatar”.

	Movie Title	Score
0	Avatar	1.000000
1	Alien	0.745468
2	Aliens vs Predator: Requiem	0.721324
3	Dragonball Evolution	0.704645
4	Prometheus	0.701645
5	Aliens	0.700045
6	Fantastic 4: Rise of the Silver Surfer	0.700000
7	X-Men: Days of Future Past	0.692344
8	Independence Day: Resurgence	0.691454
9	Sunshine	0.690000

Limitations

•Limited content analysis: If the content doesn't contain enough information to discriminate the items precisely, the recommendation itself risks being imprecise.

•Over-specialization: Content-based filtering provides a limited degree of novelty since it has to match up the features of a user's profile with available items. In the case of item-based filtering, only item profiles are created and users are suggested items similar to what they rate or search for, instead of their past history. A perfect content-based filtering system may suggest nothing unexpected or surprising.

•Since the feature representation of the items are hand-engineered to some extent, this technique requires a lot of domain knowledge. Therefore, the model can only be as good as the hand-engineered features.

•The model can only make recommendations based on existing interests of the user. In other words, the model has limited ability to expand on the users' existing interests.

•Cold start is a potential problem in computer-based information systems which involve a degree of automated data modelling. Specifically, it concerns the issue that the system cannot draw any inferences for users or items about which it has not yet gathered sufficient information. The cold start problem is a well known and well researched problem for recommender systems. Recommender systems form a specific type of information filtering (IF) technique that attempts to present information items (e-commerce, films, music, books, news, images, web pages) that are likely of interest to the user. Typically, a recommender system compares the user's profile to some reference characteristics. These characteristics may be related to item characteristics (content-based filtering) or the user's social environment and past behavior (collaborative filtering). Depending on the system, the user can be associated to various kinds of interactions: ratings, bookmarks, purchases, likes, number of page visits etc.

•If a new item is added in the data-set the the model needs retraining which takes to much time.

Conclusion

The recommendation engine implemented in this project aims at providing movie recommendation based on the many features of the movies. It recommends similar movie of selected movie to a user. Recommendation systems are widely used in today's era of internet searching for reliable and relevant information. While simple recommendation systems recommend users based on a few parameters, complex ones take many parameters into consideration. By implementing machine learning in recommendation systems, Several multinational companies have been exploiting the potential of recommendation system to lure customers into using their products. Such as Google, Netflix, Amazon etc.

Future scope

- We can implement a scope to add new movie in our data-set.
- The engine only using content-based filtering. Collaborative filtering can be added for hybrid recomendation.
- The different algorithm can be used to get better and faster recomendation model.
- Also we can improve the user experience of the application.
- A movie watching platform can be added.

Bibliography

Websites:

- https://www.google.co.in/
- https://docs.djangoproject.com/en/2.2/
- www.stackoverflow.com
- https://www.youtube.com/
- https://materializecss.com/
- https://www.wikipedia.org/
- https://medium.com/
- https://www.coursera.org/
- https://www.edureka.co/
- https://scikit-learn.org/
- https://pandas.pydata.org/
- https://numpy.org/
- https://docs.aws.amazon.com/ec2/index.html
- https://developers.google.com/
- https://towardsdatascience.com/