People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

Ferhat ABBAS Sétif 1 University

Faculty of Sciences

# MASTER THESIS

Domain: Maths and Computer Science

Department: Computer Science

Speciality: Data Engineering and Web Technologies

**A Deep Learning Model For A Hybrid Recommender System Of Online Resources**

**Presented by:**

Mr. Abdelkader SEBIHI

**Supervised by:**

Dr.MEDIANI Chahrazed

2020/2021

# A Deep Learning Model For A Hybrid Recommender System Of Online Resources

Abderlkader SEBIHI

September 2021

## Abstract

Recommender system is a tool for interacting with large scale and complex information space. They provide personalized views and prioritized items likely to be of interest to the user. The field was named in 1995 and has been greatly developed in the various problems solved, the technology used and its practical application. Recommender systems research has incorporated a wide variety of artificial intelligence techniques including machine learning, data mining, user modeling, case-based reasoning, and constraint satisfaction, among others. Personalized and prioritized recommendations are an important part of many on-line e-commerce applications such as Amazon.com, Netflix, and Pandora.

## Keywords

Machine learning, deep learning, recommender systems, Movielens dataset.

## Thanks

We want to thank God who has given us the strength and patience to accomplish this work. We express our deep gratitude and respectful gratitude to our supervisor: Ms.Mediani. We would like to express our heartfelt thanks to all those who, through their work, their ideas, their presentations, their collaborations or their re-readings, have participated directly or indirectly in the realization of this project. Likewise, We thank the members of the jury for the honor they have shown us to accept to judge this modest work. We also thank all our teachers. Finally, We cannot forget to send a kind word to all our families and friends.

## Dedication

From the depths of our hearts, We dedicate this modest work to: Our parents for their patience, their love, their support To our brothers and sisters, may God protect them. To our friends and our comrades, To all the SEBIHI's families.

# Contents

# List of Figures

# Chapter 1

# General Introduction

## 1.1 Overview

The use case for a recommender system occurs regularly in news feed video websites; a user, Mustapha, constantly visits his favorite sports channels on YOUTUBE; the homepage lists current sports content creators and also a list containing recommended sports videos to watch. These videos might be uploaded by Mustapha's favorite sports athletes or sports specialists or news videos about Mustapha's favorite sports club. Whether Mustapha will find these suggestions helpful or distracting depends on how well they match his tastes. If we take this personalization away, we are left with a random list of top viewed or trending videos and will probably not match Mustapha's taste or interest.

Recommendation system research covers this situation and many other information access environments in which users and content creators can benefit from the presentation of personalized options. In the past ten years, interest in this field has dramatically expanded, partly due to the Netflix award, and the rapid development of the annual ACM recommendation system conference also proves this. a little. At this point, it is necessary to summarize, consider the reasons for distinguishing the research of recommendation systems from other related fields of artificial intelligence research and study the success and new challenges in this field.

## 1.2 What Is A Recommender System?

A recommender system, sometimes known as a recommendation engine, is a type of information filtering system that attempts to forecast a user's "rating" or "preference" for a given item[1][2].

Recommender systems are utilized in a assortment of zones, with commonly perceived cases taking the shape of playlist generators for video and music administrations, item recommenders for online stores, or substance recommenders for social media stages and open web substance recommenders[3][4]. These frameworks can work employing a single input, like music, or different inputs inside and over stages like news, books, and look inquiries. There are moreover well known recommender frameworks for particular themes like eateries and video websites. Recommender frameworks have moreover been created to investigate inquire about articles and specialists[5], collaborators[6], and financial services[7].

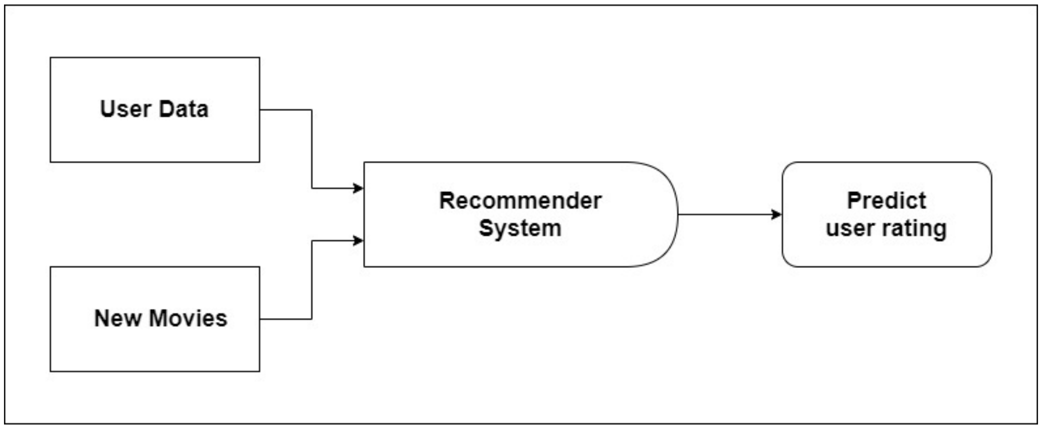


Figure 1.1: Recommender system  user ratings

## 1.3 Recommender System Approaches

### 1.3.1 Collaborative filtering

Collaborative filtering is a widely used method to the creation of recommender systems.[8] It's founded on the premise that people who agreed in the past will agree again in the future, and that they'll like the same kinds of things they loved before. Only information about rating profiles for various persons or things is used to generate suggestions by the algorithm. They generate recommendations utilizing this neighborhood by seeking peer users/items with a rating history similar to the current user or item. Memory-based and model-based collaborative filtering approaches are the two types. The user-based algorithm is a well-known example of memory-based approaches[9], while the Kernel-Mapping Recommender is an example of model-based approaches.[10]

The collaborative filtering strategy has the advantage of not relying on machine analyzable content, which allows it to accurately recommend complicated objects like movies without requiring a "knowledge" of the item. In recommender systems, many algorithms have been employed to measure user or item similarity. For instance, the k-nearest neighbor (k-NN) approach[11] and the Pearson Correlation, which Allen introduced.[12]

When creating a model based on a user's behavior, it's common to distinguish between explicit and implicit data collecting. Explicit data collection can include asking the user to rate, search or rank a collection of items from favorite to least favorite while the implicit data collection is observing the items that a user views in an online store, analyzing item/user viewing times

[13] and analyzing the user's social network and discovering similar likes and dislikes.

Item-to-item collaborative filtering (those who buy x also buy y), popularized by Amazon.com's recommender system, is one of the most well-known examples of collaborative filtering.[14]

By studying the network of connections between a user and their friends, several social networks used collaborative filtering to recommend new friends, groups, and other social connections.[2] In hybrid systems, collaborative filtering is still used.

### 1.3.2 Content-based filtering

Content-based filtering is another popular method for creating recommender systems. A description of the item and a profile of the user's preferences are used to create content-based filtering algorithms.[15][16] These strategies work best when there is known information about an item (name, location, description, etc.) but not about the user. Content-based recommenders approach suggestion as a user-specific classification problem, learning a classifier for a user's likes and dislikes based on the attributes of an item.

Keywords are utilized to define the items in this system, and a user profile is created to indicate the type of item this user prefers. In other words, these algorithms attempt to propose things that are comparable to those that a user has previously enjoyed or is now considering. It generates this frequently temporary profile without relying on a user sign-in process. Specifically, numerous candidate goods are compared to items that the user has already rated, and the best-matching items are recommended. This method has its origins in the fields of information retrieval and information filtering. The system focuses on two categories of information when creating a user profile, a model of the user's preference and a history of the user's interaction with the recommender system,

In essence, these strategies make use of an item profile to describe the object within the system. An item presentation algorithm is used to abstract the features of the items in the system. The tf–idf representation is a commonly used algorithm (also called vector space representation).[17] Based on a weighted vector of item attributes, the system produces a content-based user profile. The weights indicate the relevance of each feature to the user and can be calculated using a variety of approaches using individually assessed content vectors. In order to estimate the chance that the user would like the item, simple approaches use the average values of the rated item vector, while more sophisticated methods use machine learning techniques such as Bayesian Classifiers, cluster analysis, decision trees, and artificial neural networks.[18]

A fundamental question with content-based filtering is whether the system can learn user preferences from their activities with one form of content and apply them to other types of content. The utility of a recommendation system that is limited to recommending content of the same type as the user is now using is much lower than when different content types from other sources can be recommended. For instance, while proposing news articles based on news browsing is useful, it would be far more useful if music, movies, purchases, debates, and other services could be recommended based on news browsing. To get around this,Most content-based recommender systems now utilize a hybrid strategy to solve this.

Opinion-based recommender systems are a type of content-based recommender system. Users may be able to leave text reviews or feedback on things in some situations. These user-generated texts are implicit data for the recommender system because they are a potentially rich resource of both item features and user evaluation/sentiment. Features collected from user-generated evaluations improve item meta-data since they represent aspects of the item such as meta-data, and users are interested in these features. Users' rating scores on the respective features can be seen as sentiments gathered from the reviews. Text mining, information retrieval, sentiment analysis (see also Multimodal sentiment analysis), and deep learning are all popular techniques used in opinion-based recommender systems.[19]

### 1.3.3 Session-based recommender systems

These recommender systems provide recommendations based on a user's interactions throughout a session.[20] Youtube [21] and Amazon both use session-based recommender systems.[22] These are especially beneficial when a user's history (such as previous clicks or purchases) is unavailable or irrelevant in the current user session. Video, e-commerce, travel, music, and other domains benefit greatly from session-based suggestions. The majority of session-based recommender systems rely on the sequence of recent interactions inside a session without requiring any further information about the user (history, demographics). Recurrent Neural Networks,[49][23] Transformers,[24] and other deep learning-based techniques are used to make session-based recommendations.[25][26]

### 1.3.4 Reinforcement learning for recommender systems

The recommendation problem is a type of reinforcement learning problem in which the user is the environment in which the agent, or recommendation system, behaves in order to gain a reward, such as a click or engagement from the user.[21][27][28] The notion that the models or rules can be taught by rewarding the recommendation agent is one facet of reinforcement learning that is particularly useful in the field of recommender systems. In contrast to classical learning techniques, which rely on less flexible supervised learning approaches, reinforcement learning recommendation techniques enable for the training of models that can be optimized based on engagement and user interest indicators.[29]

### 1.3.5 Multi-criteria recommender systems

Multi-criteria recommender systems (MCRS) are recommender systems that take into account many factors when making recommendations. Rather than developing recommendation techniques based on a single criterion value, such as user u's overall preference for item I these systems attempt to predict a rating for unexplored items of u by leveraging preference information on multiple criteria that influence this overall preference value. Several researchers see MCRS as a multi-criteria decision-making (MCDM) problem and construct MCRS systems using MCDM approaches and techniques.[30] An extended introduction can be found in this chapter.[31]

### 1.3.6 Risk-aware recommender systems

The majority of existing recommender system approaches focus on recommending the most relevant content to consumers based on contextual information, but they ignore the risk of annoying the user with unsolicited messages. It's crucial to think about the possibility of upsetting the user by pushing recommendations during specific times of day, such as during a business meeting, early in the morning, or late at night. As a result, the recommender system's success is influenced in part by the degree to which risk is factored into the decision-making process. DRARS, a system that represents context-aware recommendation as a bandit problem, is one solution to this challenge. A content-based method and a contextual bandit algorithm are combined in this system.[32]

### 1.3.7 Mobile recommender systems

Smart phones with internet connectivity are used by mobile recommender systems to provide customized, context-sensitive recommendations. Mobile data is more complex than the data that recommender systems frequently work with, making this a particularly demanding field of research. It's diverse, noisy, necessitates spatial and temporal auto-correlation, and has challenges with validation and generality.[33] Context, recommendation method, and privacy are three elements that potentially influence mobile recommender systems and the accuracy of prediction outcomes.[34] Furthermore, mobile recommender systems face a transplanting problem, as recommendations may not be applicable in all areas.

The tactics adopted by firms like Uber and Lyft to develop driving routes for taxi drivers in a city are an example of a mobile recommender system.[33] This system leverages GPS data from taxi drivers' routes, including their position (latitude and longitude), time stamps, and operating status (with or without passengers). It uses this information to suggest a list of pickup locations along a route in order to maximize occupancy times and earnings.

### 1.3.8 Hybrid recommender systems

Most recommender systems currently employ a hybrid method that incorporates collaborative filtering, content-based filtering, and other techniques. There's no reason why multiple techniques of the same kind couldn't be combined. Hybrid techniques can be applied in a variety of ways, including producing content-based and collaborative-based predictions separately and then combining them; adding content-based capabilities to a collaborative-based approach (and vice versa); or unifying the approaches into a single model (see[35] for a complete review of recommender systems). Several studies comparing the performance of hybrid methods to pure collaborative and content-based methods have shown that hybrid methods can deliver more accurate suggestions than pure approaches. These strategies can also be utilized to solve difficulties like cold start and sparsity in recommender systems, as well as the knowledge engineering bottleneck in knowledge-based approaches.[36] Netflix is an excellent illustration of how hybrid recommender systems may be used.[37] The website provides recommendations by comparing comparable users' viewing and searching behaviors (collaborative filtering), as well as by suggesting films that share qualities with films that a user has rated highly (content-based filtering). Some hybridization techniques include:

- **Weighted:** Combining the score of different recommendation components numerically.[38]
- **Switching:** Choosing among recommendation components and applying the selected one.The text in the entries may be of any length.[38]
- **Mixed:** Recommendations from different recommenders are presented together to give the recommendation.[38]
- **Feature Combination:** Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.[38]
- **Feature Augmentation:** Computing a feature or set of features, which is then part of the input to the next technique.[38]
- **Cascade:** Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.[38]
- **Meta-level:** One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.[38]

## 1.4 Issues And Challenges

This section looks at the problems and obstacles that recommender systems face, as well as the many solutions proposed by researchers to address these problems.

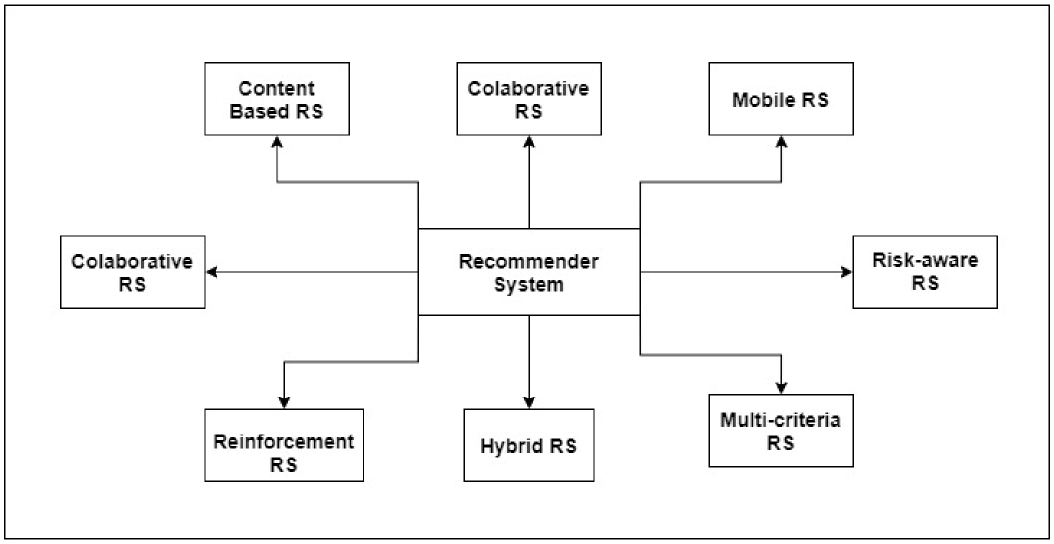


Figure 1.2: Recommender system approaches

## 1.4.1 Cold Start Problem

When new users join the system or new items are added to the catalog, this problem occurs. In such instances, neither the new users' tastes nor the new items' ratings or purchases can be expected, resulting in less reliable suggestions. The problem of a cold start can be remedied in a variety of ways, for instance, we can ask the user at the start to rate some items, or we can recommend things to the user based on the demographic data collected.[39]

## 1.4.2 Synonymy

When an item is represented by two or more separate names or entries with similar meanings, this is known as synonymy. In such circumstances, the recommender is unable to determine whether the phrases refer to separate or same things. A memory-based CF technique, for example, will treat "comedy movie" and "comedy film" differently. The use of descriptive phrases varies more than is often assumed, and the overuse of synonym words degrades the efficacy of CF recommenders. The recommender does not examine the latent relationship between items because the item contents are completely ignored.[40]

### 1.4.3 Shilling Attacks

What happens if a malevolent user or rival gains access to a system and begins to give fraudulent ratings on certain things, either to enhance or decrease their popularity . Such attacks have the potential to undermine trust in the recommender system, as well as reduce recommender performance and quality. This threat is more serious in CF techniques, although it is less serious in item-based CF techniques.[41]

### 1.4.4 Privacy

Personal information fed to recommender systems improves recommendation services, but it raises concerns about data privacy and security. Users are hesitant to provide data to recommender systems that have privacy concerns. As a result, a recommender system, whether CB or CF, should promote user confidence; nonetheless, CF recommenders are more vulnerable to privacy concerns.[40]

### 1.4.5 Limited Content Analysis and Overspecialization

Limited content analysis leads to overspecialization, with CB recommenders recommending items that are closely related to the user's profile rather than new items. We need to include extra hacks and take note of unpredictability in order to promote novel and serendipitous items alongside familiar ones, which can be accomplished by applying genetic algorithms that bring variation to recommendations. In CF recommenders, where unexpected and unique goods may be recommended, the problem is minor.[42]

### 1.4.6 Grey Sheep

Grey sheep happens in pure CF systems when a user's beliefs do not align with any of the groups and, as a result, the user is unable to benefit from recommendations. By utilizing the user's personal profile and the contents of the goods being recommended, pure CB filtering can fix this issue where items are suggested.[40]

### 1.4.7 Latency Problem

When new items are added to the database more often, CF recommenders encounter a latency problem, where the recommender only proposes previously rated items because the newly added items have not yet been rated. While CB filtering might help minimize wait times, it can also lead to overspecialization.The category-based strategy, in combination with the user archetype, can be utilized to deal with this problem.[43]

## 1.5 Recommender System Applications

### 1.5.1 e-Commerce

It was in this industry that recommender systems became widely used for the first time. E-commerce organizations are best placed to make accurate recommendations because they have millions of clients and data on their online behavior.[73]

### 1.5.2 Media

Media companies, like e-commerce, are among the first to embrace recommendations. It's rare to come across a news site that doesn't include a recommender system.[73]

### 1.5.3 Banking

Millions of people use a digital version of a mass market product. Banking for the people and SMEs are ideal candidates for referrals. Knowing a customer's financial condition in depth, as well as their historical preferences, when combined with data from thousands of other customers, is highly powerful.[73]

### 1.5.4 Telecom

Similar to banking in terms of dynamics. Telecom companies have access to millions of clients, all of whom have their every interaction recorded. In comparison to other businesses, their product choice is likewise quite narrow, making telecom recommendations a simpler task.[73]

### 1.5.5 Retail

Target frightened shoppers in the early 2000s when their technologies were able to anticipate pregnancies even before mothers were aware of their own. The most useful data is shopping data because it is the most direct indicator of a customer's intent. Retailers, who have vast amounts of shopping data, are at the forefront of businesses generating appropriate recommendations.[73]

## 1.6 Objectives And Method

To work around the problem of hybrid recommender systems, we will use a deep learning model that utilizes embeddings; Data is not always as simple as we would like. We can translate them into a format that an algorithm can understand using various approaches (numbers). In chapter two, we will take a brief look at the state-of-the-art latest machine learning approaches, and the related recommender system works. In the third chapter we will create and train the deep learning model using the movielens dataset. Last but not least we evaluate the the results and comparing them with each others.

# Chapter 2

# State Of The Art And Related Works

## 2.1 Introduction

This chapter presents the details of a number of publications and research investigations that are relevant to our thesis and have similar scope and principles, as well as the most widely utilized techniques and methods for delivering high-performing recommender systems.

## 2.2 State Of The Art

### 2.2.1 Machine learning

Machine learning (ML) is the study of computer algorithms that can learn and develop on their own with experience and data.[77] It is considered to be a type of artificial intelligence. Machine learning algorithms create a model based on sample data, referred to as "training data," in order to make predictions or decisions without being explicitly programmed.[78] Machine learning algorithms are utilized in a wide range of applications, including medicine, email filtering, speech recognition, and computer vision, where developing traditional algorithms to do the required tasks is difficult or impossible.[79]

### 2.2.2 Why machine Learning?

Machine learning is a major topic in recommender systems, in order to recommend new content (products, movies .. ), we must create a self-learning algorithm that learns from the user's actions and interactions with the items, rather than relying on a random method.

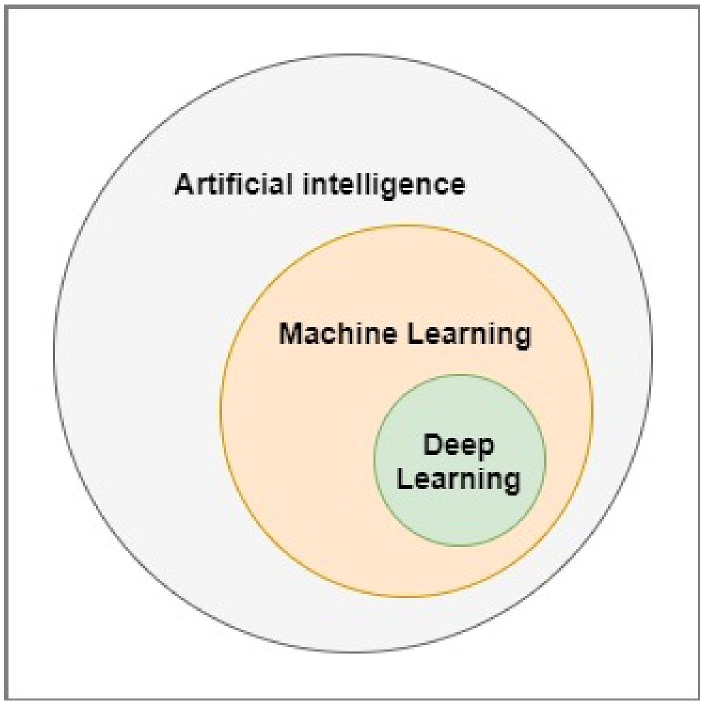


Figure 2.1: Machine learnig and deep learning

### 2.2.3 Machine learning approaches

Depending on the type of the "signal" or "feedback" available to the learning system, machine learning systems are generally categorized into three major categories:

#### 2.2.3.1 Supervised learning

Supervised learning algorithms create a mathematical model of a set of data that includes both the inputs and the outputs that are sought.[80] The information is referred to as training data, and it consists of a collection of training instances. Each training example has one or more inputs and a supervisory signal as the desired output. Each training sample is represented by an array or vector, sometimes referred to as a feature vector, and the training data is represented by a matrix in the mathematical model. Supervised learning techniques develop a function that may be used to predict the output associated with fresh inputs by iteratively optimizing an objective function.[81] The algorithm will be able to accurately estimate the output for inputs that were not part of the training data if it uses an optimum function. An algorithm that learns to complete a task is one that increases the accuracy of its outputs or predictions over time.[82]

#### 2.2.3.2 Unsupervised learning

Unsupervised learning methods take a collection of data with only inputs and detect structure in it, such as data point grouping or clustering. As a result, the algorithms learn from unlabeled, unclassified, and uncategorized test data. Unsupervised learning algorithms discover commonalities in the data and react depending on the existence or lack of such commonalities in each new piece of data, rather than responding to feedback. The field of density estimation in statistics, such as calculating the probability density function, is a key application of unsupervised learning.[83]

#### 2.2.3.3 Reinforcement learning

Reinforcement learning is a branch of machine learning that studies how software agents should behave in a given environment in order to maximize some metric of cumulative reward. Game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics, and genetic algorithms are among the numerous disciplines that study the field due to its generality. The environment is generally represented as a Markov decision process in machine learning (MDP). Dynamic programming techniques are used in many reinforcement learning systems. When exact mathematical models of the MDP are infeasible, reinforcement learning procedures are applied. Reinforcement learning algorithms are employed in autonomous vehicles and in teaching humans how to play a game.[84]

### 2.2.4 Machine Learning Algorithms

#### 2.2.4.1 Linear regression

A linear strategy to modeling the relationship between a scalar answer and one or more explanatory factors is known as linear regression in statistics (also known as dependent and independent variables). Simple linear regression is used when there is only one explanatory variable; multiple linear regression is used when there are more than one.[85] Multivariate linear regression, on the other hand, predicts numerous correlated dependent variables rather than a single scalar variable.[86]

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i \tag{2.1}$$

#### 2.2.4.2 Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (for example, whether a coin flip will come up heads or tails), each branch reflects the test's conclusion, and each leaf

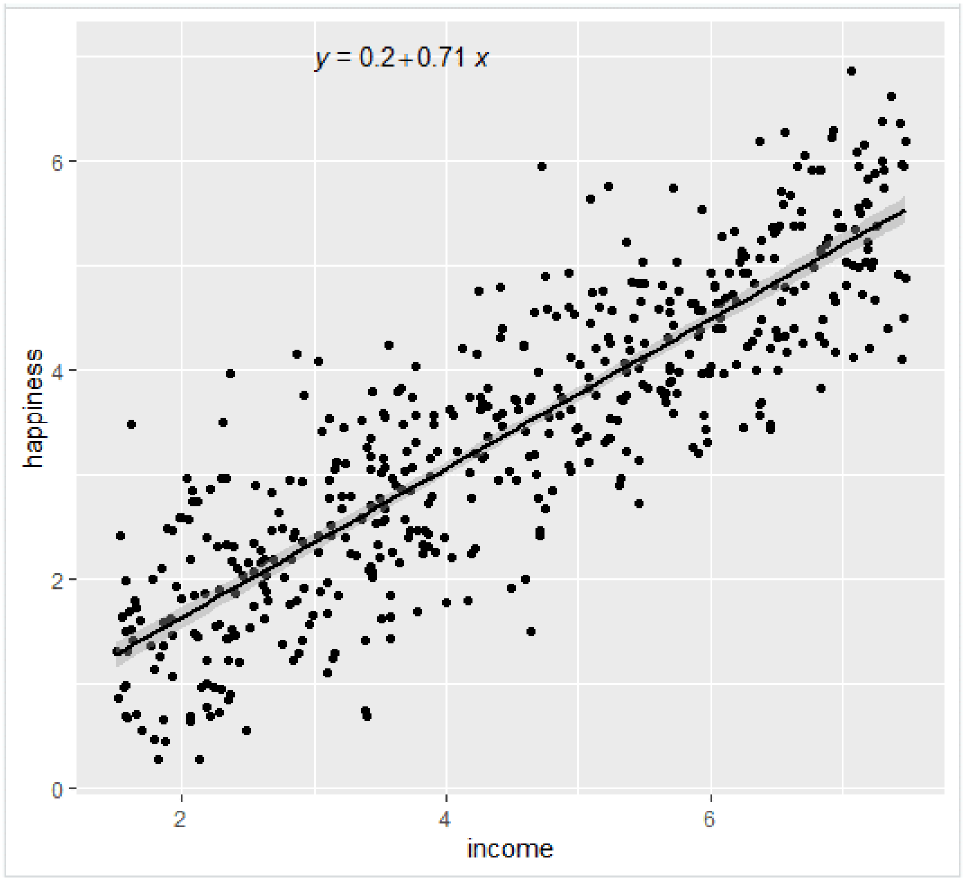


Figure 2.2: Happiness-income regression

node represents a class label (decision taken after computing all attributes). The categorization rules are represented by the pathways from root to leaf. A decision tree and its closely related influence diagram are used as a visual and analytical decision support tool in decision analysis to calculate the expected values (or expected utility) of competing alternatives.[87]

There are three sorts of nodes in a decision tree:

- Decision nodes – typically represented by squares

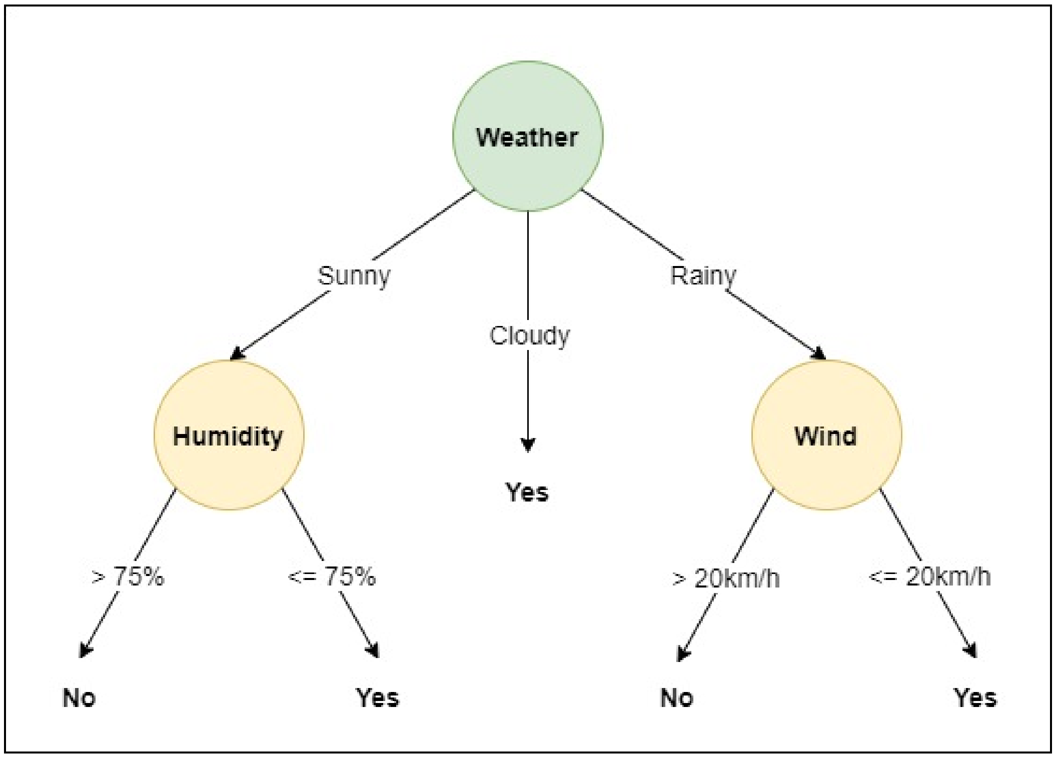


Figure 2.3: Weather decision tree

- Chance nodes – typically represented by circles
- End nodes – typically represented by triangles

#### 2.2.4.3 Random Forests

Random forests, also known as random decision forests, are an ensemble learning method for classification, regression, and other problems that works by training a large number of decision trees. For classification tasks, the random forest's output is the class chosen by the majority of trees. The mean or average prediction of the individual trees is returned for regression tasks.[88][89] Random decision forests address the problem of decision trees overfitting their training set.[90] 587–588 Random forests outperform decision trees in most cases, but they are less accurate than gradient enhanced trees. Data features, on the other hand, can have an impact on their performance.[91][92]

### 2.2.5 Deep Learning

Deep learning is a type of machine learning technique that employs numerous layers to extract higher-level features from raw data[46](pp199–200). Lower layers in image processing, for example, may recognize edges, whereas higher layers may identify human-relevant notions like numerals, letters, or faces.

### 2.2.6 Biologically inspired

The human brain consists of a network of neurons connected by receptors and effectors. Dendrites are the receptors, and axons are the effectors.[47] The dendrites collect signals from many other neurons in a small region, a cell body or soma integrates the signals and forms a response signal, and a branching axon delivers the response through interactions with dendrite trees of many other neurons[48], as shown in (Figure 2.2).

### 2.2.7 Neural Network

A neural network is a network or circuit of neurons, or in today's terms, an artificial neural network made up of artificial neurons or nodes. Thus, a neural network can be either biological (made up of biological neurons) or artificial (made up of artificial neurons) and used to solve artificial intelligence

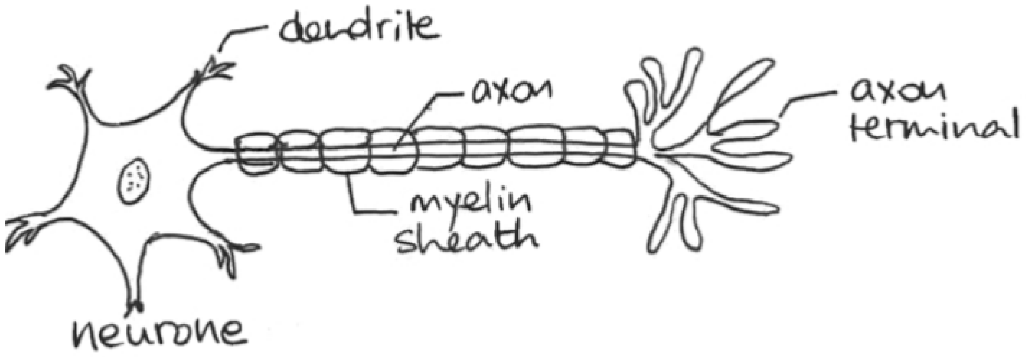


Figure 2.4: Biologically inspired

(AI) challenges. Artificial neural networks model the connections of biological neurons as weights between nodes. An excitatory link has a positive weight, while inhibitory connections have a negative weight. A weight is applied to all inputs before they are summed. A linear combination is the name for this action.[49]

### 2.2.8 Activation function

The activation function of a node in an artificial neural network determines the output of that node given an input or group of inputs. A conventional integrated circuit can be thought of as a digital network of activation functions that, depending on the input, can be "ON" (1) or "OFF" (0). In neural networks, this is analogous to the linear perceptron. Nonlinear activation functions, on the other hand, allow such networks to solve nontrivial problems with a limited number of nodes, and these activation functions are referred to as nonlinearities.[50] Here are some activation function examples:

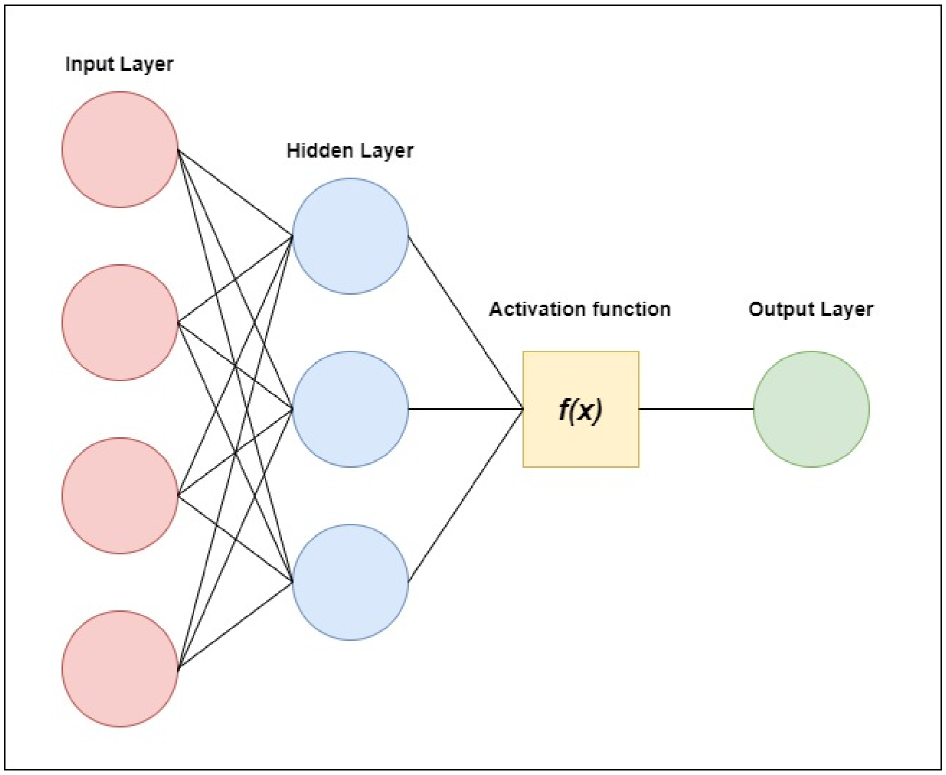


Figure 2.5: Neural network

### 2.2.9 Convolutional neural network

The name "convolutional neural network" reflects to the network's use of the convolutional mathematical procedure. Convolutional neural networks are a type of neural network that uses convolution rather than standard matrix multiplication in at least one layer. An input layer, hidden layers, and an output layer make up a convolutional neural network. Any middle layers in a feed-forward neural network are referred to as hidden since the activation function and final convolution mask their inputs and outputs. The hidden layers of a convolutional neural network include convolutional layers.This usually comprises a layer that does a dot product of the convolution kernel

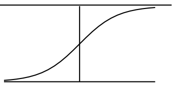
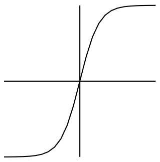
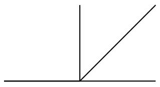
| Name | Function | Derivative | Figure |
|---|---|---|---|
| Sigmoid | $\sigma(x) = \frac{1}{1+e^{-x}}$ | $f'(x) = f(x)(1 - f(x))^2$ |  |
| tanh | $\sigma(x) = \frac{e^x - e^{-x}}{e^z + e^{-z}}$ | $f'(x) = 1 - f(x)^2$ |  |
| ReLU | $f(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \geq 0. \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 1 \geq 0. \end{cases}$ |  |
| Softmax | $f(x) = \frac{e^x}{\sum_i e^x}$ | $f'(x) = \frac{e^x}{\sum_i e^x} - \frac{(e^x)^2}{(\sum_i e^x)^2}$ | |

Table 2.1: Non-linear activation functions.

with the input matrix of the layer. The activation function of this product is usually ReLU, and it is usually the Frobenius inner product. The convolution procedure generates a feature map as the convolution kernel slides along the input matrix for the layer, which then contributes to the input of the following layer. Other layers such as pooling layers, fully connected layers, and normalization layers are added after this.[51]

### 2.2.10 Recurrent neural network

A recurrent neural network (RNN) is a type of artificial neural network in which nodes are connected in a directed graph that follows a temporal sequence. This enables it to behave in a temporally dynamic manner. RNNs, which are derived from feedforward neural networks, can process variable length sequences of inputs by using their internal state (memory).[52][53][54] As a result, they can be used for tasks like unsegmented, connected handwriting recognition[55] or speech recognition.[56][57]

### 2.2.11 Long short-term memory

Long short-term memory (LSTM) is a deep learning architecture[58] based on an artificial recurrent neural network (RNN). LSTM has feedback connections, unlike normal feedforward neural networks. It can handle not only individual data points (such as photos), but also complete data streams (such as speech or video). LSTM can be used for tasks like unsegmented, linked handwriting recognition[59], speech recognition[60][61], and anomaly detection in network traffic or IDSs. A cell, an input gate, an output gate, and a forget gate make up a typical LSTM unit. The three gates control the flow of information into and out of the cell, and the cell remembers values across arbitrary time intervals.

## 2.3 Related Works

Lianhuan Li, Zheng Zhang, and Shaoda Zhang proposed a hybrid recommender systems based on content filtering and collaborative filtering, and explored the workflow, user characteristic description, data processing algorithm, and recommendation strategy in depth. They created experiments to test the system performance of mixed mode recommendation technology, user-based collaborative filtering, and content-based filtering recommender system using the MovieLens dataset and the scientific literature experiment dataset, respectively. The experimental results suggest that the hybrid mode recommendation system, which combines content filtering and collaborative filtering, outperforms the two separate technologies.[44]

For the interactive situation, Heng-Ru Zhang, Fan Min, Xu He, and Yuan-Yuan Xu suggested a hybrid recommender system that can get the following findings by modifying the recommended parameters and comparing the random and hybrid algorithms: (1) As long as the ratio of random recommendations is not too large (e.g., not more than 0.25), (2) one should use NN as soon as possible, (3) the neighbors number should be large enough (e.g., 45), (4) the recall increases nearly linearly with the number of recommendations in each round, and (5) the hybrid algorithm is better than the random algorithm.[45]

Aaron van den Oord, Sander Dieleman, Benjamin Schrauwen looked at using deep convolutional neural networks (CNN) to infer latent characteristics from music audio when consumption data wasn't available. On an industrial-scale dataset, they tested the predictions by utilizing them for music recommendation. Despite the fact that many of the qualities of songs that influence user liking cannot be anticipated from audio data, the recommendations that result appear to be reasonable. they can infer that using music audio to predict latent characteristics is a promising strategy for promoting new and unpopular music.[93]

Based on certain assumptions and modeling aims, Young-Jun Ko, Lucas Maystre and Matthias Grossglauser presented a novel collaborative RNN-based model for analyzing sequences of user activity suitable for modern recommender systems. On two real-world datasets, they empirically demonstrated that these models consistently outperformed simple baseline techniques. The model is practical in that it can be scaled to big datasets utilizing techniques like Chen et al. (2015)'s sub-linear output layer assessment in large item sets, or Recht et al. (2011)'s parallelization training.[94]

Robin Devooght and Hugues Bersini looked into using a recurrent neural network, namely an LSTM, to solve the collaborative filtering challenge. Using RNNs necessitates re-framing collaborative filtering as a sequence prediction issue, which could result in richer models that take into account the change of users' tastes. On the Movielens and Netflix datasets, our trials revealed that the LSTM offers excellent results, particularly in terms of short-term prediction and item coverage.[95]

Ali Elkahky ,Yang Song and Xiaodong He demonstrated a broad recommendation framework that employs deep learning to link rich user characteristics to item characteristics. They also demonstrated how to extend this framework to mix data from many domains to increase suggestion quality even more. They then explored dimensionality reduction as a means to make this approach scalable to big data sets. Both existing and new users can be recommended using the proposed paradigm. Experiments on many large-scale real-world data sets revealed that the proposed method performed significantly better than other systems.[96]

## 2.4 Conclusion

We discussed some recommender system-related studies in this chapter, as well as their various work flows and outcomes. We were also able to incorporate a machine learning overview and popular algorithms.

# Chapter 3

# Analysis and Modelling

## 3.1 Implementation Environments

We'll look at the numerous and most often used environments and programming languages for developing deep learning models in this chapter, which we'll use to build the recommender system that we'll contemplate implementing.

### 3.1.1 Python

Python is a high-level, general-purpose programming language that is interpreted. Python was designed by Guido van Rossum and initially released in 1991. Its design philosophy emphasizes code readability and makes extensive use of whitespace. Its language constructs and object-oriented approach are designed to assist programmers in writing clear, logical code for both small and big projects.[62]

### 3.1.2 R Language

The R Core Team and the R Foundation for Statistical Computing maintain R, a programming language and free software environment for statistical computing and graphics. [69] For designing statistical applications and data analysis, statisticians and data miners frequently utilize the R programming language. R's popularity has risen significantly, according to polls, data mining surveys, and examinations of scientific literature databases.[70] Since August 2021, R has ranked 14th in the TIOBE index, a measure of programming language popularity.[71]

### 3.1.3 Anaconda

Anaconda is a Python and R programming language distribution aimed for simplifying package management and deployment in scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, and so on). Data-science packages for Windows, Linux, and macOS are included in the release. Anaconda, Inc., created by Peter Wang and Travis Oliphant in 2012, is responsible for its development and maintenance.[63]

### 3.1.4 Jupyter Notebook

Jupyter Notebook is an open-source web software that lets you create and share documents with live code, equations, visualizations, and narrative text. Data cleansing and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and many other applications are all possible.[64]

### 3.1.5 Kaggle

Kaggle, a Google LLC subsidiary, is an online community of data scientists and machine learning experts. Users can use Kaggle to search and publish data sets, study and construct models in a web-based data-science environment, collaborate with other data scientists and machine learning experts, and compete in data science competitions. Kaggle got its start in 2010 by offering machine learning competitions and now also offers a public data platform, a cloud-based workbench for data science, and Artificial Intelligence education. Its key personnel were Anthony Goldbloom and Jeremy Howard. Nicholas Gruen was founding chair succeeded by Max Levchin.[65][66]

### 3.1.6 Google Colab

Colab is a free Jupyter notebook environment that runs completely in the cloud. Most significantly, Colab does not require any setup, and the notebooks you create may be edited simultaneously by your team members in the same way that Google Docs documents can be edited. The most significant benefit is that Colab supports the majority of prominent machine learning libraries, which can be quickly loaded into your notebook.[67]

## 3.2 Dataset Details

We'll utilize the "Movielens 1M" Dataset to investigate our use case. The "Movilense 1M" Dataset contains 1,000,209 anonymous movie ratings from around 3,900 films.There are three main files in this Dataset:

### 3.2.1 Rating File

More than a million user ratings are stored in this file in the following format: (UserID,MovieID,Rating,Timestamp) where:

- **UserIDs** range between 1 and 6040.
- **MovieID** range between 1 and 3952.
- **Rating** are made on a 5-star scale (whole-star ratings only).

- **Timestamp** Timestamp is represented in seconds since the epoch as returned by time(2).
- Each user has at least 20 ratings.

### 3.2.2 User File

This file is used to store user data in an organized manner. All demographic data is voluntarily given by users and is not verified for correctness. This data collection only includes users who have submitted some demographic information. The structure of this file is as follows:

- **Gender** is denoted by a "M" for male and "F" for female.
- **Age** is chosen from the following ranges:

  **1:** "Under 18", **18:** "18-24", **25:** "25-34", **35:** "35-44", **45:** "45-49", **50:** "50-55", **56:** "56+".
- **Occupation** is chosen from the following choices: **1:** "Under 18", **18:** "18-24", **25:** "25-34", **35:** "35-44", **45:** "45-49", **50:** "50-55", **56:** "56+".

### 3.2.3 Movies File

Movies are stored in this file, which also contains their data, and are organized as follows:

- **Titles** are identical to titles provided by the IMDB (including year of release)

- **Genres** are pipe-separated and are selected from the following genres: **0:** "other" or not specified, **1:** "academic/educator", **2:** "artist", **3:** "clerical/admin", **4:** "college/grad student", **5:** "customer service", **6:** "doctor/health care", **7:** "executive/managerial", **8:** "farmer", **9:** "homemaker", **10:** "K-12 student", **11:** "lawyer", **12:** "programmer", **13:** "retired", **14:** "sales/marketing", **15:** "scientist", **16:** "self-employed", **17:** "technician/engineer", **18:** "tradesman/craftsman", **19:** "unemployed", **20:** "writer".

- Some MovieIDs do not correspond to a movie due to accidental duplicate entries and/or test entries.

- Movies are mostly entered by hand, so errors and inconsistencies may exist

### 3.2.4 Preprocessing Phase

As Exauhsting movielens dataset is, in this phase, we need to convert its files intoa list containing 3 batchsize-long arrays, representing the triplets. In other words we have a list of objects [A,P,N] , where A is an array of shape (batchsize, nuserfeatures), P and N are both arrays of shape (batchsize, nitemfeatures), each row of the array corresponds to a feature vector. Indeed the triplet referring to the first user (coupled to a movie evaluated positively and to a one negatively ranked) is, in order to train the model properly.

## 3.3 Proposed Model

In this section, we will propose and analyze a hybrid recommender system model based on deeplearning technology. This model includes two distinct filtering approaches for recommender systems: Content-based and Collaborative Filtering.

### 3.3.1 Encoding and Embeddings

Data isn't always as easy as we'd like it to be. We can use a variety of methods to convert them into a format that an algorithm can understand (numbers). Encoding and embedding are both methods for mapping category data into numerical vectors. The distinction between the two is that an encoder (such as a one-hot encoder) is a fixed function that associates a vector with each data row, whereas embedding vectors are low-dimensional and learnt. A neural network learns to locate things in an embedding space by clustering related elements together. To summarize, an encoding converts a categorical feature into an m-dimensional vector, where m is the number of categories. An embedding is a method of mapping categorical information into an n-dimensional vector, where n is a model hyper-parameter.[72]

### 3.3.2 Why Embeddings?

Because it doesn't make sense to treat each categorical value as wholly different from the others, we opt to employ embeddings instead (this is one-hot-encoding). In our scenario, we want to create a hybrid recommender system that will suggest movies to users. To get weights from the Embedding layer, Embeddings are based on training a Neural Network with categorical data.[72]

### 3.3.3 Siamese neural network

is an artificial neural network that computes comparable output vectors using the same weights while working in parallel on two different input vectors. One of the output vectors is frequently precomputed, creating a baseline against which the other output vector is measured. This is analogous to comparing fingerprints or, in a more technical sense, a distance function for Locality-sensitive hashing, and it's frequently used in conjunction with Triplet loss.[74]

### 3.3.4 Triplet loss

A loss function for artificial neural networks that compares a baseline (anchor) input to a positive (truthy) input and a negative (falsy) input is known as triplet loss. The distance between the baseline (anchor) input and the positive (truthy) input is reduced to a minimum, while the distance between the baseline (anchor) input and the negative (falsy) input is increased. In triplet loss, we start with a triplet as an input — (anchor, positive, negative). The anchor can be any image of the person, while the positive can be another image of the same person and the negative can be a different image of the same person.[75] The following is a description of the loss function:

$$L = \max(\delta(a,p) - \delta(a,n) + \alpha, 0) \tag{3.1}$$

where $\delta$(a,p) is the distance between anchor image and positive image.

Similarly, $\delta$(a,n) is the distance between anchor and negative image. In the training phase we try to minimise such a loss.

### 3.3.5 Bayesian Personalised Ranking

The key difference between our recommender system and the image recognition task, as you may have seen, is that the user and the items are two separate entities. We need a function to optimize, such as the triplet loss function, so that during the training phase, we alter model parameters to increase the chance of recommending positive samples and decrease the probability of recommending negative samples. Bayesian Personalized Ranking criterion comes to help.[76] The function can be expressed like:

$$L = 1 - \delta(Uemb \times Pemb - Uemb \times Nemb) \tag{3.2}$$

where $\delta$ denotes the sigmoid function while user, positive item and negative item are represented by their embedding vectors. It's worth noting that we can now exploit user and item features while also taking into consideration previously provided feedback in a very straightforward method. This is an example of a hybrid recommender system in action. We believe this effectively expresses the embedding tool's capability.

### 3.3.6 Model

Let's put it all together in a neural network. We'll make advantage of the Keras functional API. First, we must establish a training network. Because Keras lacks a triplet loss layer, as a result, the model depicted in the figure below is created. (fig 3.1)

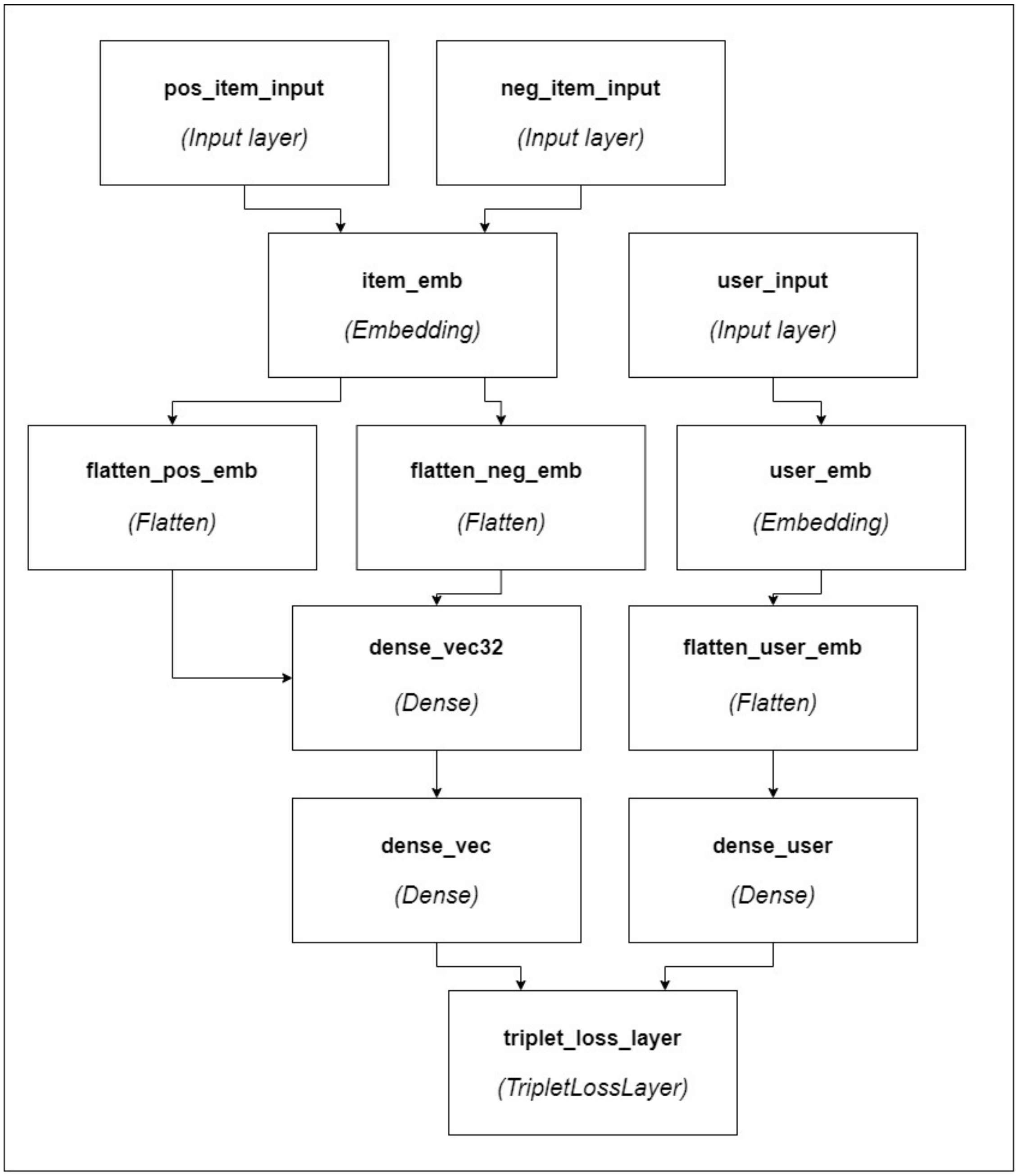


Figure 3.1: Embedding Model

# Chapter 4

# Experiments And Results

## 4.1 Our Experiment

In our case, we developed a model that generates embeddings (i.e. vectors) that may be used to calculate distances. In other words, if an object i is likely to be appreciated by a user u, their distance in the embedding space will be minimal; otherwise, they will be separated by a large distance. As a result, we require a threshold: if the found distance is less than the threshold, the item is recommended to the user; if the distance is greater than the threshold, the item is not recommended. It is a threshold that must be carefully chosen. This is a common ROC curve issue. As a result, one statistic for evaluating performance could be the Area Under the Curve (AUC).

## 4.2 Results

By changing the hyper parameters such as the interval for evaluating on one-shot tasks(`evaluate_every`) , number of training iterations (`n_iter`), the batch size (`batch_size`) and how many one-shot tasks to validate on (`n_val`) and here are the side by side comparison and results:

### Experiment 1

In the first experiment we used as hyper parameters:

- `evaluate_every` = 100
- ‘`n_iter`= 5000
- `batch_size` = 10

- `n_val`= 1000

and we got as result:

| AUC | Sensitivity | Train Loss | Execution time |
|---|---|---|---|
| 1.0 | 100% | -0.71 | 1.9 mins |

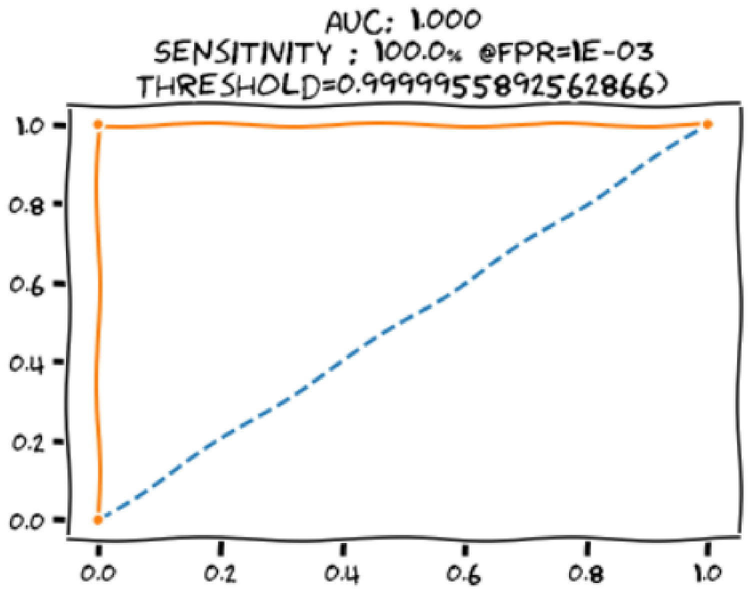


Figure 4.1: The AUC after the first experiment

## Experiment 2

In the second experiment we used as hyper parameters:

- `evaluate_every` = 100
- ‘`n_iter`= 10000
- `batch_size` = 32
- `n_val`= 1000

and we got as result:

| AUC | Sensitivity | Train Loss | Execution time |
|---|---|---|---|
| 0.999 | 99.9% | -0.76 | 9.3 mins |

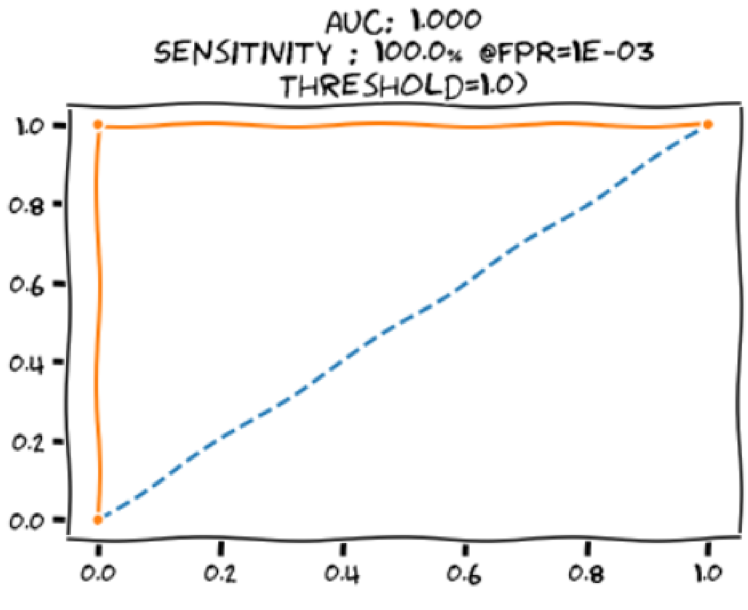


Figure 4.2: The AUC after the second experiment

## Experiment 3

In the third and last experiment we used as hyper parameters:

- `evaluate_every` = 100
- ‘`n_iter`= 100000
- `batch_size` = 10
- `n_val`= 1000

and we got as result:

| AUC | Sensitivity | Train Loss | Execution time |
|---|---|---|---|
| 0.999 | 99.7% | -0.94 | 50.4 mins |

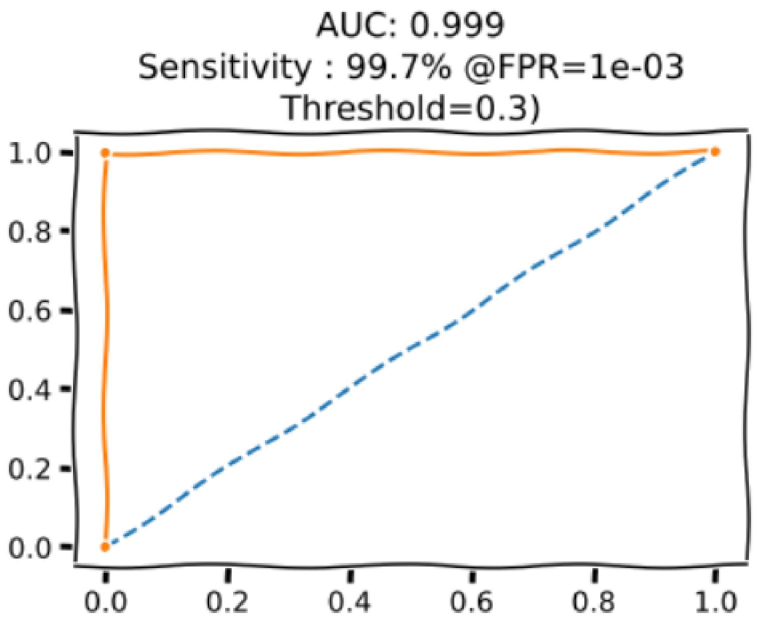


Figure 4.3: The AUC after the third experiment

## 4.3 Predictions: Making actual recommendations

This was a really straightforward application. We believe that this can be built upon to provide interesting results. Performances would be much improved with more thorough hyper parameter tuning and a better choice of Neural Network architecture. It would be interesting to see whether we could get significantly better performance by using a model to acquire optimized hyper-parameters.

# Chapter 5

# General Conclusion

The goal of a recommender system is to suggest items to users that are likely to be enjoyed. We can sort a list of items by "preferred score" if we have a user and a list of items. In this thesis, we defined recommender systems and gave a brief overview of their various approaches and obstacles, as well as some examples of their applications.
Thus, We highlighted the role of machine learning in general recommender systems, as well as renowned machine learning algorithms including linear regression, decision trees, and random forests, as well as deep learning algorithms like CNN RNN Lstm ets.

Finally, despite the limited time and resources, we were able to create a deep learning model that uses embeddings and got reasonable results after testing it.

This was a really straightforward work. Although the model was not perfectly optimized, we believe it can be improved upon to produce intriguing results. Performances would be much improved with more thorough hyperparameter tuning and a better choice of Neural Network architecture. It would be interesting to see whether we could get significantly better performance by using a model to acquire optimized hyper-parameters. Working on the input layers is another way to improve. Cleaning and maybe enriching the data, taking into account (for example) geographical data, would be good.

# Bibliography

[1] Piyush Kumar et al. “Movie Recommender System Using Machine Learning Algorithms”. In:Journal of Physics: Conference Series1916.1 (May2021), p. 012052.doi:10.1088/1742- 6596/1916/1/012052.url:https://doi.org/10.1088/1742-6596/1916/1/012052.

[2] Francesco Ricci, Lior Rokach, and Bracha Shapira. “Recommender Sys-tems Handbook”. In: vol. 1-35. Oct. 2010, pp. 1–35.doi:10.1007/978-0-387-85820-$3_1$.

[3] Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Bosagh Zadeh WTF:The who-to-follow system at Twitter, Proceedings of the 22nd international conference on World Wide Web

[4] Baran, Remigiusz; Dziech, Andrzej; Zeja, Andrzej (2018-06-01). "A capable multimedia content discovery platform based on visual content analysis and intelligent data enrichment". Multimedia Tools and Applications. 77 (11): 14077–14091. doi:10.1007/s11042-017-5014-1. ISSN 1573-7721. S2CID 36511631.

[5] H. Chen, A. G. Ororbia II, C. L. Giles ExpertSeer: a Keyphrase Based Expert Recommender for Digital Libraries, in arXiv preprint 2015

[6] H. Chen, L. Gou, X. Zhang, C. Giles Collabseer: a search engine for

collaboration discovery, in ACM/IEEE Joint Conference on Digital Libraries (JCDL) 2011

[7] Alexander Felfernig, Klaus Isak, Kalman Szabo, Peter Zachar, The VITA Financial Services Sales Support Environment, in AAAI/IAAI 2007, pp. 1692-1699, Vancouver, Canada, 2007.

[8] John S. Breese; David Heckerman  Carl Kadie (1998). Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence (UAI'98). arXiv:1301.7363.

[9] Breese, John S.; Heckerman, David; Kadie, Carl (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering (PDF) (Report). Microsoft Research.

[10] Ghazanfar, Mustansar Ali; Prügel-Bennett, Adam; Szedmak, Sandor (2012-11-15). "Kernel-Mapping Recommender system algorithms". Information Sciences. 208: 81–104. CiteSeerX 10.1.1.701.7729. doi:10.1016/j.ins.2012.04.012.

[11] Sarwar, B.; Karypis, G.; Konstan, J.; Riedl, J. (2000). "Application of Dimensionality Reduction in Recommender System A Case Study".,

[12] Allen, R.B. (1990). "User Models: Theory, Method, Practice". International J. Man-Machine Studies.

[13] Parsons, J.; Ralph, P.; Gallagher, K. (July 2004). "Using viewing time to infer user preference in recommender systems". AAAI Workshop in Semantic Web Personalization, San Jose, California.

[14] Collaborative Recommendations Using Item-to-Item Similarity Mappings Archived 2015-03-16 at the Wayback Machine

[15] Aggarwal, Charu C. (2016). Recommender Systems: The Textbook.

Springer. ISBN 9783319296579.

[16] Peter Brusilovsky (2007). The Adaptive Web. p. 325. ISBN 978-3-540-72078-2.

[17] D.H. Wang, Y.C. Liang, D.Xu, X.Y. Feng, R.C. Guan(2018), "A content-based recommender system for computer science publications", Knowledge-Based Systems, 157: 1-9

[18] Blanda, Stephanie (May 25, 2015). "Online Recommender Systems – How Does a Website Know What I Want?". American Mathematical Society. Retrieved October 31, 2016.

[19] X.Y. Feng, H. Zhang, Y.J. Ren, P.H. Shang, Y. Zhu, Y.C. Liang, R.C. Guan, D. Xu, (2019), "The Deep Learning–Based Recommender System "Pubmender" for Choosing a Biomedical Publication Venue: Development and Validation Study", Journal of Medical Internet Research, 21 (5): e12957

[20] Hidasi, Balázs; Karatzoglou, Alexandros; Baltrunas, Linas; Tikk, Domonkos (2016-03-29). "Session-based Recommendations with Recurrent Neural Networks". arXiv:1511.06939 [cs.LG].

[21] Chen, Minmin; Beutel, Alex; Covington, Paul; Jain, Sagar; Belletti, Francois; Chi, Ed (2018). "Top-K Off-Policy Correction for a REINFORCE Recommender System". arXiv:1812.02353 [cs.LG].

[22] Yifei, Ma; Narayanaswamy, Balakrishnan; Haibin, Lin; Hao, Ding (2020). "Temporal-Contextual Recommendation in Real-Time". KDD '20: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining. Association for Computing Machinery: 2291–2299. doi:10.1145/3394486.3403278. ISBN 9781450379984. S2CID 221191348.

[23] Hidasi, Balázs; Karatzoglou, Alexandros (2018-10-17). "Recurrent

Neural Networks with Top-k Gains for Session-based Recommendations". Proceedings of the 27th ACM International Conference on Information and Knowledge Management. CIKM '18. Torino, Italy: Association for Computing Machinery: 843–852. arXiv:1706.03847. doi:10.1145/3269206.3271761. ISBN 978-1-4503-6014-2. S2CID 1159769.

[24] Kang, Wang-Cheng; McAuley, Julian (2018). "Self-Attentive Sequential Recommendation". arXiv:1808.09781 [cs.IR].

[25] Li, Jing; Ren, Pengjie; Chen, Zhumin; Ren, Zhaochun; Lian, Tao; Ma, Jun (2017-11-06). "Neural Attentive Session-based Recommendation". Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. CIKM '17. Singapore, Singapore: Association for Computing Machinery: 1419–1428. arXiv:1711.04725. doi:10.1145/3132847.3132926. ISBN 978-1-4503-4918-5. S2CID 21066930.

[26] Liu, Qiao; Zeng, Yifu; Mokhosi, Refuoe; Zhang, Haibin (2018-07-19). "STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation". Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '18. London, United Kingdom: Association for Computing Machinery: 1831–1839. doi:10.1145/3219819.3219950. ISBN 978-1-4503-5552-0. S2CID 50775765.

[27] Xin, Xin; Karatzoglou, Alexandros; Arapakis, Ioannis; Jose, Joemon (2020). "Self-Supervised Reinforcement Learning for Recommender Systems". arXiv:2006.05779 [cs.LG].

[28] Ie, Eugene; Jain, Vihan; Narvekar, Sanmit; Agarwal, Ritesh; Wu, Rui; Cheng, Heng-Tze; Chandra, Tushar; Boutilier, Craig. "SlateQ: A Tractable Decomposition for Reinforcement Learning with Recommendation

Sets". Proceedings of the Twenty-eighth International Joint Conference on Artificial Intelligence (IJCAI-19).

[29] Zou, Lixin; Xia, Long; Ding, Zhuoye; Song, Jiaxing; Liu, Weidong; Yin, Dawei (2019). "Reinforcement Learning to Optimize Long-term User Engagement in Recommender Systems". KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining: 2810–2818. arXiv:1902.05570. doi:10.1145/3292500.3330668. ISBN 9781450362016. S2CID 62903207.

[30] Lakiotaki, K.; Matsatsinis; Tsoukias, A (March 2011). "Multicriteria User Modeling in Recommender Systems". IEEE Intelligent Systems. 26 (2): 64–76. CiteSeerX 10.1.1.476.6726. doi:10.1109/mis.2011.33. S2CID 16752808.

[31] Gediminas Adomavicius, Nikos Manouselis, YoungOk Kwon. "Multi-Criteria Recommender Systems" (PDF). Archived from the original (PDF) on 2014-06-30.

[32] Bouneffouf, Djallel (2013), DRARS, A Dynamic Risk-Aware Recommender System (Ph.D.), Institut National des Télécommunications

[33] Yong Ge; Hui Xiong; Alexander Tuzhilin; Keli Xiao; Marco Gruteser; Michael J. Pazzani (2010). An Energy-Efficient Mobile Recommender System (PDF). Proceedings of the 16th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York City, New York: ACM. pp. 899–908. Retrieved 2011-11-17.

[34] Pimenidis, Elias; Polatidis, Nikolaos; Mouratidis, Haralambos (3 August 2018). "Mobile recommender systems: Identifying the major concepts". Journal of Information Science. 45 (3): 387–397. arXiv:1805.02276.

doi:10.1177/0165551518792213. S2CID 19209845.

[35] Adomavicius, G.; Tuzhilin, A. (June 2005). "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions". IEEE Transactions on Knowledge and Data Engineering. 17 (6): 734–749. CiteSeerX 10.1.1.107.2790. doi:10.1109/TKDE.2005.99. S2CID 206742345.

[36] Rinke Hoekstra, The Knowledge Reengineering Bottleneck, Semantic Web – Interoperability, Usability, Applicability 1 (2010) 1, IOS Press

[37] Gomez-Uribe, Carlos A.; Hunt, Neil (28 December 2015). "The Netflix Recommender System". ACM Transactions on Management Information Systems. 6 (4): 1–19. doi:10.1145/2843948.

[38] Robin Burke, Hybrid Web Recommender Systems Archived 2014-09-12 at the Wayback Machine, pp. 377-408, The Adaptive Web, Peter Brusilovsky, Alfred Kobsa, Wolfgang Nejdl (Ed.), Lecture Notes in Computer Science, Springer-Verlag, Berlin, Germany, Lecture Notes in Computer Science, Vol. 4321, May 2007, 978-3-540-72078-2.

[39] Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: The adaptive web, pp. 291–324. Springer (2007)

[40] Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Advances in Artificial Intelligence 2009, 4 (2009)

[41] Gunes, I., Kaleli, C., Bilge, A., Polat, H.: Shilling attacks against recommender systems: a comprehensive survey. Artificial Intelligence Review 42, 767–799 (2014)

[42] Montaner, M., López, B., De La Rosa, J.L.: A taxonomy of recom-

mender agents on the internet. Artificial Intelligence Review 19, 285–330 (2003)

[43] Sollenborn, M., Funk, P.: Category-based filtering and user stereotype cases to reduce the latency problem in recommender systems. In: Advances in Case-Based Reasoning, pp. 395–405. Springer (2002)

[44] ianhuan Li, Zheng Zhang, and Shaoda Zhang. "Hybrid Algorithm Basedon Content and Collaborative Filtering in Recommendation SystemOptimization and Simulation". In:Scientific Programming2021 (May2021), p. 7427409.issn: 1058-9244.doi:10.1155/2021/7427409.url:https://doi.org/10.1155/2021/7427409.

[45] Heng-Ru Zhang et al. "A Hybrid Recommender System Based on User-Recommender Interaction". In:Mathematical Problems in Engineering2015SN - 1024-123X (Feb. 2015), p. 145636.doi:10 . 1155 / 2015 /145636.url:https://doi.org/10.1

[46] Deng, L.; Yu, D. (2014). "Deep Learning: Methods and Applications" (PDF). Foundations and Trends in Signal Processing. 7 (3–4): 1–199. doi:10.1561/2000000039. Archived (PDF) from the original on 2016-03-14. Retrieved 2014-10-18.

[47] Dr. R. R. Srikanth. "Application of ANN in Condition Monitoring: A Case Study", (Conference proceeding "Condition Monitoring of Mechanical System") (2009), Gitam University, Vishakhapattanam, Page no. 31-44.

[48] R.R.Navthar Dr. N.V. Halegowda, "Analysis of oil film thickness in Hydrodynamic Journal Bearing using Artificial Neural Networks", Ciit International Journal of Artificial Intelligent System Machine Learning (Nov. 2011), Volume 3, No.12, Page no. -762-766

[49] Hopfield, J. J. (1982). "Neural networks and physical systems with emergent collective computational abilities". Proc. Natl. Acad. Sci. U.S.A.

79 (8): 2554–2558. Bibcode:1982PNAS...79.2554H. doi:10.1073/pnas.79.8.2554. PMC 346238. PMID 6953413.

[50] Hinkelmann, Knut. "Neural Networks, p. 7" (PDF). University of Applied Sciences Northwestern Switzerland.

[51] Ian Goodfellow and Yoshua Bengio and Aaron Courville (2016). Deep Learning. MIT Press. p. 326.

[52] Dupond, Samuel (2019). "A thorough review on the current advance of neural network structures". Annual Reviews in Control. 14: 200–230.

[53] Abiodun, Oludare Isaac; Jantan, Aman; Omolara, Abiodun Esther; Dada, Kemi Victoria; Mohamed, Nachaat Abdelatif; Arshad, Humaira (2018-11-01). "State-of-the-art in artificial neural network applications: A survey". Heliyon. 4 (11): e00938. doi:10.1016/j.heliyon.2018.e00938. ISSN 2405-8440. PMC 6260436. PMID 30519653.

[54] Tealab, Ahmed (2018-12-01). "Time series forecasting using artificial neural networks methodologies: A systematic review". Future Computing and Informatics Journal. 3 (2): 334–340. doi:10.1016/j.fcij.2018.10.003. ISSN 2314-7288.

[55] Graves, Alex; Liwicki, Marcus; Fernandez, Santiago; Bertolami, Roman; Bunke, Horst; Schmidhuber, Jürgen (2009). "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (5): 855–868. CiteSeerX 10.1.1.139.4502. doi:10.1109/tpami.2008.137. PMID 19299860. S2CID 14635907.

[56] Sak, Haşim; Senior, Andrew; Beaufays, Françoise (2014). "Long Short-Term Memory recurrent neural network architectures for large scale

acoustic modeling" (PDF).

[57] Li, Xiangang; Wu, Xihong (2014-10-15). "Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition". arXiv:1410.4281 [cs.CL].

[58] Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. PMID 9377276. S2CID 1915014.

[59] Graves, A.; Liwicki, M.; Fernandez, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. (2009). "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (5): 855–868. CiteSeerX 10.1.1.139.4502. doi:10.1109/tpami.2008.137. PMID 19299860. S2CID 14635907.

[60] Sak, Hasim; Senior, Andrew; Beaufays, Francoise (2014). "Long Short-Term Memory recurrent neural network architectures for large scale acoustic modeling" (PDF). Archived from the original (PDF) on 2018-04-24.

[61] Li, Xiangang; Wu, Xihong (2014-10-15). "Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition". arXiv:1410.4281 [cs.CL].

[62] https://https://docs.python.org/3/tutorial/index.html.

[63] "About Anaconda". Archived from the original on 19 April 2020. Retrieved 27 April 2020.

[64] https://jupyter.org/

[65] Lardinois, Frederic; Mannes, John; Lynley, Matthew (March 8, 2017). "Google is acquiring data science community Kaggle". Techcrunch. Archived from the original on March 9, 2017. Retrieved March 9, 2017. Sources tell us

that Google is acquiring Kaggle [...] the official announcement could come as early as tomorrow.

[66] "Google buys Kaggle and its gaggle of AI geeks". CNET. 2017-03-08. Retrieved 2018-06-01.

[67] Dr. Michael J. "What is google colab ?" Garbadehttps://blog.education-ecosystem.com/what-is-google-colab/. January 15, 2021.

[69] R language and environment Hornik, Kurt (4 October 2017). "R FAQ". The Comprehensive R Archive Network. 2.1 What is R?. Retrieved 6 August 2018. R Foundation Hornik, Kurt (4 October 2017). "R FAQ". The Comprehensive R Archive Network. 2.13 What is the R Foundation?. Retrieved 6 August 2018. The R Core Team asks authors who use R in their data analysis to cite the software using: R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

[70] R's popularity David Smith (2012); R Tops Data Mining Software Poll, R-bloggers, May 31, 2012. Karl Rexer, Heather Allen, Paul Gearan (2011); 2011 Data Miner Survey Summary, presented at Predictive Analytics World, Oct. 2011. Robert A. Muenchen (2012). "The Popularity of Data Analysis Software". Tippmann, Sylvia (29 December 2014). "Programming tools: Adventures with R". Nature. 517 (7532): 109–110. doi:10.1038/517109a. PMID 25557714.

[71] "TIOBE Index - The Software Quality Company". TIOBE. Retrieved 16 August 2021.

[72] Thushan Ganegedara, Intuitive Guide to Understanding GloVe Embeddings, "https://towardsdatascience.com/light-on-math-ml-intuitive-guide-

to-understanding-glove-embeddings-b13b4f19c010" May 5, 2019

[73] Cem Dilmegani , Recommendation Systems: Applications, Examples Benefits "https://research.aimultiple.com/recommendation-system/" PUBLISHED ON AUGUST 11, 2017,

[74] Chicco, Davide (2020), "Siamese neural networks: an overview", Artificial Neural Networks, Methods in Molecular Biology, 2190 (3rd ed.), New York City, New York, USA: Springer Protocols, Humana Press, pp. 73–94, doi:10.1007/978-1-0716-0826-$5_3, ISBN 978-1-0716-0826-5, PMID 32804361$

[75] Chechik, G.; Sharma, V.; Shalit, U.; Bengio, S. (2010). "Large Scale Online Learning of Image Similarity Through Ranking" (PDF). Journal of Machine Learning Research. 11: 1109–1135.

[76] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, Lars Schmidt-Thieme, "BPR: Bayesian Personalized Ranking from Implicit Feedback" "https://arxiv.org/abs/1205.2618" 9 May 2012

[77] Mitchell, Tom (1997). Machine Learning. New York: McGraw Hill. ISBN 0-07-042807-7. OCLC 36417892.

[78] The definition "without being explicitly programmed" is often attributed to Arthur Samuel, who coined the term "machine learning" in 1959, but the phrase is not found verbatim in this publication, and may be a paraphrase that appeared later. Confer "Paraphrasing Arthur Samuel (1959), the question is: How can computers learn to solve problems without being explicitly programmed?" in Koza, John R.; Bennett, Forrest H.; Andre, David; Keane, Martin A. (1996). Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming. Artificial Intelligence in Design '96. Springer, Dordrecht. pp. 151–170.

doi:10.1007/978-94-009-0279-$4_9$.

[79] Hu, J.; Niu, H.; Carrasco, J.; Lennox, B.; Arvin, F., "Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning" IEEE Transactions on Vehicular Technology, 2020.

[80] Russell, Stuart J.; Norvig, Peter (2010). Artificial Intelligence: A Modern Approach (Third ed.). Prentice Hall. ISBN 9780136042594.

[81] Mohri, Mehryar; Rostamizadeh, Afshin; Talwalkar, Ameet (2012). Foundations of Machine Learning. The MIT Press. ISBN 9780262018258.

[82] Mitchell, T. (1997). Machine Learning. McGraw Hill. p. 2. ISBN 978-0-07-042807-2.

[83] Mohri, Mehryar; Rostamizadeh, Afshin; Talwalkar, Ameet (2012). Foundations of Machine Learning. The MIT Press. ISBN 9780262018258.

[84] van Otterlo, M.; Wiering, M. (2012). Reinforcement learning and markov decision processes. Reinforcement Learning. Adaptation, Learning, and Optimization. 12. pp. 3–42. doi:10.1007/978-3-642-27645-$3_1.ISBN978-3-642-27644-6$.

[85] David A. Freedman (2009). Statistical Models: Theory and Practice. Cambridge University Press. p. 26. A simple regression equation has on the right hand side an intercept and an explanatory variable with a slope coefficient. A multiple regression e right hand side, each with its own slope coefficient

[86] Rencher, Alvin C.; Christensen, William F. (2012), "Chapter 10, Multivariate regression – Section 10.1, Introduction", Methods of Multivariate Analysis, Wiley Series in Probability and Statistics, 709 (3rd ed.), John Wiley  Sons, p. 19, ISBN 9781118391679.

[87] Kamiński, B.; Jakubczyk, M.; Szufel, P. (2017). "A framework for sensitivity analysis of decision trees". Central European Journal of Operations Research. 26 (1): 135–159. doi:10.1007/s10100-017-0479-6. PMC 5767274. PMID 29375266.

[88] Ho, Tin Kam (1995). Random Decision Forests (PDF). Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282. Archived from the original (PDF) on 17 April 2016. Retrieved 5 June 2016.

[89] Ho TK (1998). "The Random Subspace Method for Constructing Decision Forests" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 20 (8): 832–844. doi:10.1109/34.709601.

[90] Hastie, Trevor; Tibshirani, Robert; Friedman, Jerome (2008). The Elements of Statistical Learning (2nd ed.). Springer. ISBN 0-387-95284-5.

[91] Piryonesi S. Madeh; El-Diraby Tamer E. (2020-06-01). "Role of Data Analytics in Infrastructure Asset Management: Overcoming Data Size and Quality Problems". Journal of Transportation Engineering, Part B: Pavements. 146 (2): 04020022. doi:10.1061/JPEODX.0000175.

[92] Piryonesi, S. Madeh; El-Diraby, Tamer E. (2021-02-01). "Using Machine Learning to Examine Impact of Type of Performance Indicator on Flexible Pavement Deterioration Modeling". Journal of Infrastructure Systems. 27 (2): 04021005.

[93] Aaron van den Oord, Sander Dieleman, Benjamin Schrauwen. Deep content-based music recommendation. "https://papers.nips.cc/paper/5004-deep-content-based-music-recommendation.pdf"

[94] Young-Jun Ko, Lucas Maystre and Matthias Grossglauser, "Col-

laborative Recurrent Neural Networks for Dynamic Recommender Systems" (2016).

[95] Robin Devooght, Hugues Bersini. "Collaborative Filtering with Recurrent Neural Networks" (3-01-2017).

[96] Ali Elkahky ,Yang Song and Xiaodong He "A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems" "http://sonyis.me/paperpdf/frp1159-songA-www-2015.pdf"