**ԵՐԵՎԱՆԻ ՊԵՏԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ**

**ՏՆՏԵՍԱԳԻՏՈՒԹՅԱՆ ԵՎ ԿԱՌԱՎԱՐՄԱՆ ՖԱԿՈՒԼՏԵՏ**

**Տնտեսագիտության մեջ մաթեմատիկական մոդելավորման ամբիոն**

**ՏՎՅԱԼՆԵՐԻ ԳԻՏՈՒԹՅՈՒՆԸ ԲԻԶՆԵՍՈՒՄ**

**ԿՐԹԱԿԱՆ ԾՐԱԳԻՐ**

**ԳՐԻԳՈՐՅԱՆ ՆՈՐԱ ԱՐՄԵՆԻ**

**ՄԱԳԻՍՏՐՈՍԱԿԱՆ ԹԵԶ**

**ՀԱՃԱԽՈՐԴՆԵՐԻ ՎԱՐՔԻ ՄՈԴԵԼԱՎՈՐՈՒՄԸ ՄԵՔԵՆԱՅԱԿԱՆ ՈՒՍՈՒՑՄԱՆ ԱԼԳՈՐԻԹՄՆԵՐՈՎ**

***«Տնտեսագիտություն» մասնագիտությամբ***

***Տնտեսագիտության մագիստրոսի որակավորման աստիճանի հայցման համար***

ԵՐԵՎԱՆ 2020

**Ուսանող՝** —————

Գրիգորյան Նորա Արմենի

**Գիտական ղեկավար**՝—————

Հակոբյան Ենոք

**«Թույլատրել պաշտպանության»**

Ամբիոնի վարիչ՝————————————

Տեխ․ գիտ․ դոկտոր, պրոֆեսոր Արամ Առաքելյան

«——»—————2020թ․

**Համառոտագիր**

**Հաճախորդների վարքի մոդելավորումը մեքենայական ուսուցման ալգորիթմներով**

**Consumer Behavior Modeling using Machine Learning**

Ժամանակի ընթացքում տեղեկատվական տեխնոլոգիաների զարգացմանը զուգընթաց կազմակերպությունները կարողանում են հավաքագրել մեծ քանակի տվյալներ իրենց հաճախորդների վերաբերյալ։ Այդ տվյալները հնարավորություն են տալիս հասկանալ հաճախորդների կարիքները և ցանկությունները։ Դա էլ իր հերթին հանգեցնում է կազմակերպության վաճառքների աճին որն էլ որպես կանոն ապահովում է ավելի շատ շահույթ։ Կախված նրանից, թե որքան լավ կազմակերպությունները կկարողանան ուսումնասիրել թիրախային խմբի տվյալները, այդքան ավելի արդյունավետ հնարավոր կլինի առաջարկել ապրանքներ հաճախորդին։

Կան մի շարք թեորեմներ որոնք ժամանակի ընթացքում օգտագործվել են հաճախորդների վարքի մոդելավորման համար։ Դեռ 1943 թ․-ին ի հայ եկավ Մասլոուի բուրգը, որի հիմքում ընկած էր այն գաղափարը, որ սպառողը ձգտում է բավարարել իր կարիքները առաջնորդվելով ըստ հերթականության դասավորված հինգ հիմնական կարիքներով, որոնք են ։ Ֆիզիոլոգիական կարիքներ, անվտանգություն, սեր, պատիվ և ինքնաիրականացում։

Բացի Մասլոուի բուրգից, առաջ եկան նաև այլ տեսությունների ինչպիսիք են իմպուլսիվ գնումների տեսությունը, որի հիմքում ընկած է այն գաղափարը, որ գնումների մեծ մասը կատարվում են ազդակների շնորհիվ, որոնք մենք ստանում ենք արտաքին աշխարհից։ Այդպիսի ազդակների օրինակներ են անընդհատ կրկնվող գովազդները կամ օրինակ գարեջրի կողքին դրված ընդեղենը, որը մարդուն հիշեցնում է վերջինս ևս գնելու մասին։

Կորոնովիրուսային հիվանդությամբ պայմանավորված մարդիկ ավելի շատ սկսեցին գնումներ կատարել առցանց սուպերմարկեդային ցանցերից կամ օնլայն խանութներից, որն էլ հնարավորություն տվել կազմակերպություններին հավաքագրել մեծ քանակությամբ տվյալներ իրենց հաճախորդների վարքը հասկանալու և մոդելավորելու համար։ Ուստի նախկինում ստեղծված տեսություններին զուգընթաց առաջ եկավ նաև արհեստական բանակությամբ և մեքենայական ուսուցման ալգորիթմներով հաճախորդների վարքի մոդելավորումը, որն էլ մենք կքննարկենք այս աշխատանքի շրջանականերում։

Հաճախորդների վարքի ուսումնասիրման համար հիմնականում օգտագործվում են դասակարգման ալգորթիմները, հաճախորդների շղթայման ալգորիթմները, սեգմենտացումը, ՌՖՄ-ը և ռեկամենդացիոն համակարգերը։

Դասակարգման ալգորիթմները ինչպիսիք են Decision Tree , Clustering օգտագործվում են հաճախորդների դասակարգման և խմբավորման համար։ Հաճախորդների շղթայման ալգորիթմը օգտագործվում է հասկանալու համար թե ինչ բնութագրեր ունեն կազմակերպության համար ավելի բարենպաստ հաճախորդները։ ՌՖՄ մեթոդը բաղացած է երեք կոմպոնենտից, առաջինը recency նշանակում է թե վերջին անգամ հաճախորդը երբ է գնում կատարել , երկրորդ կոմպոնենտը իրենից ներկայացնում է frequency (հաճախություն) , որը ցույց է տալիս հաճախորդի խանութ այցելելու հաճախությունը։ Վերջինը դա Monetary ցույց է տալիս հաճախորդի կողմից ծախսած գումարի չափը։

Ռեկամենդացիոն համակարգերը իրենց լայն տարածումը գտան հատկապես վերջին շրջանում առցանց խանութների զարգացմանը զուգընթաց, ինչպիսին է Amazon-ը, որտեղ լավ ռեկամենդացիոն համակարգերը հնարավորություն են տալիս հաճախորդին առաջարկել իր սիրելի ապրանքները կամ այն ապրանքները որոնք ամենայն հավանականությամբ հաճախորդը կհավանի և կգնի։ Ռեկամենդացիոն համակարգերի մեկ այլ կիռարումը դա ֆիլմերի դիտման հարթակներն են ինչպես Netflix-ը, որն հնարավորություն է տալիս օգատիրոջը դիտել այնպիսի ֆիլմեր , որ նա ամենայն հավանականությամբ կհավանի , ավելի շատ ժամանակ ծախսել յադ հարթակում , զգալ բավարարված և երկարացնել տվյալ հարթակի իր բաժանորդագրությունը։

Ուսումնասիրելով այս բոլոր տեսակի մոդելները, ընտրությունը կատարվել օգտագործել ռեկամենդացիոն համակարգերը, քանի որ մեր տվյալները մեծաքանակ են և դրանք հնարավորություն են տալիս մեզ կատարել նման ուսումնասիրություն և մոդելավորել մեր հաճախորդների վարքը ռեկամենադացիոն համակարգերով։ Ճիշտ է ռեկամենդացիոն համակարգերը ունեն լայն տարածում , դեռ մի շարք խնդիրներ լուծված չեն։ Այս աշխատանքի շրջանականերում, մենք կփորձենք ուսումնասիրել տվյալները, կառուցել ռեկամենդացիոն համակարգ հենց մեռ դեպքի և մեր տվյալների համար, փորձելով լուծել առկա խնդիրները։

Ամենամեծ խնդիրներից մեկը որ առկա է ռեկամենդացիոն համակարգերում, դա նոր հաճախորդին ապրանքներ առաջարկելու դժվարությունն է պատմական տվյալների բացակայության պատճառներով։ Մեկ այլ խդնիր, որը սերտ կապված էր վերջինիս հետ դա, հին հաճախորդին նոր ապրանքներն առաջարկելն է, որոնք նա ամենայն հավանականությամբ կգնի։ Այսին այնպիսի ապրանքներ, որոնք դուրս են մեր ունեցած պատմական տվյալների ապրանքների կոնտեքստից բայց այնուամենայնիվ հնարավոր է հետաքրքրեն հաճախորդին։ Մյուս խնդիրը, որը նաև առկա է մեր տվյալներում դա տվյալների մեծ մասի բացակայությունն է, այսինքն հաճախորդները հասանելի հարյուր հազարավոր ապրանքներից կարող են գնել իրենց հետաքրքրող մի քանի հազարը և մնացած ապրանքների և հաճախորդի միջև եղած կապը մնում է անհայտ։ Լուծելով այս արդի խնդիրները մեր տվյալների համար , մենք հնարավորություն կունենանք սուպերմարկետի այս հաճախորդներին շատ բարձր ճշգրտությամբ առաջարկել այնպիսի ապրանքներ, որոնք նրանք ամենանայն հավանականությամբ կգնեն։

Ուսումնասիրելով դասական ռեկամենդացիոն համակարգերը , ինչպես նաև շուկայում առկա նորարարական մոդելները, այս աշխատանքի շրջանականերում կառուցվել է մի մոդել, որը հիմնված է ապրանքների միջև եղած կապերի վրա։ Այդ կապերը հնարավոր է եղել գտնել, քանի որ մեր տվյալներում է առկա է եղել յուրաքանչյուր հաճախորդի համար իր գնման կտրոնը և այդ կտրոնում հնարավոր էր տեսնել, թե ինչ ապրանքներ են գնվել միասին և ինչ քանակությամբ։ Առանձնացնելով միայն ամենահաճախ այցելող հաճախորդներին և այդ հաճախորդների համար առանձնացնելով ամենահաճախ գնվող ապրանքները, հնարավոր եղավ լուծել պակաս տվյալների խնդիրը մեր օրինակի դեպքում։

Թեստավորելով մեր մոդելը այն տվյալների խմբի վրա, որը մոդելը հնարավորություն չի ունեցել տեսնելու մենք կարողացել ենք ապրանքների համար 83% ճշտությամբ կանխագուշակել երեսուն <<ամենամոտ>> համարվող ապրանքները։ Այսինքն ներմուծելով ցանկացած ապրանքի կոդ, մոդելը վերադարձնում է այդ ապրանքին <<ամենամոտը>> համարվող երեսուն ապրանք։ Ամենամոտի գաղափարը կարելի է հասկանալ այսպես․ Այս ապրանքները միասին շատ հաճախ հայնտվել են միևնույն կտրոնի գնման ցուցակում կամ նման հաճախորդներ հաճախ նախընտրել են այդ մոտ ապրանքները։

**Բովանդակություն**

**Բովանդակություն**

[**Ներածություն** 8](#_Toc100563853)

[**Chapter 1: Characteristics of consumer behavior modeling** 9](#_Toc100563854)

[**Chapter 1.1: Consumer behavior modeling and Machine Learning algorithms** 9](#_Toc100563855)

[**Chapter 1.2: Models for Consumer Behavior task: Churn modeling, segmentation, recommendation systems** 14](#_Toc100563856)

[**Chapter 1.3: Recommendation Systems**. 22](#_Toc100563857)

[**Chapter 2: Modeling recommendation systems based on consumer behavior** 30](#_Toc100563858)

[**Chapter 2.1: Models used for recommendation systems** 30](#_Toc100563859)

[**Chapter 2.2: Top Used recommendation systems in the market** 50](#_Toc100563860)

[**Chapter 2.3: Tools for recommender systems** 56](#_Toc100563861)

[**Chapter 3: Product recommendation system and its implementation for retail sector** 65](#_Toc100563862)

[**Conclusion** 70](#_Toc100563863)

[**References** 71](#_Toc100563864)

# **Ներածություն**

Ներկայումս տվյալագիտությունը գտնում է իր լայն տարածումը բիզնեսի հատկապես հաճախորդների վարքի մոդելավորման խնդրում։ Հետևյալ աշխատանքում ուսումնասիրելու ենք սուպերմարկետներից մեկի հաճախորդների վարքը և փորձելու ենք մոդելի օգնաթյամբ առաջարկել լուծում , որի շրջանականերում հնարավոր կլինի տվյալ հաճածորդին առաջարկել այնպիսի ապրանքներ, որ նա ամենայն հավանականությամբ կցանականա գնել։ Հենվելով հաճախորդի մասին ունեցած տվյալների վրա, ինչպիսիք են գնումների զամբյուղը, ապրանքները և դրանց քանակը և գինը։ Տվյալ խնդրի լուծման կիռարությունը հնարավորություն կտա կազմակերպություններին ավելացնել իրենց շահույթը , իսկ հաճախորդներին էլ խնայել ժամանակ և միանգամից կարճ ժամանակում գտնել իրենց անհրաժեշտ ապրանքնեը։ Տվյալ ուսումնսիրության համար օգտագործվելու են մեքենայական ուսուցման մոդելներ ինչպիսիք ենք ռեկամենդացիոն համակարգերը։

# **Chapter 1: Characteristics of consumer behavior modeling**

## **Chapter 1.1: Consumer behavior modeling and Machine Learning algorithms**

Customer behavior analysis is very important for an enterprise. It helps to companies understand what the customer wants and needs. Hence, the company can improve the service or offer a suitable product to the customer. Thanks to customer behavior analysis, the company can increase their sales and be more successful in business. Marketers may spend significant time trying to parse the thoughts, patterns, and behaviors of consumers. The better they understand their target audience, the more they can cater to that audience’s wants and needs. Over the years, many people have invented theories to try and streamline what they believe explains these behaviors.

According to the theory of reasoned action, consumers act on behaviors that they believe will create or receive a particular outcome, familiar or otherwise. As such, rational decision-making is the chief element of what drives consumers to make purchases.

In 1943, the broader psychological community felt the impact of Abraham Maslow’s hierarchy of needs: a theory that insists that individuals act to satisfy and fulfill their needs based on a system of five priorities of increasing importance — physiological survival, safety, love, esteem, and self-actualization.

In contrast to the focus on rational action found in most other theories of consumer behavior, Hawkins Stern put its focus on impulse behavior. It’s Stern’s argument that the impulse to purchase was only one-half of average consumer behavior, fitting neatly beside tendencies toward more rational purchasing decisions. These impulse decisions are influenced mostly by external stimuli like walking past a convincing advertisement and possess very little relationship to traditional decision-making habits.

Impulse buying exists on four levels of the Stern philosophy. The first level is the quick, pure impulse purchase, like making a last-minute purchase on the way out of a grocery or hardware store. The second level is known as the “reminded” impulse purchase, which makes associations between one product and another. For example, placing chips and salsa in the same aisle, so if you’re planning to buy one, you’re reminded you may want the other.

The third level is the suggested impulse purchase, such as tacking on a warranty offer as you purchase electronics or power tools. The fourth level is the planned impulse decision, which is deliberate in that consumers know they want to buy a type of product, but just aren’t sure of the specifics.

We can all agree that COVID has completely changed customer behavior. More people are buying digitally, more people are stuck at home, and more people are investing in personal comforts over trips/vacations. Nowadays, digital transformation affects all activities of enterprises. Data of products and customers can be collected via information systems. These data may be used to understand more deeply about the customer's intention by using tools in data science.

Any product or service, no matter how good it is in terms of quality, if not aligned with the customer’s needs and desires, is a fiasco. Not only geographic location, but also culture, religion, nationality, and environment influence customer behavior. AI tools mine data from social media and news to past sales and reviews to tell what the customers are expecting or on which goods they are ready to spend the extra bucks. These tools also account for the economic conditions and spending power of the customers. The best thing about these tools is that they are dependable as it is proven that they predict the future demand and supply with the highest accuracy level. The companies can capitalize on these valuable insights to offer personalized goods and services to the targeted regions.

Besides accurately predicting customer behavior, AI tools are also beneficial in devising efficacious marketing strategies. The online data in the form of past reviews, online searches, and the number of views is gold for marketers. It is unthinkable for any business that wants to remain competitive in the business world, to not leverage the powers of AI for devising its marketing strategy. With the help of AI, marketers can determine which mode of marketing received more engagement from the customers. Based on this, marketers can choose that medium for future advertisements to generate more sales.

Social media is the greatest tool to analyze customers' sentiments regarding goods and services. Sentiment analysis uses text analysis techniques to decipher customers' emotions (positive, negative, and neutral) towards specific goods and services. For example, AI tools can analyze 10,000+ online reviews about your product to help you determine if customers are happy with the quality and price of your product or not. Interpreting people's emotions is essential for the success of the business. No business can survive without learning from unhappy customers. As the founder of Microsoft Bill Gates said, “Your most unhappy customers are your greatest source of learning”. Customers not only post reviews but discuss each aspect of the products and services, from quality to price and customer service, on social media. AI tools can analyze this content from social media to divulge customers' sentiments about your goods and services. Besides this, these tools also help to determine the customer's expectations from the business. Based on these actionable insights’ businesses can take imperative decisions to improve their quality, affordability, and customer service in the future.

Purchase decision process can be described with five stages. The first stage is problem recognition where consumer recognizes a problem or a need. The second stage is search for information via heightened attention of consumer Modelling In-Store Consumer Behavior Using Machine Learning 125 towards information about a certain product, which can even resolve in actual proactive search for information. The third stage represents the evaluation of alternatives, which usually involves a comparison between various options and features based on the models of the expected value and beliefs. In the fourth stage of the purchase decision process a provider, place, time, value, type and quantity of the selected product or service are determined. The fifth and final stage describes the post purchase use, behavior and actions.

We distinguish between three different types of purchase decisions. They differ in value and frequency of purchase, covering different intensity levels of involvement and time invested in the purchase decision: (i) routine response behavior (for frequently purchased, low involvement products and services), (ii) limited decision-making (unfamiliar brand choices in the known category of products and services), and (iii) extensive decision-making (high involvement, high value and low frequency of purchasing).

There is no single customer behavior model. In fact, there are many. Here are the *ten* most popular customer behavior models:

1. Pavlovian Model

The Pavlovian theory refers to a learning procedure that pairs a stimulus with a conditioned response. For example, the word ‘sale’ can generate the urge to shop for many people.

2. Economic Model

Here, the central theme is the innate desire of consumers to make the maximum gains while spending the minimum possible amount. The model takes into account homogenous buying patterns, such as when the price of a product is less, consumers tend to buy more of that product.

3. Input, Process, Output Model

In this simple model of consumer behavior, the input for the customer is a brand’s marketing effort (such as product, price, etc.) and the social environment, which consists of the family, culture, etc. that influence the decision-making process of the customer.

4. Psychological Model

A. H. Maslow postulated the psychological model of customer behavior in his hierarchy of needs. This model propounds that the behavior of an individual is driven by his or her strongest need at the time. The model further says that needs have priority, and individuals first satisfy basic needs, followed by secondary needs.

5. Howarth Sheth Model

In the Howarth Sheth Model, consumer behavior is dependent on inputs in the form of Stimuli. The model also defines outputs, which are reactions to a given stimulus and end with the purchase decision. Between the inputs and outputs are the variables that affect learning. They are hypothetical in nature as they cannot be directly measured.

6. Sociological Model

This model takes into account the impact of society in the decision-making process of a buyer. For example, if a buyer belongs to an elite category that only wears a certain kind of dress, the buyer will conform to the choices of his or her society and purchase similar stuff.

7. Family Decision-Making Model

In this model, the impact of one’s family in buying-decisions is analyzed. Family decision making refers to the collective decision making by the family, even if the product is being purchased by an individual.

8. Engel-Blackwell-Kollat Model

This is a comprehensive model that interconnects four components in consumer behavior, which are information processing (exposure, attention, etc.), central control unit (personality and attitude of the consumer), decision process (problem recognition, information retention, etc.), and environmental influences (income, social class, etc.).

9. Industrial Buying Model

The industrial model of consumer behavior is influenced by organizational factors or task-oriented objectives, such as best-product quality, lowest price, and non-task objectives such as job security, promotions, personal treatment, etc.

10. Nicosia Model

The Nicosia Model focuses on the relationship between the organization and its potential customers. According to this model, the messages from an organization (like ads) influence the predisposition of a consumer towards its product or service, which may lead the consumer to find out more information about the product.

Machine learning comes in handy for this task. However, there are many models we will focus on Machine Learning models that are used in e-commerce. In recent years, the use of machine learning methods to study problems in customer behavior analysis has become more and more attractive. The problems can be considered as supervised or unsupervised models in machine learning. The used models may be clustering models, regression models or classification models, such as decision tree, random forest, support vector machine, neural network, logistic regression. Although machine learning is promising for solving customer behavior prediction problems, the number of research in this field is limited. Besides, we need to preprocess data before applying a machine learning model. Moreover, there does not exist a good method for every dataset. Hence, the study of preprocessing methods and machine learning models to improve the results for a specific dataset is necessary.

Let’s deep dive to these methods to understand the most relevant one.

## **Chapter 1.2: Models for Consumer Behavior task: Churn modeling, segmentation, recommendation systems**

Data mining is a complete process, which mines previously unknown, effective and practical information from large databases, and uses this information to make decisions or broaden knowledge. Based on data mining, machine learning algorithm is used to predict customer purchase behavior. The prediction process is shown below.

Diagram

Description automatically generatedFig 1 The Prediction Process

Decision Trees.

Decision trees are directed and acyclic tree structures which used to classify instances. A decision tree consists of a node and a directed edge. The node includes internal nodes and leaf nodes. Internal nodes are used to distinguish between different attributes or features, and leaf nodes represent different categories. For different attributes or features, leaf nodes represent different classifications. Among them, the root node has no parent node, the other nodes have and only one parent node [19], and the node without child node is called Leaf Node. Each leaf node corresponds to the value of a class identifier C, and the other internal nodes correspond to the Splitting Attribute. The core idea of decision tree is to select appropriate labels for input values, select test attributes at the decreasing speed of information entropy, process training sets with location attribute values by information gain or information gain rate, and classify unknown attribute values by estimating the probabilities of various possible results until the decision tree can train classification data effectively. Decision tree is a common classification and prediction algorithm in data mining. It is generated by repeatedly dividing data into homogeneous data groups. Its generalization ability is strong. It mainly includes two steps: starting from the root node, data points are divided into two groups according to similarity, and then each group is divided into two groups according to similarity, until the data points of leaf nodes are the same prediction category or further divided. The branch terminates when the homogeneity cannot be improved because it exceeds the minimum threshold. Finally, the termination criteria can be selected by cross validation. Decision tree has low requirement on data set. It can process both continuous data and categorized data. The algorithm complexity is not high, and it is easy to understand and implement

Cluster Analysis.

Cluster analysis is a process of dividing the research object into several classes based on similarity. The similarity between the same classes is high, while the similarity between the different classes is low. Clustering analysis belongs to unsupervised learning with simple logic and strong ability to process low-dimensional data. Classification mainly depends on the characteristics, nature and clustering purpose of the data itself. It aims to divide the samples in the data set into several disjoint subsets. The specific process is shown in Fig. 2. K-Means algorithm is a classical clustering analysis method, which distributes clustering members by average distance value. Input a data set containing n objects, randomly select k objects as clusters, select the nearest cluster according to the distance between the remaining objects and the center of each cluster, and then recalculate the average value of each cluster until the function converges, so that the similarity within the cluster is high, while the similarity between clusters is low.

Background pattern

Description automatically generatedFig 2. Clustering Process

Steps for clustering.

1st step - Pick the number of clusters, K.

2nd step - Select K random points from the data as centroids. (Fig. 3)

A picture containing bird, rain, flock, nature

Description automatically generatedFig 3. Select random centroids

3rd step - Next, the cluster assignment step. Assign each data point to the cluster centroid they are closest to. (Fig 4)

Chart, scatter chart

Description automatically generatedFig 4. Assignment Process

4th step - Centroids are moved to the average positions of the data associated with them.

Chart, scatter chart

Description automatically generatedFig 5. Centroids moved to average positions

Step 5th - Repeat steps 3 and 4 until

* Centroids of newly formed clusters do not change
* Points remain in the same cluster
* Maximum number of iterations are reached

Churn Analysis

Churn is defined in business terms as ‘when a client cancels a subscription to a service they have been using.’ A common example is people cancelling Spotify/Netflix subscriptions. So, Churn Prediction is essentially predicting which clients are most likely to cancel a subscription i.e ‘leave a company’ based on their usage of the service.

From a company point of view, it is necessary to gain this information because acquiring new customers is often arduous and costlier than retaining old ones. Hence, the insights gained from Churn Prediction helps them to focus more on the customers that are at a high risk of leaving.

Many factors influence the reasons for a customer to Churn. It may be the fact that there’s a new competitor in the market offering better prices or maybe the service they are getting has not been up to the mark, so on and so forth.

Similarly, the churn rate is the rate at which customers or clients are leaving a company within a specific period of time. A churn rate higher than a certain threshold can have both tangible and intangible effects on a company's business success. Ideally, companies like to retain as many customers as they can.

Hence, there is no correct answer as to why exactly the customer wants to churn because as you can see there are many influencing factors.

Churn analysis answers to these questions.

1. Which customers are leaving?
2. Why are they leaving?
3. Which customers are likely to churn shortly?
4. What can you do to reduce churn?

To understand the steps of churn analysis let’s consider the steps based on the dataset. Considering a dataset with 14 columns, where 13 columns represent the independent variable, while the last column is the dependent variable that contains a binary value of 1 or 0. Here, 1 refers to the case where the customer left the bank after 6 months, and 0 is the case where the customer didn't leave the bank after 6 months. This is known as a binary classification problem, where you have only two possible values for the dependent variable—in this case, a customer either leaves the bank after 6 months or doesn't. Not all columns should be kept for the analysis, so we just keep the interested columns. For the algorithm it would be better to convert the textual data into numerical one. To train our model we need to separate the predicting column from the dataset and later we can evaluate the performance of the model. However, there are many machine learning algorithms we can keep as an example the random forest. To understand how well our model perform we need 4 most important metrics accuracy, precision, f1 measure and recall. Based on these metrics we can evaluate the performance of our model.

RFM

RFM (Recency, Frequency, Monetary) analysis is a behaviour-based approach grouping customers into segments. It groups the customers based on their previous purchase transactions considering the factors like

· How recently

· How often

· How much did a customer buy

RFM filters customers into various groups for the purpose of better service. It helps managers to identify potential customers to do a more profitable business.

There is a segment of customer who is the big spender but what if they purchased only once or how recently they purchased? Do they often purchase our product?

Also, it helps the company to run an effective promotional campaign for personalized service

· Recency (R): Who have purchased recently? Number of days since last purchase (least recency)

· Frequency (F): Who has purchased frequently? It means the total number of purchases. (high frequency)

Monetary Value(M): Who have high purchase amount? It means the total money customer spent (high monetary value)

After calculating recency, frequency and monetary the distribution graphs have been pictured to find the most frequent values in each part. (Fig 6)

Chart, histogram

Description automatically generatedFig 6. The distribution of each component

Let’s start with the segmentation. At first, we’ll assign each customer a specific score for their individual recency, frequency, and monetary value. Then, we’ll aggregate those individual scores and get a combined segmentation score. It will be like college grades. Your individual subject marks are converted into subject grades, and later by combining individual grades a final grade is computed.

We’re going to divide our customers into three equal sections (33% in each section) and assign scores from 1 to 3 (best to worst) to each section.

For recency, we’ll assign a score of “1” to the customers who have purchased recently (first 33%), score “2” to the mid group, and “3” to the third group (customers who last purchased long ago). Since customers who have purchased recently are more likely to do the business again, we are assigning better scores to them.

But for frequency and monetary, we’ll assign a score “1” to the last 33% of customers, to those who shop more frequently and spend more. And assign “3” to the first 33% of customers who shop less frequently and spend less. (looking to the graph, the first 33% of customers are near to the 0 so their frequency is low that’s why they get the point 3)

In the end, we’ll get a segmentation grade ranging from “111” to “333” (best to worst), And an aggregated scores ranging from 3 to 9. (Fig 7)

Table

Description automatically generatedFig 7.

We can finalize to get the total number like 6 and can analyze the way how we get that summarized 6 value (1+1+4, 2+2+2, 1+2+3) and so on. (Fig 8)

Chart, bar chart

Description automatically generatedFig 8. The scores in RFM model

Now depending on business requirements customers divided to these groups.

Text

Description automatically generated

To prove our approach regarding the platinum customers, gold and silver we can go through the graphs grouped by the customer segments and their characteristics.

Chart, scatter chart

Description automatically generated

As we can see Platinum customers have the largest numbers of monetary for the recent shopping(Fig 9), so we can conclude that not only they are loyal customers and frequently visit the shop also spent the huge amount of money. While the silver customers don’t have any huge number of monetary and equal distributed recency.

## **Chapter 1.3: Recommendation Systems**.

While the widespread use of the Internet and increasing data storage capabilities make it easy to access large volumes of data, it becomes harder to find relevant, engaging, and useful  
content for daily computer users due to information overload.  
Over the last few decades, there has been a significant amount of research on computer  
applications that can discover tailored appropriate content. Recommender systems are one of  
those applications that can filter information in a personalized manner (Schafer et al. 2001).  
Recommender systems produce suggestions and recommendations to assist their users  
in many decision-making processes. With the help of the recommender systems, users are  
more likely to access appropriate products and services such as movies, books, music, food,  
hotels, and restaurants.  
In a typical recommender system, the recommendation problem is twofold, i.e., (i) esti-  
mating a prediction for an individual item or (ii) ranking items by prediction (Sarwar et al.  
2001). While the former process is triggered by the user and focuses on precisely predicting  
how much the user will like the item in question, the latter process is provided by the rec-  
ommendation engine itself and offers an ordered top-N list of items that the user might like.  
Based on the recommendation approach, the recommender systems are classified into three  
major categories (Adomavicius and Tuzhilin 2005):  
CF recommender systems produce recommendations to its users based on inclinations  
of other users with similar tastes.  
Content-based recommender systems generate recommendations based on similarities of  
new items to those that the user liked in the past by exploiting the descriptive character-  
istics of items.  
Hybrid recommender systems utilize multiple approaches together, and they overcome  
disadvantages of certain approaches by exploiting compensations of the other.

Goals of recommendation systems.

1. First problem that recommendation system deals with, is a rating prediction. It’s assuming that training data has the information regarding user preference for items. For m users and n items, this is an incomplete matrix, where we have data from training set and using it to predict missing ratings.
2. In practice we need to recommend the user the top-k items rather just predicting the rating. The determination of top-k items is more common than determination of top-k users.

The main goal of recommendation system is to raise the profit of the business by recommending right items to the user.

Basic models of recommendation systems.

Basic recommendation systems deal with two kinds of data. 1) User -item interactions such a user behavior and buying or purchasing. 2) The attribute information about users and items such as textual data. Models that used the first type of data are categorized as collaborative filtering methods, whereas the models that are using the second type of data are categorized as content-based recommender methods. In knowledge-based recommender systems are relies on explicitly specific user requirements. Another recommendation systems called hybrid recommendation systems used the two approaches of the recommendation systems. Now let’s dig deep to each recommendation to understand the main idea of each type.

Collaborative filtering.

Collaborative filtering uses the power of rating given by users to make recommendations. The principal difficulty for the method is that the matrix of the ratings is sparse. For instance, in movie recommendation, we could see that each user watched small number of movies from the large universe of available movies. The specified rating is ‘observed ratings and unspecified ratings are ‘unobserved ratings’.

The basic idea of collaborative filtering is that the unobserved ratings can be input since they are highly correlated with observed ratings. Considering two users, Alice, and Bob, we could see that if their ratings are mostly similar, we can input the unobserved rating if the item rating exist for the one of them.

There are two types of methods that are commonly used in collaborative filtering. Memory based methods and model-based methods.

1. Memory based model are also referred to the neighborhood based collaborative filtering algorithms.

User-based collaborative filtering. The basic approach of this method is to find the like-minded users for the target user A and use this group of users to predict the unobserved ratings for the target user A by calculating the weighted average of the observed rating of the link minded users. Therefore, if Alice and Bob have rated movies in a similar way in the past, then one can use Alice’s observed ratings on the movie *Terminator* to predict Bob’s unobserved ratings on this movie. In general, the *k* most similar users to Bob can be used to make rating predictions for Bob.

Item-based collaborative filtering. To make predictions regarding the rating for item B by user A, the first step would be to determine the sample S which will include similar items to item B. So, we use this item set S to predict rating for the target item B.

Advantages of this approach are the explanation and easy implementation. However, this approach doesn’t work well with spars matrices. Nevertheless, the lack of coverage is often not an issue, when only the top-*k* items are required

2.Model based methods. In case of model-based recommendation systems machine learning models are used in predictive framework. In cases where the model is parameterized, the parameters of this model are learned within the context of an optimization framework. Some examples of such model-based methods include decision trees, rule-based models.

Missing Values.

As we have already discussed recommendation systems have the problem regarding missing values.

Collaborative filtering can be viewed as a generalization of regression and classification methods, where the dependent variable can be viewed as a attribute with a missing values. In case of classification and regression methods, only the dependent column is allowed to have a missing value, meanwhile in collaborative filtering any column is allowed to have missing values. In collaborative filtering we don’t speak about training and testing rows, but we can talk about training and testing entries. Since missing values can be in any row. The illustration of this paradigm can be represented by this picture, to show difference regarding missing values and train test datasets.

Chart, box and whisker chart

Description automatically generated

Content based recommendation systems.

In content-based recommendation systems, the description of the item is used to make a prediction. In this method the rating and buying behavior taken from the user are used with the content (descriptions) to do our recommendations. Content based recommendations are often used for the new customers for whom the rating or other information isn’t available.

However, content-based recommendation also has their disadvantages. The item has its own specific key words for the description, and if the user never used that item the method would not recommend that item based on the keyword’s absence in the description.

Second disadvantage of content-based recommendation system is that this recommendation works well with item recommendation and doesn’t work with the user recommendation. This problem occurs since, for the user recommendation we need to have the whole ratings for the target user, and it is important to have the whole list for the target user ratings.

Knowledge based recommendation systems

The knowledge-based recommendation systems are highly related to the items that are not purchased very often. For example, automobiles, real estate, tourism requests. As we are dealing with rarely purchased items it is hard to have rating for every combination of the requirements, and it also includes the cold start problem. Furthermore, the requirements regarding this rarely purchased items can change over the time. For instance, when we are considering the case of automobiles, each customer is interested in different properties of automobiles like engine, color, model etc. That’s it would be impossible to relate some specific rating with these huge number combinations.

In these cases, knowledge-based recommendation system can be used. In collaborative filtering or content-based recommendation we used the historical data of the user buying behavior or the ratings. In case of knowledge-based recommendation system users explicitly specify wat they want.

Hybrid and ensemble-based recommendation systems

The methods that have been mentioned above have been used in different cases. For example collaborative filtering is used in cases where we have historical data of the users and their rating, while content based recommendation are used when we have also descriptions of the products. When we have the huge number of attributes the ensemble-based models can be used to combine different aspect in one to achieve the most efficient numbers.

Since, during our analysis we deal with the supermarket data. It would be useful to concentrate on collaborative filtering, content-based recommendations and if it would be applicable also on the hybrid recommendations. The main approach will be collaborative filtering and we need to deep dive to understand every aspect of this method.

Challenges for recommender systems

Cold-start problem   
Any recommender system can face cold start problem with inclusion of new user, an item or a system. Less information is available with a new user registration or a new item inclusion which makes it difficult to recommend an item to a user. Collaborative filtering can’t make recommendation effectively in case of new user and new item problem. However, content-based recommendation system is the answer for this problem as it does not rely on item rating. Other solution is to identify visitors who are only there to browse item. For example, if user explore items from a pin to sofa-set within short time then he is there for browsing only and do not consider his clicking   
history. Ramp up problem is also similar to cold start.

Scalability problem   
Scalability is ability of system to handle growing amount of information in elegant manner. In the age of internet, enormous growth in information is encountered and it is a big challenge for recommender system to handle this continuously growing demand. Computation complexity of recommending products is proportional to the number of active user and items. In collaborative filtering computation grow exponentially and system may steer towards inaccurate results due to scalability problem. Some approximation mechanisms are proposed to handle this problem. But most of the time approximation techniques result into accuracy reduction. Scalability problems   
may also be resolved by distribution of users.

Sparsity problem   
Sparsity is one of the most important problem in recommendation systems as data sparsity affect the quality of recommendation. Generally, the number of ratings already recorded for an item is usually very less compared to the ratings that are required to be predicted. Such behavior of sparse rating occurs when most of the users do not provide rating. As a consequence, there can be items rated “good” by few users but may not be included in the recommender list.   
One way to overcome this to find user similarity from demographic segment along with the usual ratings. Scalability problems may be resolved by distribution of users.

Privacy reservation problem   
Recommender systems needs information of the user population in order to perform recommendations. Providing information actively increases accuracy of recommendation, but that increases the risk of exposure of user’s personal information. This information can be location, age, or any other critical information which raise the question of security, reliability and confidentiality. Many recommender systems provide privacy and security by using privacy-preserving   
algorithms [32] or dataset [33].

Over fitting problem   
Recommender systems restrict the item recommendation which resembles to those already defined in their profile. Sometimes user can explicitly provide his interest and system provides recommendation. Generally, user’s interest is dynamic in nature while the system prevents user from   
discovering other more options and narrow down the recommendation list. However, this can be diversified to make the recommendation list better. Algorithms like genetic algorithms are proposed which have a set of wide range of alternatives for recommendation.

Long tail problem   
Items which have low rating or no rating introduce long tail in the system. In that case the system recommends only top items and avoids the newly added items or those items which   
has low rating. Nowadays, most of the recommendation systems suggest only popular items and ignore items with less rating or newly introduced. This type of characteristic narrow down the recommendations which makes hard to use the system for consumers. Eliminating long tail problem is one of the main concern for improving performance of recommendation system.

# **Chapter 2: Modeling recommendation systems based on consumer behavior**

## **Chapter 2.1: Models used for recommendation systems**

As we have discussed above, there were two types of collaborative filtering user based collaborative filtering and item based collaborative filtering.

The user – item matrix that is used for collaborative filtering is a matrix with m users and n items. The rating for the user for item is denoted by . Only the small amount of rating is specified in the matrix. As discussed above the specified rating denoted as the training set and the not specified ratings denoted as the test set.

The ratings can be categorized as continues, interval based, ordinal, binary and unary.

*Continues ratings*. An example can be an engine where user can give any rating in specific interval. It can take any value between -10 and 10.

*Interval based ratings.* Interval base ratings are often drawn from 5 point or 7 point scale. It can be numerical integer values from 1 to 5 or from -2 to 2.

*Ordinal ratings.* This category of ratings are similar to interval based rating, except the part of the rating order. In ordinal ratings we have categorical values like ‘Strongly Disagree’ , ‘Disagree’ , ‘Neutral’, ‘ Agree’ and ‘ Strongly Agree’. A major difference from interval-based ratings is that it is not assumed that the difference between any pair of adjacent ratings values is the same.

*Binary ratings.* In case of binary ratings, we have only two options, one represents the negative feedback and the other one represents the positive feedback. Any example of like or dislike can be considered as binary rating.

Unary ratings. In such cases the system allows the customer to specify the positive feedback, but there is no mechanism to specify the negative feedback. In real world example of such rating can be the Facebook where the only option was a like and there is no option for the dislike.

In real world we have the property that only the small number of items are often rated. These items are called popular items, while vast majority of items rated rarely. It is obvious that this will lead to very skewed distribution of underlying ratings.

This distribution issues are a very large opportunity for the business and needed to pay attention.

Graphical user interface, application, Word

Description automatically generated

1. Sometimes, the frequently rated items bring the low revenue to the business, while the items that are rated rarely bring the higher revenue. In such cases maybe it would be helpful to suggest the items with rare ratings.
2. As we have items that are rated rarely it also affects on the accuracy of the model. Many models tend to recommend the items with the higher frequency. This would also can cause the customer to become bored, since it would continue to recommend the same frequently rated items.

User based neighborhood models

Since we have users and their ratings specified for some items, it is hard but not impossible to find similarities among the users based their specified rating. For example, if we have users A and B and their specified rating set it would be meaningful to find their intersections. and , so the . Based on this we can keep 1st and 5th rating to understand the similarity between the users. The on way to do that is using the Pearson correlation coefficient. For the computation Sim(A,B) the first step needed to be done is computing the mean rating for each user using the specified ratings.

Pearson correlation coefficient would be.

Sim(A,B) = Pearson(A,B) =

Pearson coefficient is computed for the target user and for other users, the highest Pearson correlation users would be classified as top - k users. Since, for each user intersection of ratings will be different it is meaningful to calculate different Pearson correlations for each user.

The problem related to this approach is that the one user can rate all the items positively and other items negatively, so to solve this problem we need to mean centered the rating before any computation. The mean centered rating of a user u for item j is defined by the raw rating and the mean rating difference.

Than, the mean rating is used to provide prediction for original rating of target user u and item j.

Similarity function that is used to understand the similarities between users can vary and one of them is cosine similarity function that is based on raw rating rather than on mean centered ratings.

Graphical user interface, application, Word

Description automatically generated

Besides the variants of similarity functions there also variants of the prediction function. Instead of using mean centered rating one can also use the z score. The main idea of using the z score is dividing the by the .

Chart

Description automatically generated with medium confidence

A picture containing watch, clock

Description automatically generated

Let denote the set of top k similar users of target user u for which the ratings of item j have been observed.

Text, letter

Description automatically generated

Item based neighborhood models.

In case of item-based neighborhood models the similarities are computed for items rather than users. To do that for each column, each row of ratings matrix centered to a mean of zero. As in the case of users each average item rating is calculated from the ratings of that item. In case of users we have been using the intersection of item rating for similar users, in case of items we find the users that have rated the particular item. For example wesee that the item has been rated from the first, third and fourth user so we have

Than the adjusted cosine similarity between the items i and j defined as follows.

A picture containing shape

Description automatically generated

Consider a situation where you need to determine the rating of user u target item t. The first step is to determine the top k items that are most similar to item t according to the foregoing Adjust cosine similarity.

Text

Description automatically generated with medium confidence

Clustering based methods are considered weaker, since all users in the cluster would be recommended the same thing. Meanwhile it would be helpful to use clustering-based methods at the first stage, when we have a big dataset and then for shrinking the selection of relevant neighbors in collaborative filtering algorithms.

Sometimes when we have a large number of customers and large number of items, the dimensionality problem accuracies. To avoid that problem we are using different techniques to map high dimensional space into a reduced matrix. One way of to do that is matrix factorization when the matrix D represents by the multiplication of different matrices.

In SVD technique ( singular value decomposition ) mxn matrics R is factored into three matrices.

R = UxSxV

Where S is a diagonal matrix and U and V are orthogonal matrices. All values of vector S are positive and ordered in descending order.

By SVD we can solve the two problems related to the recommendation systems. First of all it helps to find the latent relationships of customer and their products, to help us calculate the estimated likelihood of purchasing the product. And the Second it is used to make a low dimensional representation of the original customer-product matrix.

Model based collaborative filtering

Model-based recommender systems often have a number of advantages over

neighborhood-based methods:

1. *Space-efficiency:* Typically, the size of the learned model is much smaller than the

original ratings matrix. Thus, the space requirements are often quite low. On the other

hand, a user-based neighborhood method might have *O*(*m*2) space complexity, where

*m* is the number of users. An item-based method will have *O*(*n*2) space complexity.

2. *Training speed and prediction speed:* One problem with neighborhood-based methods

is that the pre-processing stage is quadratic in either the number of users or the

number of items. Model-based systems are usually much faster in the preprocessing

phase of constructing the trained model. In most cases, the compact and summarized

model can be used to make predictions efficiently.

3. *Avoiding overfitting: Overfitting* is a serious problem in many machine learning algorithms,

in which the prediction is overly influenced by random artifacts in the data.

This problem is also encountered in classification and regression models. The summarization

approach of model-based methods can often help in avoiding overfitting.

Furthermore, *regularization* methods can be used to make these models robust.

The kNN is used as a prediction algorithm for the RS, and there are several variants of kNN approaches such as Basic kNN, kNN taking into account each user's mean ratings, and kNN taking into account each user's z-score normalisation. Table 2 shows the formula for each variant of the kNN algorithm. To clarify, a prediction rating 𝑟𝑢,𝑖 of user 𝑢 on item 𝑖 in kNN using user's mean ratings is obtained by firstly selects the k best correlated (similar) users to user 𝑢. Then, the rating prediction for item 𝑖 is made from the weighted combination of ratings of the selected neighbours, which is measured as the weighted deviation from the mean of the neighbours (i.e., according to the degree of similarity shared with user 𝑢, the weight of each neighbour is determined) (Isinkaye et al. 2015).

Graphical user interface, text, email

Description automatically generated

Decision and Regression Trees

Decision and Regression trees are often used in data classification. The decision trees are intended for cases where the dependent variable is categorical while regression trees are designed for cases where the dependent variable is numeric. When discussing the generalization of decision trees to collaborative filtering, we will first discuss use of decision trees for classification.

The decision tree is a hierarchical division of data space using a set of hierarchical decisions known as split criteria for the division of independent variables. One-sided decision tree, one attribute is used simultaneously to make a division.

When each node in a decision tree has two children, the resulting decision tree is said to be a binary decision tree

The quality of the split can be estimated using Gini's weighted average index child knots created by splitting. If *p*1 *. . . pr* are the fractions of data records belonging to *r* different classes in a node *S*, then the Gini index *G*(*S*) of the node is defined as follows:

A picture containing text, watch

Description automatically generated

The Gini index is between 0 and և 1, and smaller values ​​are greater discriminatory force. The total Gini index of the split is equal to the weighted average Gini index of child knots. The weight of the node is determined by the number data points in it. If S1 և S2 in a binary decision are the two children of node S

tree with data records n1 և n2, respectively, then և S G Gini index of division (S1, S2)

can be assessed as follows:

A picture containing text

Description automatically generated

Diagram

Description automatically generated

The Gini index is used for selecting the appropriate attribute to use for performing the

split at a given level of the tree.

The approach is hierarchical, from top to bottom each node contains only records of data belonging to that class. It is possible to stop tree growth early when the minimum number of node records belongs to a

specific class. Such a node is called a terrestrial node, it is labeled dominant the class record of that node.

Extending Decision Trees to Collaborative Filtering

Decision Trees main challenge of expanding collaborative filtering is to anticipate

Records and the observed records are not explicitly separated by a column as a feature and class variables. Moreover, the rating matrix is ​​very sparsely populated where most of the records are missing. This poses challenges for hierarchical distribution training data during the tree construction phase. Moreover, dependent and independent variables (elements) are not clearly defined in the collaborative filter, question arise, what point

should be predicted by the decision tree?

Consider the m × n rating matrix R with m users and n items. An individual decision tree must be constructed, fixing each feature (element) depending on the other features as independent. Hence the decision number

constructed trees are exactly equal to n number of attributes (elements). Predicting the rating of a particular product for the user, the corresponding decision tree is used for forecasting.

The lack of independent features is more difficult to resolve. Consider the case when a certain material (say, a specific film) is used as a division character. Users with rating is below the threshold are assigned to one branch of the organization tree, while users whose ratings exceed the threshold are assigned to the other

Branch. Rating matrices are sparce, most users will not have special ratings

this element. Question arise to which branch these ratings should be assigned?

Assuming the scenario where it is necessary to predict the rating of point j. From the very beginning the matrix of scores m × (n - 1) is converted, except for column j.

in the lower representation m × d, where d n − 1 և has all the attributes

are completely specified. Consider the scenario, where the rating of the *j* th item needs to be predicted.

At the very beginning, the *m ×* (*n −* 1) ratings matrix, excluding the *j*th column, is converted

into a lower-dimensional *m×d* representation, in which *d < n−*1 and all attributes

are fully specified.

The i-th row of the mxd matrix is ​​used as a test instance to predict the j-point rating for user i , j-th decision / regression tree is used as a model to predict the value of the corresponding rating. The first step is to use the remaining n − 1 elements (except for point j) to create a reduced d-dimensional image of the test example. This is a broader approach to combining the reduction scale with classification model is not limited to decision trees. This approach is relatively easy to use in conjunction with almost any classification model. In addition, there are methods of reducing the size used separately to predict ratings in offering systems.

b. Matrix Factorization Matrix factorization (MF) is one of the popular algorithms of the model-based CF that has obtained excellent results in the Netflix prize problem (Koren et al. 2009). It attempts to compact large databases into a single model and apply a reference mechanism to perform recommendation tasks. The MF algorithm is explicitly used to depict items and users using vectors of latent factors inferred from items' ratings (Isinkaye et al. 2015). Basically, MF tries to find a low-rank approximation to the rating matrix (Chen&Peng 2018). The fundamental assumption of the MF model is that it is possible to embed user preferences and item characteristics in the same subspace. If the latent factor vector of the user matches the latent factor vector of the item efficiently in the subspace, it is likely that the user is interested in the item (Chen&Peng 2018). MF maps users and items to a shared dimensionality subspace, based on the assumption that the rating matrix has a low rank, such that a user-item rating can be modelled as the user's inner products of the item and user latent factor vectors in that subspace (Chen&Peng 2018). A simple illustration for the MF concept is depicted in Figure 8, in which the rating matrix R (M x N) decomposed into two smaller matrices, P (M x K) and Q (K x N), where K is the number of features. Several models are built using MF; Singular value decomposition (SVD) is the most traditional one used for rating prediction (Alter et al. 2000). It has gained popularity due to its good accuracy and scalability (Chen&Peng 2018). Then an extension of the SVD model is proposed by Koren (2008) that incorporates implicit feedback to enhance prediction accuracy. Furthermore, there is another MF model proposed by Lee (1999) that factorized the rating matrix into two matrices with the property that there are no negative elements in all three matrices; this model called the Non-negative matrix factorization (NMF). Typically, the prediction rating using the MF model is given by the inner product of a user feature vector and the item feature vector (i.e., each item 𝑖 is associated with an item-factors vector, and each user 𝑢 is associated with a user-factors vector). For more details, in the SVD model, two-factor matrices are learned through stochastic gradient descent. The first is the user factors matrix P = [p1, p2, p3… pM], and the second is the item factor matrix Q = [q1, q2, q3… qK]. The prediction rating of user 𝑢 on item 𝑖 is done by taking an inner product of a user feature vector pm and the item feature vector qk as the following equation: 𝑟𝑢𝑖 ̂ = 𝑝𝑢 𝑇 𝑞𝑖 = ∑ 𝑝𝑢,𝑘 𝑞𝑘,𝑖 𝑘 𝑖=1 .

Diagram

Description automatically generated

Evaluation Metrics for the Recommender System As discussed in section 2.2, RSs have different goals that vary from the main one to the sub-goals. A particular RS's success and efficiency often depend on the RS's primary objective and the domain characteristics to which it is applied (Schröder et al. 2011). Evaluation metrics for RSs can be broadly categorized into three (Herlocker et al. 2004): predictive accuracy metrics, classification accuracy metrics, and rank accuracy metrics. a. Predictive Accuracy Metrics Predictive accuracy or rating prediction metrics answer the question of how similar a recommender's predicted ratings are to actual user ratings (Schröder et al. 2011). These metrics evaluate the RS performance by measuring the average error between the real ratings and the ratings predicted by the system. These metrics are mostly used as they are easy to compute and understand and particularly useful for evaluating tasks in which the predicting rating will be displayed to the user (Herlocker et al. 2004). As discussed in the works (Gunawardana&Shani 2009; Herlocker et al. 2004), there are three crucial prediction accuracy metrics used: Mean Absolute Error (MAE), Mean Square Error (MSE) and Root Mean Square Error (RMSE). Usually, these metrics calculate the error difference between the predicted ratings and the user's actual ratings; thus, a lower value implies a better RS performance for these metrics. Table 3 defines the formulas of the three metrics. b. Classification Accuracy Metrics Classification metrics determine the extents to which a RS makes correct or incorrect decisions about whether an item is good (i.e., good means relevant to user preferences) (Herlocker et al. 2004; Musat et al. 2013). Generally, a list of items for users is recommended by the RS. In general, they are organised horizontally or vertically, and users only care about the front parts of most items or the back parts of the items. A top-n recommendation is the name of this way of recommendations. The three most popular metrics are used to evaluate the RS recommendations' efficiency: precision, recall, and F-Measure.

Graphical user interface, text, application, table

Description automatically generated

Classification Accuracy Metrics Classification metrics determine the extents to which a RS makes correct or incorrect decisions about whether an item is good (i.e., good means relevant to user preferences) (Herlocker et al. 2004; Musat et al. 2013). Generally, a list of items for users is recommended by the RS. In general, they are organised horizontally or vertically, and users only care about the front parts of most items or the back parts of the items. A top-n recommendation is the name of this way of recommendations. The three most popular metrics are used to evaluate the RS recommendations' efficiency: precision, recall, and F-Measure.

The item set must be divided into two classes to calculate the previous metrics for RS (Gunawardana&Shani 2009; Herlocker et al. 2004): the first class determines the relevance of the item: relevant and irrelevant. The second class determines the quality of the returned items to the user selected/recommended and not selected/not recommended. The rating scale in specific datasets is not binary, so it needs to be converted.

Graphical user interface, text, application

Description automatically generated

Rank Accuracy Metrics Rank accuracy metrics measure a recommendation algorithm's ability to create a recommended order of items that matches how the same items would have been ordered by the user (Herlocker et al. 2004). Unlike the two previous accuracy metrics, this type does not seek to calculate the RS ability to accurately predict a single item's rating. Rank accuracy metrics are suitable for the RSs that provide the user with ranked recommendation lists and concern with differentiating between the "best" alternatives and just "good" alternatives of the recommended items. This type is suitable for the domains interested in ordering items and emphasizes the differences between the elements (Herlocker et al. 2004). The most famous examples of the metrics in this category include Hit Ratio and Mean Reciprocal Rank (Chen&Chen 2014). Hit Ratio at N (HR@n) is a metric for calculating how many "hits" are done in an n-sized list of ranked items. A "hit" can describe an activity done by the user, such as liking, purchasing, clicking on, or saving/favourite. HR@n measures the percentage of RS successes by calculating the average probability that the pushed item will be recommended by a top N recommender (Sarwar et al. 2001). Simply, HR@n shows whether the desired items appear on the recommendation lists (presence) and how many times they appear (frequency) (Dias&Fonseca 2013). In contrast, Mean Reciprocal Rank (MRR) is a metric using to evaluate the generated recommendation lists in terms of determining the ranking position of the target user’s choice in the recommendation list (Chen&Chen 2014). The Reciprocal Rank (RR) calculates the reciprocal of the rank at which the first relevant item was retrieved. If the relevant item was retrieved at rank 1, the RR is 1 if it was not Selected Not Selected Total Relevant 𝑁𝑟𝑠 𝑁𝑟𝑛 𝑁𝑟 Irrelevant 𝑁𝑖𝑠 𝑁𝑖𝑛 𝑁𝑖 Total 𝑁𝑠 𝑁𝑛 𝑁 0.5 if the relevant document was retrieved at rank 2 and so on. When averaged across multiple queries, the measure is called the Mean Reciprocal Rank (MRR).

DL based recommendation systems.

To better understand how DL-based recommenders work and how they usually make use of the available data, it is important to discuss a few practical examples of these types of systems. Each model discussed in this section is also implemented on the DRecPy framework, which will be discussed in chapter 4. Each model will include a brief description of what class of NNs it uses, its layer structure, what kind of activation functions it supports between layers, and the loss function

3.2.1 Deep Matrix Factorization The Deep Matrix Factorization (DMF) model is introduced by Xue et al. [2017] and it is a very simple model that uses two MLPs: one for learning user’s interests and the other to specialize on item’s relationships, therefore the model itself belongs to the Feedforward NNs class. Its intended use is for item ranking tasks, despite using a point-wise loss function. Assuming that Y ∈ R |U|×|I| represents the interaction matrix, where |U| is the total number of users and |I| the total number of items, then a prediction for the user u and item i will require the user u interaction vector (Yu,∗) and the item i interaction vector (Y∗,i). Each user interaction vector should contain the interaction value (or rating value) that the user provided to each item, and when no interaction has occurred yet between a user-item pair, then that interaction value should be set to 0. The same logic applies to computing item interaction vectors. It is also possible to set the interaction vector’ values to 1 when an interaction is found, and to 0 when none is found, but this alternative is quickly discarded in its original publication, since this option does not preserve the preference degree of a user on an item.

3.2.1.1 Layer Structure As stated before, this network actually involves the use of two separate networks, let us denote the user MLP as MLPU , and the item MLP as MLPI . The motivation for this split is to specialize each network to represent each entity in a latent low-dimensional space, maintaining its relationships (user interactions and item interactions). The model’s neural architecture is depicted in Fig. 3.1.

Diagram

Description automatically generated

The dimension of each layer of the MLPU network does not need to match the dimension of the same layer on the MLPI network, although the output dimension of the last layer of both networks must be the same, due to how the model’s predicted value is computed. Once each input vector reaches the end of network, it is transformed into a latent vector: pu = MLPU (Yu,∗) is the user latent vector and qi = MLPI (Y∗,i) is the item latent vector. The model predicted value yˆu,i is computed by taking the cosine similarity of both latent vectors, as denoted in equation 2.3. Since the cosine similarity ranges from [−1, 1], it is possible that negative predictions occur, so it is also necessary to set prediction values to a very small positive number (e.g. 1e−6) whenever they are negative (so that the loss function remains unaffected). The final predicted value is computed as follows: yˆu,i = max(cosine(pu, qi), 1e−6)

3.2.1.2 Activation Functions As described in the model’s original publication, the activation function to be used between each layer should be the ReLU activation function. This function is denoted in equation 3.1.

Text, letter

Description automatically generated

3.2.1.3 Loss Function DMF uses a point-wise loss function called (binary) cross-entropy, as shown in equation 2.38. There are actually two possible variants that completely affect the model behavior: the DMF-ce, using crossentropy and the DMF-nce, using a normalized cross-entropy loss. This final variant consists of scaling the desired value yu,i to the [0, 1] range. The goal of the optimization process is to minimize the resulting loss values by applying the backprop algorithm to get the gradients for each network and adjust their weights accordingly.

3.2.2 Collaborative Denoising Autoencoder The Collaborative Denoising Autoencoder (CDAE) was introduced by Wu et al. [2016], and it uses a modified denoising autoencoder, that incorporates an extra node on the input layer, that changes accordingly to the user for which the prediction is being made. Besides the extra user-specific node, the input layer also receives the corrupted interaction vector of that user. The corruption level is controlled by a model hyperparameter q, and this input is drawn from a conditional distribution y˜ ∼ p(˜y|y), where: p(˜yu = δyu) = 1 − q p(˜yu = 0) = q (3.2) where δ = (1−q) −1 , so that the corruption is not biased, due to some of the dimensions being overwritten with 0.

3.2.2.1 Layer Structure As this model incorporates a denoising autoencoder, it must consist on three layers only: an input layer with |I| nodes (which receives the corrupted input), a hidden layer that must have a reduced dimensionality K, where K < |I|, and then an output layer with the same dimension as the input layer. Figure 3.2 represents the model’s structure, also including the user-specific node. Note that this user node is connected to every other unit from the hidden layer, therefore we must have an additional matrix V ∈ R |U|×K, where the Vu entry corresponds to the u-th user-specific node. This user latent vector is the key difference between CDAE and a simple DAE. To compute predictions for the i-th item with respect to a given user u, we provide CDAE with the u-th user corrupted interaction vector, along with Vu, and the i-th element of the output layer corresponds to the model’s prediction for the u, i pair. For out-of-training predictions, note that the provided input vector should not be corrupted.

Diagram

Description automatically generated

3.2.2.2 Activation Functions The original publication discusses several options for the activation functions used by CDAE, such as the identity function 2.35, the sigmoid function 2.36 and the hyperbolic tangent function 3.3

A picture containing text, clock

Description automatically generated

3.2.2.3 Loss Function On the evaluation experiments presented on the CDAE publication, four variants are compared, where two distinct loss functions are used: the square loss function 2.37 and the log loss/binary cross-entropy function 2.38. It is also mentioned that there should be no reason for CDAE to not support other loss functions, provided that they are differentiable and allow for the model to adjust and improve its behavior.

3.2.3.1 Layer Structure This model consists on three components: embedding look-up, convolutional layers, and fully-connected layers. After sampling L items that a given user has interacted with before, an embeddings matrix E ∈ R L×d is computed. This matrix is used as input for two types of convolutional layers: an horizontal convolutional layer, to compute union-level patterns, and a vertical convolutional layer, to compute point-level sequential patterns. To compute the predictions, the outputs of these convolutional layers are concatenated and processed by two fully connected layers, also making use of a user embeddings vector P ∈ R d , and resulting in a prediction vector y ∈ R |I| . At prediction time, we take the T items that obtained the largest prediction values. Figure 3.3 represents the overall model architecture, separated by the three components.

Diagram

Description automatically generated

Negative feedback is introduced by defining a factor dependent on the number of target predictions T, which can be defined as T − = ηT. It is expected that the predictions of the T target items approach 1, and that the predictions of the T − negative items approximate 0.

3.2.3.2 Activation Functions There are two activation functions that must be defined for this model: for the horizontal convolutional layer, and for the feedforward layers. The authors evaluated multiple functions on their practical experiments, such as the hyperbolic tangent 3.3, identity 2.35, the sigmoid 2.36 functions, but the one that obtained better results was the relu 3.1 function. 3.2.3.3 Loss Function The loss function used by Caser is the categorical cross-entropy function 2.39. As mentioned before, each prediction over a target item in T is expected to approximate 1, and all predictions over negative items are expected to approximate 0, which makes Caser work very well with the categorical cross-entropy loss function.

## **Chapter 2.2: Top Used recommendation systems in the market**

In the market the common definition for recommendation is that it is a system that presents items to users in a relevant way.

Graphical user interface, website

Description automatically generated

Personalization system presents recommendations in a way that is relevant to the individual user.

Graphical user interface, website

Description automatically generated

Interactions can be positive (hearts, stars, likes etc.), negative (angry faces, 1 star, unfollows , returns) and explicit vs implicit. Where the explicit stands for user action that impacts their personalized experience. Implicit is all other interactions between user and items.

The recommendation system can also be interpreted as a set of top level made decisions.

To construct our recommendation system it would be helpful to investigate recommendations systems that are already in the market.

Youtube(Covington, Adams, Sargin)

YouTube has one of the largest and most advanced recommendation systems in the industry. As one of the world’s leading websites, to satisfy its customers it must recommend relevant videos. YouTube is slightly different than other services that utilize recommendation systems(i.e Netflix, Hulu, Spotify) because users upload thousands of hours of video to the platform every second. YouTube’s corpus is constantly changing, and they aren’t in control of the content being added. This creates the need for a robust model that can handle constant incoming data and will output quality recommendations in real time. The model below is the author's response to this need.

Diagram

Description automatically generated

The recommendation system they designed has two stages. The first being a neural network for candidate generation and the latter for ranking. ***Candidate generation*** network takes events from users YouTube history. This can only provide broad personalization using collaborative filtering. These users are then compared by identifiers such as the number of videos watched, demographic information and search query tokens. The ***ranking network*** operates a little differently. It assigns a score to each video using a ‘rich set of features describing the video and user’. This two-tiered system allows the system to handle millions of videos, but also scale down to individual users and provide them with meaningful content.

Stage 1: Candidate Generation

Recommendation as Classification

The authors referred to recommendation systems as an ‘extreme multiclass classification where the prediction problem becomes accurately classifying a specific video’. In this setting, embedded data is sparse data collected about the video:(i.e user, interaction, time watched, etc…) Deep neural networks are able to handle this data and use it to discriminate between videos and users. In order to train a model with millions of classes(videos), the authors use a sample of the data and then correct the sampling using weights. Using this technique they are able to gain similar accuracy with less computing power.

Model Architecture

The model is inspired by the architecture of a *continuous bag of words*language model. The neural network is fed learned high dimensional embeddings about each video which are organized in a fixed vocabulary.

In order to track user watch data, the data is transformed into varying arrays of video ids which are mapped into a dense vector representation via the embeddings from the videos mentioned earlier. The embedding information is then learned along with the other model parameters, using gradient descent backpropagation.

Managing New Data

Neural Networks have several advantages in this specific example. Using a neural network as a generalization of matrix factorization allows arbitrary data(continuous and categorical) to be added to the model easily, which this model needs because of the constantly changing corpus.

The search query history can be used by tokenizing it into unigrams and bigrams(categories specifying how many words are in the phrase). These insights are then averaged and they create a fairly dense search history that the model can then use to help predict what the user may like in the future. Demographic features of the user, which can be helpful when predicting users who haven’t watched any video- are changed into binary options[0,1].

Freshness:

Recent, relevant content is vital to YouTube as a platform, it helps keep users engaged and up to date, however, models are typically biased to making predictions from past data. To correct this, they set the age of the training data as a feature and set it very close to zero(essentially making all of the data from the same point in time) this helps the model optimize and make predictions at the latest part of the training — predicting at the tail end of the data means that it will likely predict videos that are coming next since it closest to the present, and these videos will also still have relevance to the user.

Context:

Users utilize YouTube in a variety of ways. They can watch directly from the site, or they can watch videos embedded on websites, or on other social media sites. The model uses all of this information to collect data about the user and propagates it through its network. There are several key insights with context and information that needed to be restricted from the model in order to improve performance and avoid overfitting. One of the insights was created a fixed number of inputs for each user because if there were unlimited inputs then high volume users would dominate the loss function and the model would overfit to their tendencies.

*How is this important*?

There are various types of videos and genres on YouTube. Some shows have a chronological order, meaning they will be watched in sequence -whereas music videos usually start with the most popular song by that artist and then the user diverges to the smaller niche work by that artist. If these recommendations are being influenced by high volume users, this may interrupt how a low volume user may see videos in their feed. They may miss the niche markets or may not be shown the most popular video based on the actions of higher rate users in that space.

In turn, more success was found by predicting the next video the user was going to watch, as opposed to using historical data to randomly select a video using the holdout data. That method has its flaws because it leaks future data to the model which can lead to overfitting.

Feature Experimentation:

Adding quality features to a model that is attempting to tackle such a significant problem helps improve the accuracy and scalability tremendously. Over 1million videos and 1M search tokens were embedded as decimals into bags with a maximum of 50 recent watches and 50 recent searches. The key issue is using these features to help create a temporal(time) stamp of each action of each user.

Stage 2: Ranking

The second part of the recommendation system involves ranking videos. In order to recommend quality content, YouTube needs a way to determine which content users are watching and enjoying.

Feature Engineering

The authors observed that previous interactions with a video or videos similar to it were very important when predicting recommendations. This is intuitive because if a viewer enjoys particular types of content they are likely to view many videos in that niche. They also noticed that videos coming from particular channels was also very important in deciding what to recommend next. On the opposite end of this, if the user was recommended videos, but ignored them — then the next time the page refreshed the videos will have shuffled around and changed. As a user of YouTube, I have experienced this plenty of times.

The other features were engineered in a typical machine learning fashion. Categorical variables were one-hot encoded, while continuous features were standardized(neural networks perform much better with standardized data) and learned through the model. After the initial seperation is where the model begins to gain complexity. The features were separated into univalent and multivalent groups, meaning that they either contribute a single value or multiple values respectively. The features were also classified as impression and queries. Impressions were computed every second whereas query features were only computed once per request.

Modeling Expected Watch -Time

Videos that retain the viewer's attention are usually regarded as higher quality. In order to recommend quality videos, the model needs to be trained so that it can predict how long a viewer will watch a video. By predicting how long a video will be watched, the model can then rank the video. After it ranks the video, it will be able to decide whether to recommend the video or not.

To predict the watch time, the authors used weighted logistic regression. The reason it is weighted is that only positive videos are given weight. A positive video is where a user has actually clicked on the video at all, this helps the algorithm learn only about videos that the user interacted with.

Conclusions

The authors “deep-collaborative filtering model was able to effectively assimilate many signals and model their interaction with layers of depth”, outperforming any models being used beforehand at YouTube.

Key Insights:

* Using the age of the training example as an input feature removes the inherent bias towards the past and helps the model represent the time-dependent behavior of popular videos.
* Ranking is a more classical machine learning approach, but the deep learning model outperforming classic linear prediction models.
* Logistic regression was modified by weighting the training examples with watch time for positive examples and unity for negative examples allowing odds to be learned more accurately.

YouTube, Spotify, and Netflix are major players in my life. I use all of these services for entertainment and or education daily. I rarely search for videos manually on YouTube and rely heavily on their recommendation system. Learning the intricate mathematical processes behind a seemingly simple concept was eye-opening for me. Data Science provides impacts people’s lives every day and I can’t wait to get started.

## **Chapter 2.3: Tools for recommender systems**

Scikit – Learn

What is scikit-learn?

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

What are the features?

The library is focused on modeling data. It is not focused on loading, manipulating and summarizing data. For these features, refer to NumPy and Pandas.

* Clustering: for grouping unlabeled data such as KMeans.
* Cross Validation: for estimating the performance of supervised models on unseen data.
* Supervised Models: a vast array not limited to generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

he 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
* Preprocessing, including Min-Max Normalization

Three factors conspire to make recommendations challenging for us. Firstly, our system contains a very large number of items. This makes our data very sparse. Secondly, we deal in fashion: often, the most relevant items are those from newly released collections, allowing us only a short window to gather data and provide effective recommendations. Finally, a large proportion of our users are firsttime visitors: we would like to present them with compelling recommendations even with little data. This combination of user and item cold-start makes both pure collaborative and content-based methods unsuitable for us.

To solve this problem, I use a hybrid content-collaborative model, called LightFM due to its resemblance to factorisation machines (see Section 3). In LightFM, like in a collaborative filtering model, users and items are represented as latent vectors (embeddings). However, just as in a CB model, these are entirely defined by functions (in this case, linear combinations) of embeddings of the content features that describe each product or user. For example, if the movie ‘Wizard of Oz’ is described by the following features: ‘musical fantasy’, ‘Judy Garland’, and ‘Wizard of Oz’, then its latent representation will be given by the sum of these features’ latent representations. In doing so, LightFM unites the advantages of contentbased and collaborative recommenders. In this paper, I formalise the model and present empirical results on two datasets, showing that: 1. In both cold-start and low density scenarios, LightFM performs at least as well as pure content-based models, substantially outperforming them when either (1) collaborative information is available in the training set or (2) user features are included in the model. 2. When collaborative data is abundant (warm-start, dense user-item matrix), LightFM performs at least as well as the MF model. 3. Embeddings produced by LightFM encode important semantic information about features, and can be used for related recommendation tasks such as tag recommendations.

his has several benefits for real-world recommender systems. Because LightFM works well on both dense and sparse data, it obviates the need for building and maintaining multiple specialised machine learning models for each setting. Additionally, as it can use both user and item metadata, it has the quality of being applicable in both item and user cold-start scenarios.

The structure of the LightFM model is motivated by two considerations. 1. The model must be able to learn user and item representations from interaction data: if items described as ‘ball gown and ‘pencil skirt’ are consistently all liked by users, the model must learn that ball gowns are similar to pencil skirts. 2. The model must be able to compute recommendations for new items and users. I fulfil the first requirement by using the latent representation approach. If ball gowns and pencil skirts are both liked by the same users, their embeddings will be close together; if ball gowns and biker jackets are never liked by the same users, their embeddings will be far apart. Such representations allow transfer learning to occur. If the representations for ball gowns and pencil skirts are similar, we can confidently recommend ball gowns to a new user who has so far only interacted with pencil skirts. This is over and above what pure CB models using dimensionality reduction techniques (such as latent semantic indexing, LSI) can achieve, as these only encode information given by feature co-occurrence rather than user actions. For example, suppose that all users who look at items described as aviators also look at items described as wayfarers, but the two features never describe the same item. In this case, the LSI vector for wayfarers will not be similar to the one for aviators even though collaborative information suggests it should be. I fulfil the second requirement by representing items and users as linear combinations of their content features. Because content features are known the moment a user or item enters the system, this allows recommendations to be made straight away. The resulting structure is also easy to understand. The representation for denim jacket is simply a sum of the representation of denim and the representation of jacket; the representation for a female user from the US is a sum of the representations of US and female users.

To describe the model formally, let U be the set of users, I be the set of items, F U be the set of user features, and F I the set of item features. Each user interacts with a number of items, either in a favourable way (a positive interaction), or in an unfavourable way (a negative interaction). The set of all user-item interaction pairs (u, i) ∈ U × I is the union of both positive S + and negative interactions S −. Users and items are fully described by their features. Each user u is described by a set of features fu ⊂ F U . The same holds for each item i whose features are given by fi ⊂ F I . The features are known in advance and represent user and item metadata. The model is parameterised in terms of d-dimensional user and item feature embeddings e U f and e I f for each feature f. Each feature is also described by a scalar bias term (b U f for user and b I f for item features). The latent representation of user u is given by the sum of its features’ latent vectors:

The same holds for item i.

The bias term for user u is given by the sum of the features ‘ biases.

The same holds for item i.

The model’s prediction for user u and item i is then given by the dot product of user and item representations, adjusted by user and item feature biases:

bui = f (qu · pi + bu + bi)

There is a number of functions suitable for f(·). An identity function would work well for predicting ratings; in this paper, I am interested in predicting binary data, and so after Rendle et al. [16] I choose the sigmoid function

The optimisation objective for the model consists in maximising the likelihood of the data conditional on the parameters. The likelihood is given by

L(

TensorRec

First of all, we instroduce the general ideal of tensors and go deep to the recommendation systems.

We will regard an array of numbers with more than 2 dimensions as a tensor. This is a natural extension of matrices to a higher order case. A tensor with m distinct dimensions or modes is called an m-way tensor or a tensor of order m. Without loss of generality and for the sake of simplicity we will start our considerations with a 3rd order tensors to illustrate some important concepts. We will denote tensors with calligraphic capital letters, e.g. T ∈ RM×N×K stands for a 3rd order tensor of real numbers with dimensions of sizes M, N, K. We will also use a compact form T = [tijk] M,N,K i,j,k=1 , where tijk is an element or entry at position (i, j, k), and will assume everywhere in the text the values of the tensor to be real.

Tensor fibers. A generalization of matrix rows and columns to a higher order case is called a fiber. Fiber represents a sequence of elements along a fixed mode when all but one indices are fixed. Thus, a mode-1 fiber of a tensor is equivalent to a matrix column, a mode-2 fiber of a tensor corresponds to a matrix row. A mode-3 fiber in a tensor is also called a tube. Tensor slices. Another important concept is a tensor slice. Slices can be obtained by fixing all but two indices in a tensor, thus forming a two-dimensional array, i.e. matrix. In a third order tensor there could be 3 types of slices: horizontal, lateral, and frontal, which are denoted as Ti::, T:j: , T::k respectively. Matricization. Matricization is a key term in tensor factorization techniques. This is a procedure of reshaping a tensor into a matrix. Sometimes it is also called unfolding or flattening. We will follow the definition introduced in [48]. The n-mode matricization of a tensor T ∈ RM×N×K arranges the mode-n fibers to be the columns of the resulting matrix (see Figure 2). For the 1-mode matricization T(1) the resulting matrix size is M × (NK), for the 2-mode matricization T(2) the size is N ×(MK) and the 3-mode matricization T(3) has the size K × (MN). In the general case of an m-th order tensor T ∈ R I1×I2×···×Im the n-mode matricization T(n) will have the size In × (I1I2 . . . In−1In+1 . . . Im). For the corresponding index mapping rules we refer the reader to [48]. Diagonal tensors. Another helpful concept is a diagonal tensor. A tensor T ∈ R I1×I2×···×Im is called diagonal if ti1i2...im 6= 0 only if i1 = i2 = . . . = im. This concept helps to build a connection between different kinds of tensor decompositions.

**Tensor-based models in recommender systems**

Treating data as tensor may bring new levels of flexibility and/or quality into RS models, however there are nuances that should be taken into account and treated properly. This section covers different tensorization techniques used to build advanced RS in various application domains. For all the examples we will use a unified notation (where it is possible) introduced in Section 4, hence it might look different from the notation used in the original papers. This helps to reuse some concepts within different models and build a consistent narrative throughout the text. 5.1 Personalized search and resource recommendations There is a very tight connection between personalized search and RS. Essentially, recommendations can be considered as a zero query search [6] and, in turn, personalized search engine can be regarded as a query-based RS. Personalized search systems aim at providing a better search experience by returning the most relevant results, typically web pages (or resources), in response to a user’s request. A clicktrough data (i.e. an event log of clicks on the search results after submitting a search query) can be used for this purpose as it contains an information about users’ actions and may provide valuable insights 15 into search patterns. The essential part of this data is not just a web page that a user clicks on, but also a context, a query associated with every search request that carries a justification for the user’s choice. The utility function in that case can be formulated as: fR : User × Resource × Query Relevance Score, where Resource denotes a set of web pages and Query is a set of keywords that can be specified by users in order to emphasize their current interests or information needs. In the simplest case a single query can consist of one or a few words (e.g. “jaguar” or “big cat”). More elaborate models could employ additional natural language processing tools in order to breakdown queries into a set of single keywords, e.g. a simple phrase “what are the colors of the rainbow” could be transformed into a set {“rainbow”, “color”} and further split into 2 separate queries, associated with the same (user, resource) pair.

CubeSVD

One of the earliest and at the same time very illustrative works where this formulation was explored with help of tensor factorization is CubeSVD [86]. The authors build a 3-rd order tensor Y = [yijk] M,N,K i,j,k=1 . Values of the tensor represent the level of association (the relevance score) between the user i and the web-page j in presence of the query k: ( yijk > 0, if (i, j, k) ∈ S, yijk = 0, otherwise, where S is an observation history, e.g. a sequence of events described by the triplets (user, resource, query). Note that authors in their work use simple queries without processing, e.g. “big cat” is a single query term. The association level can be expressed in various ways, the simplest one is to measure a co-occurrence frequency f, e.g. how many times a user has clicked on a specific page after submitting a certain query. In order to prevent an unfair bias towards the pages with high click rates, it can be restricted to have only values of 0 (no interactions) or 1 (at least one interaction). Or it can be rescaled with a logarithmic function: f 0 = log2 (1 + f /f0), where f 0 is a new normalized frequency and f0 is, for example, an IDF (Inverse Document Frequency) measure of a web page. Another scaling approach can also be used. The authors proposed to model the data with a third order TD (13) and in order to find it they applied the HOSVD. Similarly to SVD (3), factors U ∈ RM×r1 , V ∈ R N×r2 and W ∈ R K×r3 represent embedding of users, web pages and queries vectors into a lower-dimensional latent factors space with dimensionalities r1, r2 and r3 correspondingly. The core tensor G ∈ R r1×r2×r3 16 defines the form and the strength of multilinear relations between all three entities in the latent feature space. Once the decomposition is found, the relevance score for any (user, resource, query) triplet can be recovered with (14). With the introduction of new dimensions the data sparsity becomes even higher, which may lead to a numerical instabilities and general failure of the learning algorithm. In order to mitigate that problem, the authors propose several smoothing techniques: based on value imputation with small constant and based on the content similarity of web pages. They reported an improvement in the overall quality of the model after these modifications. After applying the decomposition technique the reconstructed tensor T will contain new non-zero values denoting potential associations between users and web resources influenced by certain queries. The tensor values can be directly used to rank a list of the most relevant resources: the higher the value tijk is the higher the relevance of the page j to the user i within the query k. This simple TF model does not contain a remedy for some of the typical RS problems such as cold start or real-time recommendations and is most likely to have issues with scalability. Nevertheless, this work is very illustrative and demonstrates the general concepts for building a tensor-based RS.

Build our recommendation System.

Lets try build our recommendation model for specific case in particular for retail based on the sale transaction data.

Features can be either explicit or implicit, visible or hidden, either at face value or derived through some feature engineering. As a data scientist, feature engineering is an essential part of data wrangling, almost necessary in every ML algorithm, this recommendation system nonetheless. This sample data does not always tell us everything we require, but as a skillful data scientist (I like to call ourselves ninja), we should know how to unlock hidden value. To achieve this, we will perform some unsupervised ML and some data wrangling on the transaction data to derive new features.

Diagram

Description automatically generated

The key to Tensorrec is to realize that there are three important components — User Features, Item Features, and Interactions. As shown in the diagram above, the input data consists of user and item features, and interaction. The engine takes these three data to build a model that ranks the relevant interaction for each user. The example data that we obtain from Kaggle contains roughly adequate content befitting these requirements. What we need to do now is transform raw retail data into these three data requirements.

Diagram

Description automatically generated

# **Chapter 3: Product recommendation system and its implementation for retail sector**

Our data is about one of the supermarket data where we have online purchase information for customers. Like id card id, item ids their prices, the documents from customers so we can see their purchased items and the whole sum for each document.

Our data is for three months September, October and November. Before we our modeling of the data , it would be helpful to do some manipulations on data. Since the datasets are too big and we will do our analysis only for customers that are frequently visited the store and for the items that are purchased the most, we load only datasets for top 1000 customers for each month and top 400 items for each month.

After filtering for each month top 1000 customers and top 400 items we combine all 3 datasets from each month into one total dataset which involve only target users and target items.

**EDA(Exploratory Data Analysis)**

Before constructing the model we need to clean the data , the first step of doing that is understanding the outliers. Since we have already filtered out frequently visited customers we double check if there is no outliers there , to do that it is useful to use the boxplot.

Table

Description automatically generated with medium confidence

Based on this graph it is obvious that we have one customer which has enormous number of visits. After calling this customer we see that the purchases have been made almost every minute.

Dropping out the customer and calling the box plot to see if we have such outliers or not.

Chart, box and whisker chart

Description automatically generated

As we can see we still have the outliers however we need those outliers in our analysis since the average customer with visits around 200 times aren’t our most important target. We focus on top customers that are generating our revenue, so in case we will keep these outliers in the data.

We also interested in items and it is meaningful to do the same steps for items.

Chart, box and whisker chart

Description automatically generated

Based on this graph it is concluded that we have no outliers in our items.

We are construct our model based on K nearest neighbor models and as the distance it was used cosine similarity function to do our estimations. Before applying this method to the dataset we will divide the data into two parts training and testing datasets. As we have time series data it is reasonable to pick as a test data the second half of November.

Our matrix has the users as rows and items as columns. We will be doing our item based recommendations. We will input the item with id number 52 and will see the top 30 recommended items for that particular item.

Text

Description automatically generated with medium confidence

After this we will input the 52nd item to test dataset to see the recommended items for test dataset.

A screenshot of a computer

Description automatically generated with medium confidence

Looking on our results we see that for 52nd item we have 25 right predicted items and we can say that we can get 83% right prediction for each item.

# **Conclusion**

As we analyze most of existing models that are used in consumer modeling, based on our data specification recommendation systems are chose as a final algorithm to model our data. In this work has been used item based recommendation system which in 83% cases is doing right recommendation to the user based on the item that is already in the bucket. The analysis in this work is very important and useful, since it is common problem for businesses to analyze and model their data. Above, we have also discussed the problems regarding the recommendation systems that are still exist and solve one of them for our case.

# **References**

[1] Aciar, S., Zhang, D., Simoff, S. & Debenham, J. 2007. Informed Recommender: Basing Recommendations on Consumer Product Reviews. IEEE Intelligent Systems 22(3):

[2] Adamopoulos, P. & Tuzhilin, A. 2014. On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected. ACM Transactions on Intelligent Systems and Technology (TIST) 5(4): 1-32.

[3] Adomavicius, G., Manouselis, N. & Kwon, Y. 2011. Multi-Criteria Recommender Systems. Dlm. Ricci, F., Rokach, L., Shapira, B. & Kantor, P. B. (pnyt.). Recommender Systems Handbook, hlm. 769-803. Boston, MA: Springer US.

[4] Adomavicius, G., Sankaranarayanan, R., Sen, S. & Tuzhilin, A. 2005. Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach. ACM Transactions on Information Systems (TOIS) 23(1): 103-145.

[5] Adomavicius, G. & Tuzhilin, A. 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.

[6] Vall A, Eghbal-zadeh H, Dorfer M, Schedl M, Widmer G (2017) Music playlist continuation by learning from hand-curated examples and song features: alleviating the cold-start problem for rare and out-of-set songs.

[7] Van Meteren R, Van Someren M (2000) Using content-based filtering for recommendation. In: Proceedings of the workshop on machine learning in the new information age, Barcelona, Spain, pp 47–56

[8] Verbert K, Duval E, Lindstaedt S, Gillet D (2010) Context-aware recommender systems. J Univ Comput Sci 16(16):2175–2178

[9] Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol PA (2010) Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J Mach LearnRes11 (Dec):3371–3408

[10] Volkovs M, Yu G, Poutanen T (2017) Dropoutnet: addressing cold start in recommender systems. In: Proceed-ings of the 31st annual conference on neural information processing systems, Long Beach, CA, USA, pp 4964–4973

[11] Vuurens JBP, Larson M, de Vries AP (2016) Exploring deep space: learning personalized ranking in a semantic space. In: Proceedings of the 1st workshop on deep learning for recommender systems, Boston, MA, USA, pp 23–28

[12] Zheng Y, Liu C, Tang B, Zhou H (2016a) Neural autoregressive collaborative filtering for implicit feedback. In: Proceedings of the 1st workshop on deep learning for recommender systems, Boston, MA, USA, pp 2–6

[13] Zheng Y, Tang B, Ding W, Zhou H (2016b) A neural autoregressive approach to collaborative filtering. In: Proceedings of the 33rd international conference on international conference on machine learning, New York, NY, USA, vol 48, pp 764–773

[14]Zhou J, Albatal R, Gurrin C (2016) Applying visual user interest profiles for recommendation and personali-sation. In: Proceedings of the 22nd international conference on multimedia modeling, Miami, FL, USA,pp 361–366

[15] X. Ning and G. Karypis: SLIM: Sparse linear methods for top-n recommender systems. In Data Mining (ICDM), 2011 IEEE 11th International Conference on, pp. 497-506. IEEE 2011.

[16] M. Deshpande and G. Karypis: Item-based top-n recommendation algorithm. ACM Transaction on Information Systems (TOIS), 22(1), pp. 143-177. ACM 2004.

[17] CC. Aggarwal: Knowledge-based recommender systems. In Recommender Systems pp. 167-197. Springer 2016.

[18] J S. Breese, D. Heckerman, and C. Kadie, “Empirical Analysis of Predictive Algorithms for Collaborative Filtering,” Proc. 14th Conf, Uncertainty in Artificial Intelligence, pp.43-52, 1998

[19] JB. Schafer, J. Konstan and J. Reidel, Recommender Systems in E-Commerce, In Proc. ACM Conf. E-commerce, pp.158-166. 1999.

[20] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa, Effective personalization based on association rule discovery from Web usage data, ACM Workshop on Web information and data management. 2001.

[21] F. Ricci, L. Rokach, and B. Shapira, Introduction to recommender systems handbook. In Recommender systems handbook, pp.1-35. Springer US. 2011.

[22] U. Panniello, A. Tuzhilin, M. Gorgoglione, C. Palmisano, A. Pedone, Experimental comparison of pre vs post-filtering approaches in context-aware recommender systems. In Proceedings of the third ACM conference on Recommender systems, pp.265-268. ACM. 2009

[23] Z. Hyung and K. Lee, Music recommendation using text analysis on song requests to radio stations. Expert Systems with Applications, 41(5), pp.2608-2618. 2016

[24] AA. Kothari, and WD. Patel, A Novel Approach Towards Context Based Recommendations Using Support Vector Machine Methodology. Procedia Computer Science, 57, pp.1171-1178. 2015

[25] R. Burke, “Hybrid Recommender System: Survey and Experiments,” User Modeling and User-Adapted Interaction, vol. 12, 4, 2002, pp.331–370

[26] S. l. Huang, “Designing Utility-based Recommender Systems for E-commerce: Evaluation of Preference-elicitation Methods,” Electronic Commerce Research and Applications, vol.10, 2011, pp.398–407.

[27] D Feng, Utility-based Recommender Systems Using Implicit Utility and Genetic Algorithm. 2015

[28] Y. YANG, Using radial basis function networks to model multi-attribute utility functions. Master's Thesis (Information Management)–National Sun Yat-sen University. 2004

[29] D. Jannach, Z. Karakaya, F. Gedikli, “Accuracy Improvements for Multi-criteria Recommender Systems,” in: Proceedings of the 13th ACM Conference on Electronic Commerce pp.674–689. 2012

[30] M. Nilashi, D. Jannach, O. Ibrahim, et al, “Clustering- and regression-based multi-criteria collaborative filtering with incremental updates,” Information Sciences, vol. 293, pp.235–250. 2015

[31] K. Christakopoulou F. Radlinski and K. Hofmann. Towards Conversational Recommender System, In KDD, 2016, 815-824. ACM 2016

[32] X. Zhao W. Zhang and J. Wang: Interactive collaborative filtering. In Proceedings of the 22nd ACM International Conference on Information & Knowledge management, pp. 1411-1420. ACM

[33] X. Zhao, W. Zhang, and J. Wang, J., Interactive collaborative filtering. In Proceedings of the 22nd ACM international conference on Conference on information & knowledge management pp.1411-1420. ACM. 2013