MILLIMAN REPORT

Life Insurance Sales Recommender System

December 2020

Wing Wong, FSA , Principal & Consulting Actuary
Yun-Tien Lee, FSA, Data Scientist and Actuary
Shenglan Ko, PhD , Data Scientist
Kwan Yau, Data Scientist

Peer reviewed by Chihong An, Kshitij Srivastava and Fanny Pouget

Table of Contents

1. Executive Summary

In the past two decades we have seen every aspect of our lives transformed by the internet, exemplified by the meteoric rise of digital giants like Google, Facebook, Amazon, Airbnb, and Netflix. At the heart of these operations we often find some kind of personalized ranking or filtering algorithms that make use of customer attributes to provide more accurate predictions.

The insurance business, data-intensive as it is, traditionally relied on human agents to analyze customer data, generate sales leads, and recommend suitable products to their clients. The purchase of even a simple basic protection would often require a client to undergo the daunting task of studying and comparing the terms and prices of products provided by various brokers or providers, filling in forms and waiting days if not weeks for approval, sometimes only to end up finding that their specific needs were not covered.

Innovative pioneers, sometimes starting with humble price comparison websites, have since developed sophisticated *decision-support tools* which drastically simplified the process of insurance purchases, generating new business, cutting costs, and increasing customer satisfaction at the same time. In some occasions, underwriting or claim applications were processed within minutes or even seconds after submission. Some of these companies even went on to acquire government licenses to sell their own insurance products. This trend of technological innovation, often called *Insurtech,* has picked up its pace in the past few years, with billions of dollars of investments hitting newspaper headlines.

Traditional insurers often lack the sort of customer behavior data digital retailers have. It is not to say that the traditional insurers' data are worthless. In an effort to enhance insurers' fortunes in using their own data, we conducted this research into product recommendation systems to demonstrate how recommendation techniques commonly used by digital retailers can also assist insurers.

To test our recommendation systems, we partnered with an insurance company which provided the data sample taken from its own experience. We surveyed existing paradigms and algorithms of active recommender services from within the insurance industry and also from the outside. We looked at the various problems the companies face, how the companies are adapting their algorithms to solve them, and tried to draw some common themes and new lessons that are specifically useful for the insurance business. We chose a few classical algorithms, twisted with our own adjustments, and tested them with data provided by our partnering insurance company. We estimated the customer value to the insurer of the cross-selling opportunities generated by the recommendations.

In the process, we also discovered a natural way to segment or classify customers based on their behaviors, which is distinct from naively classifying the customers based on their raw attributes like age, gender, or occupation, which is what common customer clustering tools do. The segmentations made by the model have subtle differences in their education levels and occupation classes but very distinct buying patterns. Such customer segmentations would not be found if focusing on the demographics data only. In view of this, the recommender system opens a window to developing a strategic marketing plan which generates the most profit. We demonstrated that the recommendation system clearly generated value compared to a strategy of simply recommending the most popular product.

In addition to making product recommendations, we were also interested in knowing what products were commonly bought together, what was commonly labelled as “bundle sales” in marketing. Viewing the insurance purchases as market transactions, we utilized *association rule* analysis for insights. An association rule is an implication expression describing the common purchase behaviors with some specified statistics. Knowledge of the rules is crucial for understanding the structure of products of the company. Agents with the relevant association rules in mind in customer visits enhance the chance of bundle sales.

2. Introduction

2.1 BACKGROUND

E-commerce giants such as YouTube, Netflix, Amazon, and Spotify use recommender systems as a key to visitor satisfaction and revenue growth. Recommender systems use a customer's expressed interests, purchase records, ratings, and other attributes to generate a list of personalized recommended products for each customer. Commentators have quoted that 80% of Netflix streaming activities are influenced by the recommender system. The equivalent figures for Amazon and YouTube are reported to be 30% to 60%.[25]

A recommender system is an information filtering algorithm that aims to predict the 'rating' or 'preference' that customers would give to a product they had not yet considered. Recommender-like systems are not new. With the rapid development of new technologies and widely used machine learning techniques, the recommender system can automatically provide customized recommendations by mining through large amounts of customer and product data. It can also help target potential customers by clustering customers in groups that share similar interests. A responsive recommender system that generates recommendations in real time can help provide leads to the salesforce. As the feedbacks are gathered, more insights on the customers can be gained.

In insurance sales, the know-how of products, customer needs, and behaviors vary from a salesperson to another, and the subjective personal experience is hard to pass on. Selling insurance products is unlike selling books, movies, or music. An insurance product helps mitigate uncertainty and maintain economic stability. Insurance products are long-term investments and most customers would only acquire a few in a lifetime, hence insurers do not have the benefits of a large amount of high-frequency and high-recency transactional and behavioral data as online retailers like YouTube or Amazon, and the problem of predicting purchases has a different nature. Moreover, it is worth noting that a fundamental difference between these e-commerce giants and insurance lies in their business models. The value of e-commerce comes from usage and adoption of customers while the value of the insurance industry comes from the ability to price the risk appropriately.

However, insurers do have a large amount of policyholder data and agent activities data that insurers have not typically tapped into. There is potential value to be unlocked. Our research aimed at shedding some light on the usefulness of the internal data that insurers already have, as we believe insurers underappreciate the value of the internal data in general. The fact that insurers have been selling, pricing, and reserving products with those data historically speaks to the usefulness of the internal data, despite its limitations.

2.2 RESEARCH OBJECTIVES

- Survey relevant customer-segmentation techniques and other contemporary analytic technologies people used in recommender systems, especially in the insurance market.
- Design algorithms that can provide real-time personalized recommendations using purchase history and apply them to de-identified insurance customer data.
- Evaluate the potential value of a recommender system to the insurer.

3. Literature Review

3.1 THEORIES AND MODELS

A recommender system (or service) is an information filtering system that provides personalized recommendations of products to customers. It usually involves several components such as data acquisition, core recommender algorithms, output units such as visualization or application-programming interfaces (API), means of feedback, and evaluations. Common approaches to automatic recommendations include knowledge-based algorithms, content-based algorithms, and collaborative filtering (CF).

3.1.1 Knowledge-based algorithms

Knowledge-based algorithms make recommendations to customers based on experts' knowledge.[1] This is essentially a rule-based system where customers specify their needs and the system matches the needs with a subset of products according to rules set by domain experts. This approach dominates in the current insurance market. Customers receive recommendations after their profiles have been built by interactive questions sessions either with a chatting bot or an insurance sales agent. This approach requires highly skilled insurance agents or well-established rules to be effective. That is the reason insurance companies invest heavily in agent training.

3.1.2 Content-based algorithms

Content-based algorithms use the pre-defined attributes of the products or the customers to make a recommendation.[2] For example, in a movie recommender system, product attributes could be genre, movie length, director, etc. In the case of insurance product recommendations, the customer attributes are characterized by their basic demographics and so on.

Product attributes or customer attributes are needed as similar products or customers can be defined and recommendations made. For example, the system will recommend dark comedy movies to a customer who likes dark comedy. However, content-based approaches are known to be less diversified as the model will tend to stick to predefined classifications and keep recommending only dark comedies to the customer in question in the example of movies.

3.1.3 Behavior-based algorithms

Collaborative filtering (CF) algorithms are a class of behavior-based approaches that exploit the past purchase behavior of customers to make recommendations.

In a CF model, customer attributes are not mandatory. The algorithms do not need to know who the customers are, but they need to know what the customers have done as far as purchase history is concerned.

CF models are developed to introduce diversity and serendipity into recommender systems.[4] Serendipity is achieved by the wisdom of crowds. A CF model makes recommendations by utilizing the opinions of other people having similar tastes.

To define people with similar tastes, historical purchases or rating behavior are used. This kind of data represents the explicit interactions between customers and products. There are two types of approaches to describe “similar tastes”: *neighborhood-based* and *latent-space-based*.

In a neighborhood-based approach, similarity between customers is defined using their behaviors instead of their backgrounds. Likewise, similarity between products can be defined with the buying records of products. Popular products bought by the customer's cohort are then recommended to the customer in question.

In latent space-based approaches, it is assumed that customers and/or products are characterized by a few latent features, i.e., *embeddings* or *mappings* of the raw meaningful features into abstract low-dimension numerical vectors, which would ideally remove redundancies in the feature values in the raw data and capture the essential characteristics of the customers. These latent features span the latent space in which each customer and/or product is profiled as a dot. These latent features are not predefined but brewed by the model as a result of past behavior. For example, in a movie recommender system, movie features like genre are not an input of the model but appear as a latent feature which is obtained after parsing through the rating data. After all, the tastes of customers and latent features of products are encoded in behavioral data such as historical purchasing data. The behavioral data is typically sparse as it is rare that customers could experience many of the products. By sparse data we mean that interactions between customers and products are quite limited. However, customers and/or products in latent space are densely represented as they share the same latent features.

Hence, the models can technically provide better recommendations by embedding customers/products in a latent space even if the raw purchase data is sparse.

3.2 COLLABORATIVE FILTERING MODELS

In this report, we apply several state-of-the-art CF models to the recommendation of insurance products. A CF approach particularly fits the insurance setting as many insurers do not have rich or up-to-date customer background information such as life experiences, networks of friends, or financial status. Afterall, one easily obtains the purchase history of customers as long as one sells something. The data collection burden is therefore less severe.

The desired properties of a good recommender system include the accuracy of recommendations, the scalability of the system, and the diversity of recommended products. Most importantly, the customer experience is the key.

The *accuracy* or *success rate* is the ratio of the number of relevant recommendations to the number of total recommendations. The success rate not only evaluates the effectiveness and usefulness of the system, it is also important for customer satisfaction as nothing sours the purchase experience more than repeated failed recommendations. Moreover, it is important to present the recommendations at the right time. Therefore, the model must be scalable so that the service is in real time. Finally, the system must recommend products across all domains of customer interest, so that the system is healthy and in a positive feedback loop.

Taking these into account, three CF models are selected for study in this report.

Before introducing the models, it is useful to have a detailed understanding of the format of the input data. The purchase history can be cast into an interaction matrix, where each row represents a customer and each column denotes a product. If the customer C bought product P, the cell at row C and column P is labelled as 1. In other words, each row records the products a customer bought and each column of the matrix records the customers who bought that product. Such a binary customer-product matrix is used in this report. This is depicted in Figure 1. In the literature, instead of the binary cell, one or nought, cells can have values indicating the interaction strength between customers and products. We will refer to such strength as *ratings* hereafter.

FIGURE 1: DATA LAYOUT IN CF MODELS

	A&H_others	Saving_curr1_SP	A&H_Cancer	Retire_curr1_annuity	…	A&H_surgery
policyholder 1		1		1		
policyholder 2		1				
policyholder 3			1			
policyholder 4			1			
policyholder 5						
policyholder 6	1	1	1			
policyholder 7				1		1
policyholder 8						
policyholder 9	1		1			
policyholder 10				1		

Figure 1 demonstrates purchase history data fed into CF models. In the figure, policyholder 1 has purchased Saving_curr1_SP and Retire_curr1_annuity and nothing else. Those cells representing products which have not been purchased by a customer are then left as blanks. Those blanks are of particular interest as usually customers would gradually fill in those blanks as time goes by when they make more purchases. A good recommender system should lead customers to fill in those blanks. When there are only a few 1s existing in the matrix, the interaction matrix is said to be sparse. The more sparse the matrix is, the fewer patterns the model has to learn and hence the model performance is considered poorer. For a selling system with low frequency transaction records, one needs to be careful about the sparsity issue of the input data.

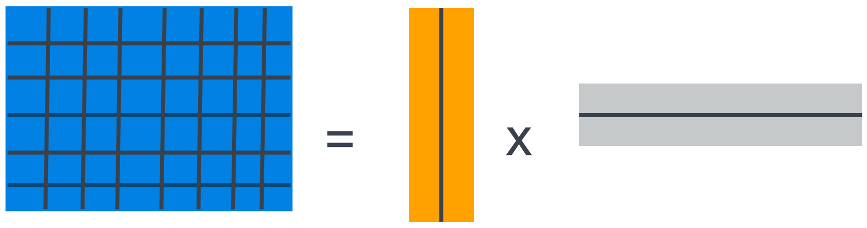
3.2.1 Matrix Factorization model

The Matrix Factorization (MF) model[5] became well known during the Netflix prize competition after Simon Funk's 2006 blog post appeared.[6] The idea of MF is to map each customer and each product to a latent space. Each latent dimension represents a feature characterizing the taste of the customers and the intrinsic features of the products. For example, for an interaction matrix of the movie ratings, the latent features could be comedy versus drama, or degree of children orientation, from the product point of view. However, it is possible that some of the latent features are uninterpretable.

After mapping customers and products to a latent space, each customer and product is represented by vectors of the same rank. The predicted rating or the strength of the interaction is then obtained via the inner product of the two vectors. If the customer prefers products of a certain latent feature and the products happen to possess a high degree of that latent feature, the rating will be predicted high. In other words, the rating is predicted high if the customer and product have a high correspondence in the latent space. The number of latent features each customer and product has is called a *hyperparameter*. That is, we have to make an assumption in a priori how many latent features characterize each customer and product and this number is usually just a few.

Mathematically, this amounts to a *factorization* of the original interaction matrix into a product of two lower-rank matrices. Such decomposition of a matrix is not particularly fancy as the procedure is standard in the literature. The novelty is that the matrix is split based on only known (non-missing) values in the cells. In Figure 2, the empty cells denote the interactions which have not happened between customers and products. Via the matching of the customer and product vectors in the latent space, the empty cells are filled-in numbers which imply the possibility that the associated customer will buy the corresponding product. These are the predictions we want to make. If there are fewer empty cells to fill in, we say that the data is denser. In this case, making recommendations is easier as we have more information. Reversely, if there are many empty cells initially, the data is sparse and making recommendations is then challenging.

FIGURE 2: ILLUSTRATION OF MATRIX FACTORIZATION PROCESS



The above picture shows a factorization of the original matrix (on the left-hand side) into a product of two lower submatrices with the choice of two latent features. On the left-hand side, the customer in the input dataset is characterized by a row which describes the corresponding past purchasing records. On the right-hand side, each policyholder is simply represented by two latent features. Similarly, products on the right-hand side are also represented by the same two latent features. The inner product of the two-dimensional vectors of customer and product is the model prediction of the cell value on the left-hand side.

3.2.2 Restricted Boltzmann Machine model

A Restricted Boltzmann Machine (RBM)[7] is a two-layered neural network. One layer is called a *visible layer* whose nodes correspond to the observations (or ratings) made, and the other one is a *hidden layer* which encodes the interdependencies between the visible nodes. The visible and hidden layers are inter-connected symmetrically, while intra-links among visible nodes or hidden nodes are forbidden. This restriction of links enables efficient learning algorithms and fits the purpose of recommender systems. The values of the hidden nodes are restricted to binary units.

During the training process, each row of the interaction matrix is fed into RBM in turn. The number of visible nodes equals the number of total products. On the other hand, the number of hidden nodes is a hyperparameter. RBM adjusts its parameters to allow the maximum likelihood of observing the known ratings. The customer row vector first propagates forward and gets encoded in the hidden layers, then a back propagation is carried out trying to maximize the likelihood of observing the input data. RBM is intrinsically stochastic as nodes that are lighted or not are determined stochastically. If a hidden node is lit, its value is set to 1. Just like in MF, latent features may represent interesting attributes of the products, with hidden states in RBM corresponding to certain buying patterns of a cohort.

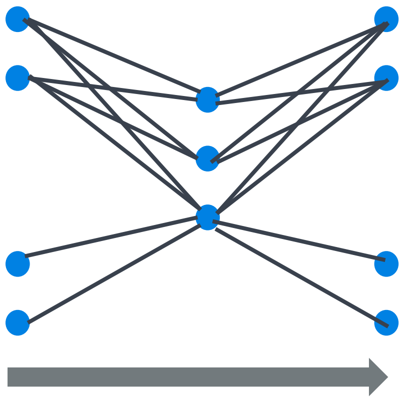
Once trained, an RBM can provide recommendations as follows. Past purchases of a customer, encoded as a Boolean 0/1 vector, are fed into the visible layer of the model, and forward propagated to the hidden layer with the weights (and appropriate biases) to create a *representation* of the customer's behavior. Then a backward propagation is performed with the same weight matrix and appropriate biases to reconstruct a purchase vector in the visible layer. This reconstruction is then interpreted as recommendations for the customer.

3.2.3 Autoencoder model

Another unsupervised simple neural network model serving CF purpose is autoencoder[8] The algorithm tries to find a pair of functions, an *encoder* that maps the inputs to their essential patterns, and a *decoder* that best recovers the input values from these patterns. The plain *vanilla autoencoder* consists of three layers, an input layer, an encoding hidden layer, and a decoding output layer. The data just propagates forward and gets encoded in the hidden layer and then reconstructed at the output layer to be as close as possible to the input, measured by metrics such as the *Mean-Square Error* or *Binary-Crossentropy*. In the application to CF, the number of nodes in the input layer is the number of total products. The number of hidden nodes is a matter of user choice. One can also insert more hidden layers to refine the encoding and/or reconstruction mechanism. These assumptions together with the nonlinear activation functions associated with nodes are all hyperparameters.

The autoencoder model is an example of the latent-space approach in the behavioral-based CF models. In the autoencoder model, rows of the interaction matrix in Figure 3 are fed into the network one by one. Purchasing records of each customer are recorded in the input layer and then such information is propagated forward to reach the hidden layer. In this way, the purchasing records become the latent features of the customer and get encoded in the hidden layer. In the second stage of the autoencoder model, the latent features of a customer are transformed to the predicted interests of the customer in all products available. Customized recommendations can then be made to each customer accordingly. Just like in the case of MF, it is interesting to explore whether these latent features mean anything and whether the customers are segmented in the latent space.

FIGURE 3: THE NETWORK ARCHITECTURE OF THE VANILLA AUTOENCODER



The above plot depicts the vanilla autoencoder which is the simplest type of autoencoder. It consists of three layers. The leftmost and rightmost layers are visible and serve as input and output. The middle layer is a hidden layer which encodes and stores the input information. In this picture, the number of latent features is chosen to be three.

3.3 USER CASES

Recommender services differ in their methods for data acquisition, recommendation generation, output or visualization, obtaining feedback, and overall evaluations. In this section we survey existing recommender services in insurance. As technology changes every day, this survey is far from complete, but shows the diversity of the systems, which also reflects features inherent in the respective insurance markets.

3.3.1 The insurance industry

Insurance technology, or *Insurtech*, has seen massive investments in recent years. In the competitive US employee benefits market, we have seen a proliferation of *decision support tools*, or recommender-system-like algorithms. Armed with vast databases of domain knowledge and natural language processing and other advanced analytic techniques, these tools help customers narrow down a vast field of insurance products to a handful of top choices or even a specific profile recommendation, based on how close the products combined match each individual customer's needs and preferences.

Prudential Financial, in their quest for improving sales to the middle market, acquired AssuranceIQ*,* a direct-to-customer online platform which combines data science and human domain knowledge to sell insurance policies from a multitude of providers.[13] The company also sponsored a *Kaggle* contest in 2015 for automated solutions to ease the insurance application processes.[26]

Massachusetts Mutual Life Insurance Company (MassMutual) acquired Picwell*,* a decision-support platform focused on employee benefits and health insurance. Founded by a few university professors, the platform predicts future care expenses for individual customers and provides product recommendations that match their risk preferences, based on short questionnaires which go as far as what medications the clients are currently taking.[14,15] Claims and demographic data, involving both public and private information, were used to train the models.

The private health care benefits exchange Liazon (founded in 2007 and purchased by Towers Watson & Co. in 2013) is another well-known provider of online decision support tools.[15] Based on a short questionnaire completed by the client, their engine would recommend a number of top product choices matching the client’s expected expenses and preferences in premium and physician networks. It is reported that a majority of the clients would select from the top recommendations.[16]

In 2019, Willis Towers Watson Public Limited Company also acquired Tranzact, a marketing solutions provider that uses multiple data science tools such as social media, email campaigns, search engine optimization techniques, and website traffic analyses to generate sales leads.[17]

The private health care exchange, GoHealth, became publicly traded in July 2020, after receiving US$1.5 Billion from the investment firm Centerbridge Partners in September 2019, in exchange for a majority stake. GoHealth started originally as a website for comparing health insurance products for agents and then became an approved exchange in 2012. Sales leads are generated from multiple sources such as advertising on social media, search engines, and television.[18]

Chaucer Holdings, the specialty reinsurer, has formed a partnership with Artificial Labs which specializes in developing more responsive and efficient underwriting.[19]

The online Ladder Life Insurance (founded in 2015) partners with A-rated insurers to offer straightforward adjustable term-life contracts to clients between 20 and 60 years of age. The company use crowd-sourcing to grow their client network and pay their registered members for referrals*.* Application is done purely online and a simple calculator is provided for estimating the applicants’ future needs for coverages. As is typical of online tech companies, APIs are provided in addition to their web-interface.[23]

Sproutt (started in 2019), is another online marketplace for life insurance. Their proprietary *Quality of Life index* is a 15-minute interactive Q&A session based on medical research; *cognitive computing* (computer models to simulate human thought processes) aims not only to assess the risk factors of the applicant but also provide suggestions for promoting healthy lifestyles.[24]

In 2016, a team of actuaries and data scientists in Hong Kong founded Seasonalife Ltd., an online insurance sales support platform for registered insurance agents and brokers.[11] Seasonalife provides insurance product comparison and recommenation services based on simple facts that the clients provide about themselves and the product information that agents upload to the system. Members of the team partnered with Sun Life Hong Kong (which provided the funding of US$30 Million), to found Bowtie Life Insurance Company, which obtained the first virtual insurance license in 2018 from the Hong Kong Insurance Authority. The focus of Bowtie is to provide end-to-end online self-served services for low-cost, simple protection products like the standardized *Voluntary Health Insurance Scheme* sponsored by the government.[12]

Some recommender engines are bundled with another emerging technology—the *chatbot.* Lemonade Insurance Company, which sells home and pet insurances, design and build chatbots to support the insurance cycle from purchase to customer services and claims.

The London-based automated insurance agent, SPIXII, has at its core a chatbot that uses *natural language processing* as well as multiple sources of user data and behavioral economics to interact with clients and predict the clients' needs. It can be integrated into common messenger apps such as Telegram, Line, or Facebook.[20]

In Taiwan, the private insurance market is dominated in dollar terms by life insurance, as the country provides national health insurance in which every resident in Taiwan must enroll. Researches on insurance purchase behaviors mostly focus on specific sectors of the market, such as teachers, senior citizens, or effects of personal habits. Recommendations of sales are mostly done in the traditional way—through agents they know in person. A few limited online recommender-like services are provided by insurance companies or private third-party agents. Several insurance companies, such as Taiwan Life Insurance Company, Fubon Life Insurance, and TransGlobe Life Insurance, provide on their websites case studies or typical scenarios for the potential customers, complete with simple rule-based online-calculators for premiums. Some provide simple filtered searches for their products.

The Taiwan Life Insurance Company developed *iKash,* a digital platform for supporting sales, client recommendations, and management for agents in the fields.[21]

For the past few years, Cathay Life Insurance Company and its parent Cathay Financial Holding have invested heavily in their AI chatbot *Alpha*. Its current focus is on the customer service side, and only provides limited keyword-based website referrals to requests for recommendations of insurance agents.[22]

The third-party website, Finfo.tw, compares and ranks insurance products of various providers. It provides personalized services for their members, registered through their Facebook accounts, through a chatbot. The company provides recommendations by popularity, occupation, or personalized needs analysis.

Despite the on-going digitalization, we expect human interactions offered by the insurance companies to remain focused on discovering customer needs and delivering the best customer experience. That is the reason why traditional insurers continue to invest in direct distribution and solutions to augment their direct channel efficiency. We could expect agents to continue to play a vital role in the insurance distribution process in the near to mid-term with the help from Insurtech and big data analytics.

3.3.2 Notable cases from other industries

Every designer of automated recommender systems must solve problems specific to their own requirements. Key questions include data acquisition, data representation, cold-start, result interpretation, scalability, and balancing of objectives. Most companies like Facebook, Amazon, Netflix and YouTube*,* start the learning curve by applying simple-and-proven algorithms such as Nearest Neighbors or MF to their in-house data, fine-tuning their algorithms and parameters with offline testing, gaining user feedbacks with online A/B testing, and then iterate. Many have since moved towards Deep Learning, which harnesses big data to train highly nonlinear recommender models with millions of parameters to create closer fits to diverse needs.

A common design of a scalable recommender system is to represent (or embed*)* the customer and product profiles in some meaningful low dimensional vector space and pre-train the model in batch mode in regular time intervals. During production time, simple algorithms such as a distance function or Nearest Neighbors are applied to the low dimensional representations for a low-latency service.

In terms of timescale and significance, signing a new insurance contract is sometimes similar to professional recruitment. The professional networking site LinkedIn developed the *Recruiter* tool to process search queries and internal data for recruiting agents to produce lists of profiles of suitable candidates, optimized jointly by match and response likelihood. It is built on top of the sitewide recommender-platform called the *Browsemap,* which serves tailor-made recommendations to various kinds of requests. It uses a range of algorithms, including MF, Decision Trees, Deep Learning, and Unsupervised Learning. Professional specialties are inferred by a Topic Modelling algorithm called *Latent Dirichlet Allocation*. The algorithms are context-aware, which means the browsing actions of the current session are recorded and used as inputs to the recommender. All the data of the website is organized into the big *LinkedIn Economic Graph.*

The lodging and travelling website, Airbnb, maps the features of every listed property into 32 real numbers, creating a dense and meaningful representation. It then uses simple k-Means Clustering and Cosine Similarity algorithms to find similar properties. The cold-start problem for a property is solved by data imputation from Nearest Neighbors.

UberEats, the food ordering and delivery service, uses natural Language Processing to build an in-house knowledge graph that connects similar products, and represents customer tastes and restaurant cuisine profiles as vector spaces. Diversity is created by solving the Multi-Armed Bandit Problem via the *Upper Confidence Bound* approach. Feedbacks come explicitly through customer feedbacks and the time between repeated purchases.

Facebook has moved from the original MF approach in 2015 to an in-house Deep Learning Recommender Model approach as published in 2019. The sparse categorical data and the numerical data are both mapped into latent factors, creating a dense vector representation. With data size measured in billions the algorithms are highly parallelized, supported by a variant of the Apache Giraph infrastructure.

For companies that were built upon recommender services, *Amazon* described their product-to-product CF algorithm in an award-winning scientific paper. *Netflix* sponsored a million-dollar competition to improve their recommender algorithm, and at the heart, the winning algorithm is a combination of MF and RBM. Topic Modelling is used to create pages of highly personalized listings to optimize user experience. The *Yahoo News* team in Japan moved from a Bag-of-Words model to a data embedding model, generating representations of articles via a Denoising Autoencoder and representation of users by Recurrent Neural Network in batch mode. Realtime ranking and recommendations are computed by a simple Dot Product algorithm, after some additional domain-specific feature engineering, like freshness and expected pageviews.

4. Research Methodology

4.1 DATA SOURCE

For this study we partnered with a life insurer which provided sample data. The purchase history of the policyholders was used as the implicit feedback data for the recommender system.

The dataset contained policyholder records of the past 19 years. There were roughly 900,000 policyholders in the raw dataset. We also obtained the basic demographics of the policyholders including gender, education levels, occupation classes, and birth dates.

To offer recommendations in an efficient and practical way, we aggregated insurance plans into product types. There were over 5,000 plan codes in total and many of them were actually quite similar as many were just new eras of the same product. To prevent the model from learning these artificial correlations and make the data denser, we grouped them into 31 product types. The recommendations were then based on these 31 product types.

Next, we truncated the dataset by removing inconsistent and unnecessary records. For example, we removed records associated with product types which were no longer offered for sale or sales that occurred during the testing period. The number of policyholders was reduced to 300,000 after data preprocessing.

We recast the purchase records into a form of customer-product matrix where each row represented a policyholder and each column represented a product type. If a policyholder ever bought a product type, it was marked “1” in the corresponding cell. Notice that there may be repurchases or multiple purchases. However, the customer-product matrix just defined was binary in order to avoid the popularity bias.

In the end, we had both training and testing customer-product matrices of shape (300,000, 20). The remaining 20 product types are shown in Figure 4.

FIGURE 4: INSURANCE PRODUCT TYPES AND THE ASSOCIATED DESCRIPTIONS

Product type code	Description
A&H_1yr_Renewable	Health: one year renewable
A&H_Cancer	Health: cancer
A&H_surgery	Health: one year rider with surgery protection
A&H_health	Health: pure health
A&H_others	Health: others including disability, long term care and so on
Invest_annuity_GMXB	Investment with annuity GMXB
Invest_annuity_other	Investment others
Invest_life_BE	Investment life product backend
Invest_life_FE	Investment life product frontend
Prot_curr2_6	Protection in currency 2 and paying period 6 years
Prot_curr2_>6	Protection in currency 2 and paying period more than 6 years
Prot_curr1_other_6+	Protection in currency 1, others, paying period more than 6 years
Prot_curr1 _term_6+	Protection in currency 1, term life, paying period more than 6 years
Retire_ curr1_annuity	Retirement product in currency 1, annuity
Saving_curr2_RP2-5	Saving in currency 2, regular pay, paying period 2-5 years
Saving_curr2_RP6+	Saving in currency 2, regular pay, paying period more than 6 years
Saving_curr2_SP	Saving in currency 2, single pay
Saving_curr1_RP2-5	Saving in currency 1, regular pay, paying period 2-5 years
Saving_curr1_RP6+	Saving in currency 1, regular pay, paying period more than 6 years
Saving_curr1_SP	Saving in currency 1, single pay

4.2 RESEARCH DESIGN

4.2.1 Model building

The data were structured in a format of customer-product interaction matrices, i.e., the purchasing history characterizing the customer behaviors. Such format of data is suitable for CF models. CF models find customers of similar tastes and recommend products shared by the cohort to customers who haven't explored those products. Among the many CF approaches and models, we implemented three classical algorithms—MF model, RBM model, and autoencoder model. All of them are latent space approaches which map customers and/or products to a latent space in which the vectors characterize the profiles of either customers or products.

We divided the purchase data into two periods, one for training purposes and the other for testing. The training period was from 2000-01-01 to 2018-01-01 and the testing period was from 2018-01-02 to 2020-09-30. The longer the training period, the denser the dataset. However, considering both recency and density, we chose the recent 18 years as the training period. The data in the training period was used for model building, while the data in the testing period was used for the evaluation purposes.

4.2.2 Evaluation

The output of each model was a customer-product interaction matrix in which the unknown ratings were filled in. For a system in which the exact number of ratings is important, the absolute values in the output matrix are relevant. For example, in the Netflix movie rating system, there are five possible ratings, 1,2,3,4,5, corresponding to extremely dislike, dislike, neutral, like, and very like, respectively. In such a case, the root mean squared error (RMSE) or sum of squared errors is useful.

For this study, we were interested in personalized recommendations for the appropriate insurance products to customers. Therefore, what mattered were the top-n recommendations made by the models. Once the number of recommended products was determined, we computed the success rate (which is also known as precision) and the coverage ratio (which is also known as recall) of the recommendations. The success rate was defined as the ratio of successful recommendations over the number of total recommendations. The coverage ratio was defined as the ratio of the number of successful recommendations over the number of actual purchases. A high success ratio implied a high customer satisfaction while a high coverage ratio meant the recommender system uncovered all potential revenue sources.

4.3 METHODOLOGY

4.3.1 Matrix Factorization model

The matrix model briefly introduced in Section 3.2.1 was implemented. Both customers and products had a profile in a k-dimensional latent space, where k was a hyperparameter to be chosen. The degree of correspondence between customer and product profiles in the latent space determined the possibility of the purchase. In Section 5, we show the success rates and coverage ratios for each product with the choice of $k = 2$ and $k = 7$.

One can either use stochastic gradient descent or the alternating minimization approach to minimize the sum of squared errors of the known ratings. In this example, an L2 regularization term was inserted to avoid overfitting, as the sample size was not large. The alternating optimization learning approach had the advantage of being parallelizable.

As both customers and products were characterized by the profiles in the latent space, it was interesting to observe the clustering of customers and products in such lower dimensional spaces. For example, internal customer segmentations based on the purchasing behavior utilized by the CF model was explored to gain insights about customer relationship management. By doing clustering of products in the latent space, one is able to better classify the products and elicit the design of the future products. It is shown in Section 5 that policyholders were well separated into four groups in the subspace of the latent space composed of the first and third latent features. We also show that some of the demographic features of policyholders happened to distinguish the groups too.

4.3.2 Restricted Boltzmann Machine model

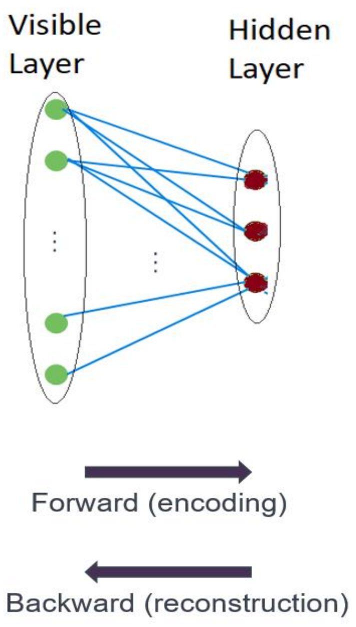
The RBM, an early success for neural networks, formed one of the pillars to the winning ensemble solution to the million-dollar Netflix prize in 2009, along with MF, K-nearest neighbors, and Global effects.

RBMs, once unfolded, were quite similar in structure to the autoencoder model (to be described in the next subsection). The main differences were: 1) The hidden layers of RBMs were Boolean valued and those of an autoencoder were real-valued; 2) RBMs were trained by stochastic *Markov Chain Monte-Carlo (MCMC) sampling,* whereas autoencoders were trained by *backpropagation.*

RBMs, once very popular, are very simple in structure. RBM is stochastic in nature and needs some care to train properly. As computing powers have increased dramatically in recent years, interests in the machine learning community have since shifted to autoencoders which are more robust and fit well in backpropagation software frameworks such as Tensorflow or Pytorch.

An RBM has one visible layer and one hidden layer with *weighted edges* connecting each pair of *nodes* from the two layers, and a *bias vector* for each layer. The number of visible nodes is equal to the number of product features (columns) in the dataset. The number of hidden nodes is a hyperparameter. Each node, visible or hidden, takes a binary value (1 or 0), i.e., it is either light or dark. The weight attached to each edge represents the relationship strength between the corresponding pair of nodes from the two layers. The bias vectors represent over-all biases for individual nodes, in addition to the weighted averages coming from the edges. Given the node values on one layer, the node values on the other layer are stochastically determined with probabilities coming from the weighted sums of nodes connected to it. Figure 5 provides a conceptual diagram of this process.

FIGURE 5: TRAINING PROCESS OF RBM



During training, data is fed into the visible layer whereby the hidden layer will try to learn to represent underlying (latent*)* features and the edge weights will learn to represent the relationship strength between each pair of nodes from the two layers by minimizing a quantity called *energy* which has its origin in statistical physics. The standard algorithm for training the weights is called *contrastive divergence*.[9] It is a kind of statistical MCMC sampling which bypasses computational complexities from the original definition.

4.3.3 Autoencoder

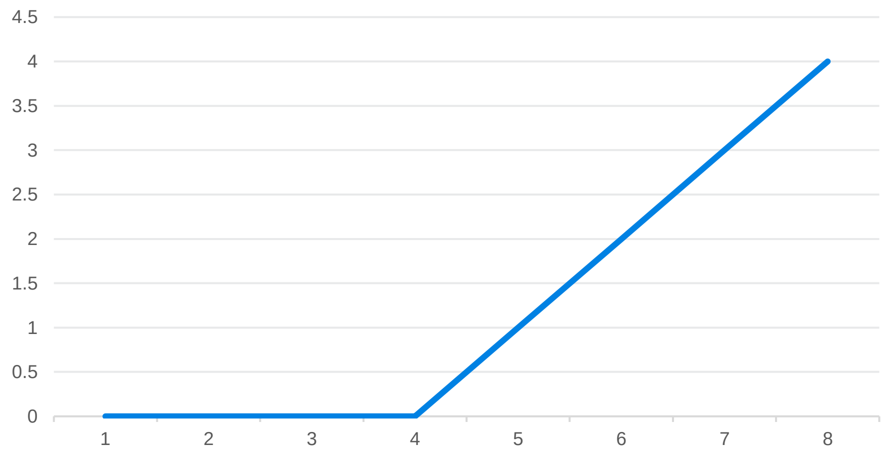
Autoencoder is a simple neural network briefly addressed in Section 3.2.3. It tries to encode the customer behaviors in terms of a few hidden nodes and then decode the compression to regenerate the original input. Usually, an autoencoder comprises several layers of neurons arranged in a symmetric manner in which the middle layer is the bottleneck that contains the least number of neurons. In this report, the autoencoder is implemented by five layers where two of them are input and output and three of them are hidden. The activation functions are simple *nonlinear rectified linear units (*ReLU*)*. Figure 6 summarizes the network architecture of the autoencoder which is used to produce the result in Section 5.

FIGURE 6: NETWORK ARCHITECTURE OF THE AUTOENCODER MODEL USED IN REPORT

Layer	Output Shape	Parameter number
input layer	(none, 20)	0
hidden 1	(none, 8)	168
hidden 2	(none, 4)	36
hidden 3	(none, 8)	40
output layer	(none, 20)	180

A plot of the ReLU function is shown in Figure 7 below. The activation function transforms an input node value to an output node value which is then ready to feed forward to the next layer of nodes. The nonlinear operation of activation function provides nonlinearity of the model.

FIGURE 7: THE SHAPE OF THE NONLINEAR RELU FUNCTION



4.3.4 Feedback loop

The possibility of real-time feedback was implicit in our choice of algorithms in this research. These were parameters tunable at run time. And because our recommendation methodologies were essentially based on grouping customers' prior purchases, if the agent detected interest of a customer in a certain product in real time, a real-valued weight between (0, 1) could also be entered into the recommender for a more personalized recommendation. This is similar to *product-to-product collaborative filtering* in well-known recommender models such as that of Amazon.com. It is worthwhile to test these methods in the field, but it was out of the scope for the current project.

4.3.5 Software tool

The autoencoder can be easily implemented in the Tensorflow 2.0 framework. Both the MF and RBM models can be built using the scikit-learn package of Python. In this study we coded our RBM from scratch as it facilitated fine tuning and optimization of the training process, as suggested in Geoff Hinton's well-known *Practical Guide.*[10]

5. Results & Applications

5.1 COLLABORATIVE FILTERING MODELS

5.1.1 Matrix factorization

Our client, a major local life insurer, provided us with about 20 years of purchase data, about 300,000 customers and 500+ products grouped into 20-30 distinct classes, covering the spectrum from life, health, accident, and annuity. Our initial research focused on MF with a standard tool in linear algebra called *Singular Value Decomposition*, or SVD. For convenience of visualization, we chose two hidden dimensions, and we found that the customers were naturally grouped into clusters in the latent space.

For evaluation we used *Success* and *Coverage* rates (also called *Precision and Recall* in the machine learning literature). To test the model, we performed *back-testing*. We took a convenient cut in time, putting customer behaviors before the cut into “training” data and those that came after the cut into “testing” data. For the famous “Netflix” recommender model, a random masking approach is applied to test the model. In a random masking approach, one takes all the purchases of every customer in all of the data given, masking some, and then evaluating those missing values to predict whether the customer would purchase or not. This approach tests the ability of the model to complete the purchases; however, it is somewhat harder to interpret or apply in practice than the back-testing evaluation method.

We chose to recommend two products to each customer, i.e., top-2 recommendation. The number of products presented to each customer is a matter of choice. It should be a balance between the amount of information presented to the customer and the chance for creating leads. With top-2 recommendations, we obtained the following result.

5.1.1.1 Success rate and coverage ratio

If we assumed each product type and each policyholder were characterized by only two latent features, we obtained the following success rates and coverage ratios of each product type. The overall success rate and coverage ratio in the case of two latent dimensions was 38.43% and 42.03%, respectively. In the case of seven latent dimensions, the overall success rate and coverage ratio was 27.17% and 29.71%, respectively. The breakdown of the rates for each product is listed in Figures 8 and 9.

FIGURE 8: SUCCESS RATES AND COVERAGE RATIOS WITH TWO LATENT DIMENSIONS

2 latent dimensions	popularity	success rate	coverage ratio
A&H_1yr_Renewable	33.7	46.99	94.48
A&H_others	15.26	51.73	63.61
A&H_health	8.33	NA	0
Saving_curr1_RP6+	4.77	NA	0
A&H_Cancer	11.48	NA	0
Prot_curr1_other_6+	4.22	NA	0
Invest_life_FE	11.73	8.78	55.74
Saving_curr2_RP6+	2.76	NA	0
A&H_surgery	0.24	NA	0
Invest_annuity_other	4.26	1.71	15.50
Prot_curr1_term_6+	0.1	NA	0
Prot_cur2_6	0.01	NA	0
Prot_curr2_>6	0.34	NA	0
Saving_curr1_RP2-5	1.54	NA	0
Saving_curr1_SP	0.76	NA	0
Saving_curr2_RP2-5	0.12	NA	0
Saving_curr2_SP	0.08	NA	0
Invest_life_BE	0.24	NA	0
Invest_annuity_GMXB	0	NA	0
Retire_curr1_annuity	0.06	NA	0
Overall rates		**38.43**	**42.03**

FIGURE 9: SUCCESS RATES AND COVERAGE RATIOS WITH SEVEN LATENT DIMENSIONS

7 latent dimensions	popularity	success rate	coverage ratio
A&H_1yr_Renewable	33.7	51.99	58.75
A&H_others	15.26	50.45	27.88
A&H_health	8.33	19.6	23.58
Saving_curr1_RP6+	4.77	14.84	28.49
A&H_Cancer	11.48	13	36.19
Prot_curr1_other_6+	4.22	10.77	17.47
Invest_life_FE	11.73	10.65	27.11
Saving_curr2_RP6+	2.76	9.63	3.02
A&H_surgery	0.24	2.83	13.33
Invest_annuity_other	4.26	1	11.07
Prot_curr1_term_6+	0.1	NA	0
Prot_curr2_6	0.01	NA	0
Prot_curr2_>6	0.34	NA	0
Saving_curr1_RP2-5	1.54	NA	0
Saving_curr1_SP	0.76	NA	0
Saving_curr2_RP2-5	0.12	NA	0
Saving_curr2_SP	0.08	NA	0
Invest_life_BE	0.24	NA	0
Invest_annuity_GMXB	0	NA	0
Retire_curr1_annuity	0.06	NA	0
Overall rate		27.17	29.71

The popularity column, which is defined as the percentage of purchasing counts, was inserted as a baseline reference. In general, we found that the customized recommendations outperformed recommending popular products only.

The NAs in the success rate column denote a situation where the corresponding products were not recommended at all. This was possible because only two product types were recommended to each policyholder.

Both the overall success rate and coverage ratio in the case of seven latent dimensions were lower than the corresponding numbers when using only two latent dimensions. However, the recommendations in the case of seven latent features were more diversified. This allowed us to explore more products and not only the most popular ones. The choices of the number of latent dimensions and the number of recommendations to each customer are the trade-off between *exploitation* and *exploration* as well as customer satisfaction and the amount of potential revenues uncovered.

As defined above, the success rate is defined as the ratio of successful recommendations over the number of total recommendations, and the coverage ratio is defined as the ratio of the number of successful recommendations over the number of actual purchases. In Figure 9, the coverage ratio for A&H_1yr_Renewable was quite high (94.48%) while the success rate was only 46.99%. This means that the MF model with only two latent dimensions recommended A&H_1yr_Renewable many more times than it was purchased. As a result, the chance to recommend other products was missed. This kind of concentration on a few products happens when there are dominant popular products. In this case, it is helpful to increase the number of latent dimensions such that more information of less popular products can be captured, thus more recommendations on these products. Indeed, as shown in Figure 9, we found that the coverage ratio of A&H_1yr_Renewable significantly dropped while the success rate increased slightly. Moreover, more products, which are less popular, were explored in Figure 9 than in Figure 8, though success rates of these less popular products were not as high as the popular ones. Note that the trade-off between success rate, coverage ratio, and recommendation diversity is dependent on context and is a matter of user choice. We chose the MF model with seven latent dimensions for later discussions in this report as it provided more balanced recommendations than the MF model with two latent dimensions.

5.1.1.2 Customer segmentation

The idea of the MF is to group the customers into cohorts of distinct preferences and then make the recommendations accordingly. As a result, a by-product of the MF model is usually a data agnostic clustering of customers.

It is interesting to note that customers are clearly separated into groups in the first and third latent features. Empirically, those latent features often offer insights about customers or products in the hindsight. It is helpful for the development of different marketing strategies aiming at different segments of customers. In Figure 10, we show that these four groups of people have very different buying patterns and demographic profiles.

FIGURE 10: THE FOUR CLUSTERS OF CUSTOMERS IN THE LATENT SPACE

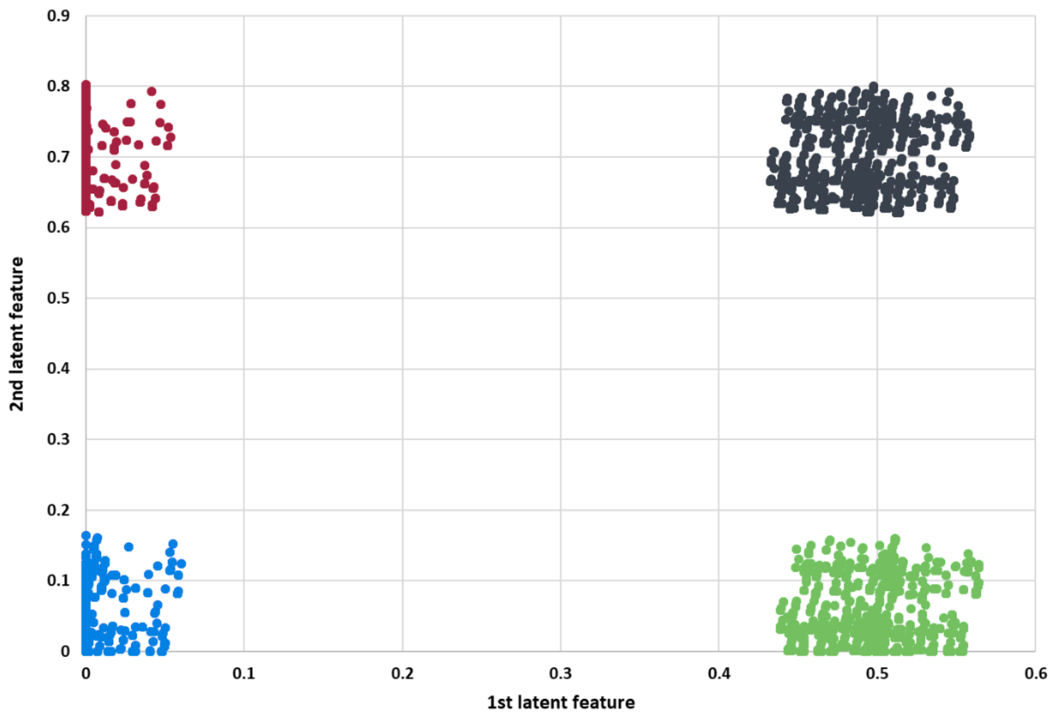


FIGURE 11: THE BUYING PATTERN OF CLUSTER BLUE

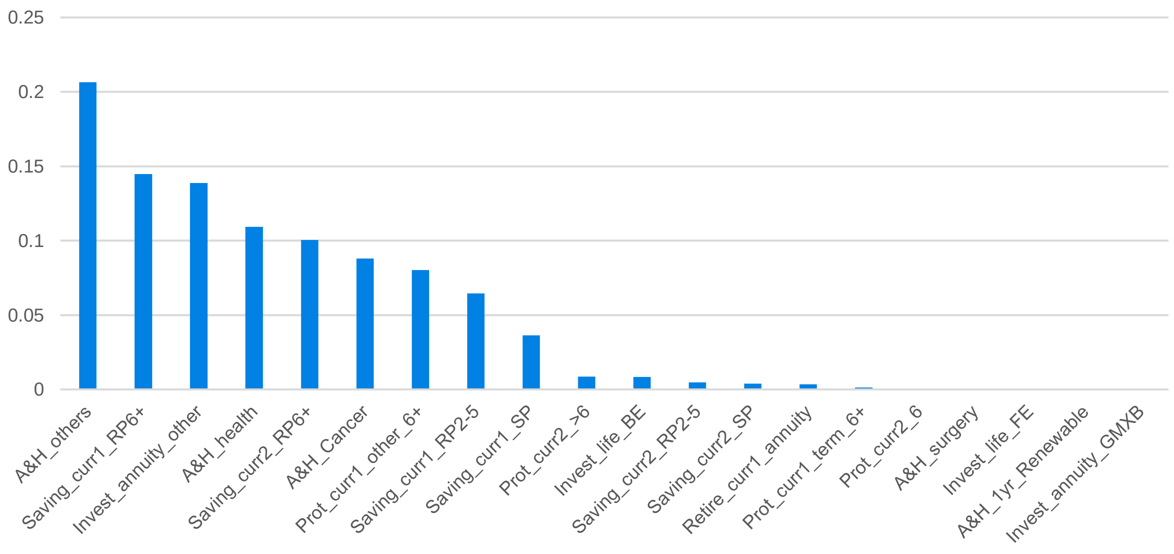


FIGURE 12: THE BUYING PATTERN OF CLUSTER GREEN

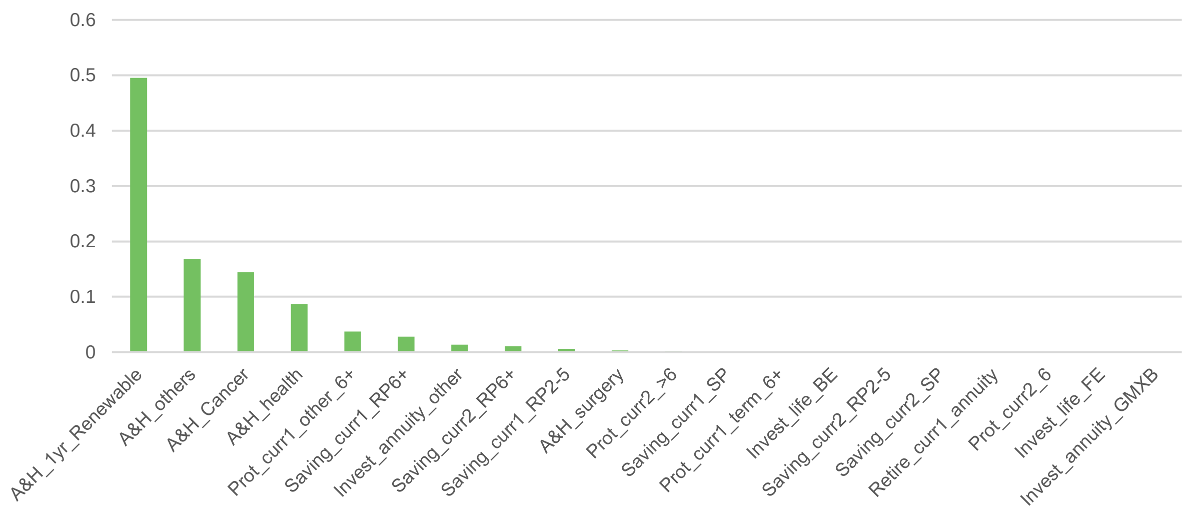


FIGURE 13: THE BUYING PATTERN OF CLUSTER RED

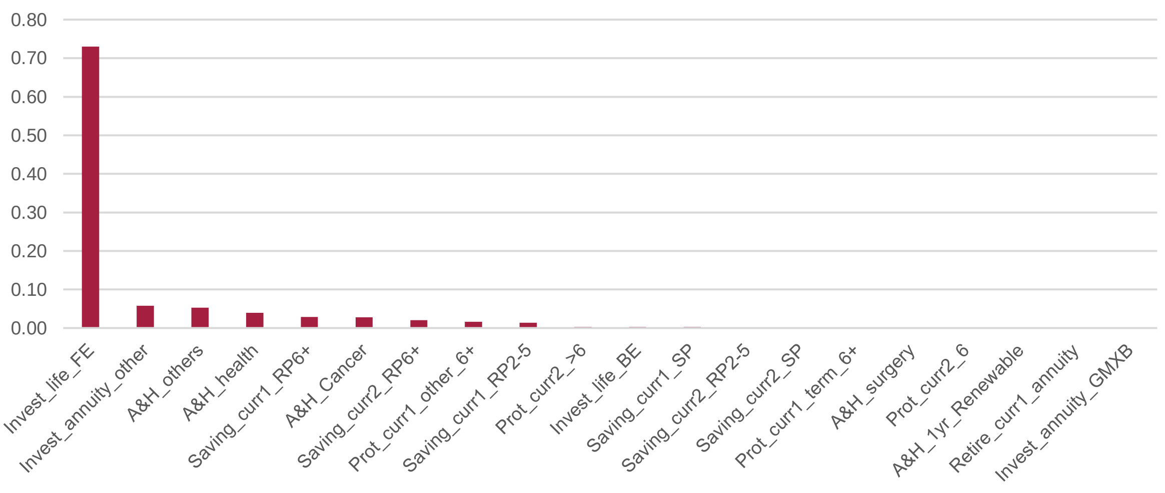
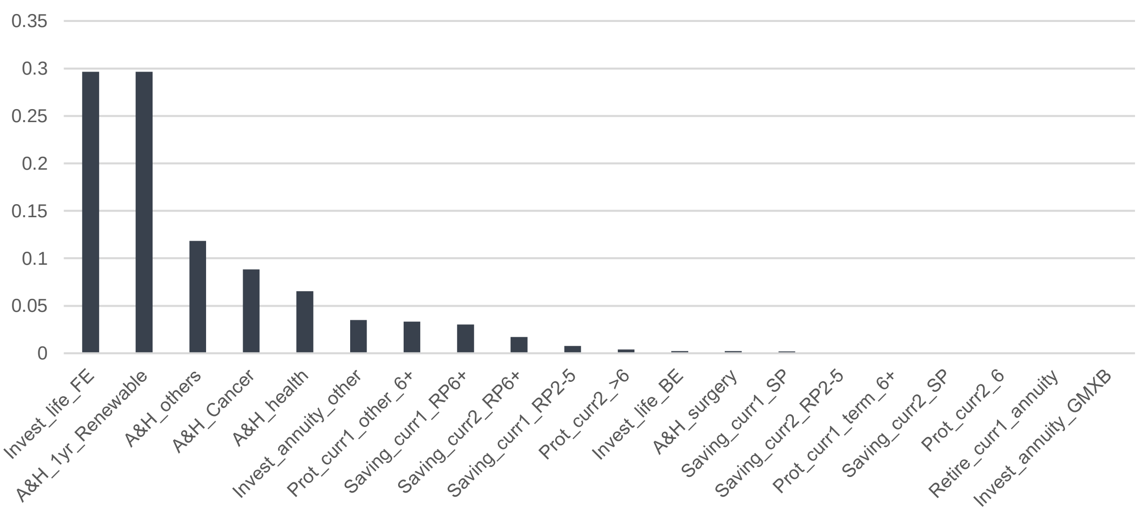
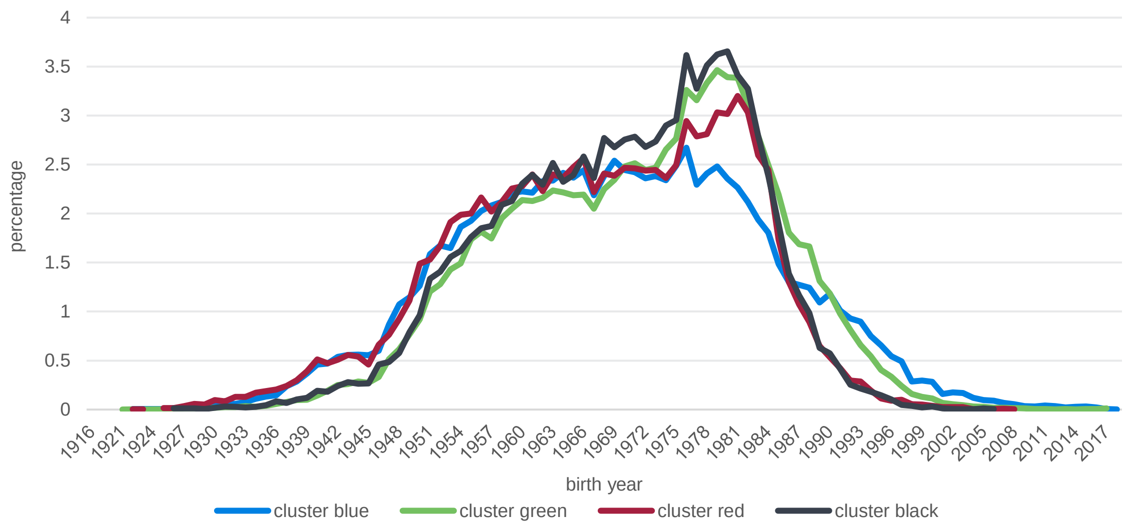


FIGURE 14: THE BUYING PATTERN OF CLUSTER BLACK



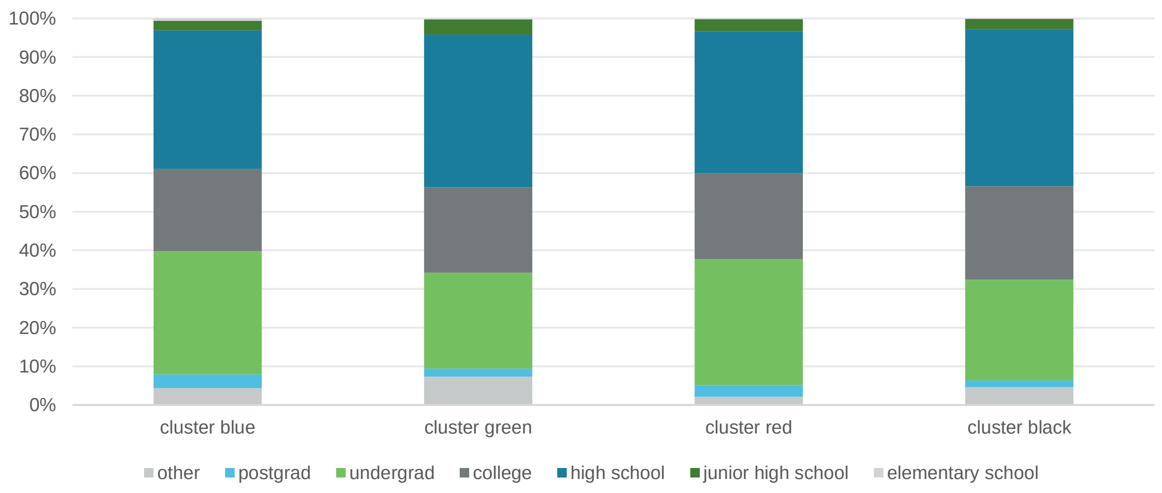
The buying patterns of the four groups were indeed very different. The red group preferred Invest_life_FE. The green cluster liked A&H_1yr_Renewable. Invest_life_FE and A&H_1yr_Renewable stood out in the black group. The blue group obviously had more diverse interests in various kinds of product types including savings and investments. It is then clear that the x-axis of the latent space measured degree of preference for A&H_1yr_Renewable, while the y-axis measured the preference of customers for Invest_life_FE. The A&H_1yr_Renewable is a cheap and popular rider for accident and health protection. The Invest_life_FE is a relatively risky investment product. Based on the buying patterns of those four groups, we can infer that the higher value of y-axis a customer has, the more willing that customer will be to make a risky investment.

FIGURE 15: THE DISTRIBUTIONS OF BIRTHDAYS OF THE FOUR CLUSTERS OF CUSTOMERS



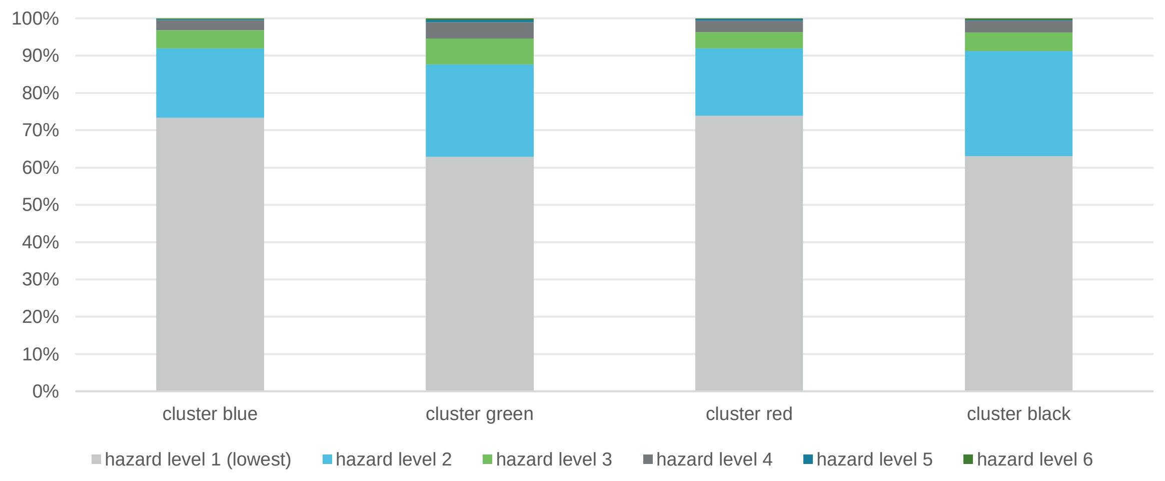
Regarding age, each group had a roughly bell-shaped distribution. The blue and red groups had relatively higher populations for the elderly. All groups except the blue peaked inside the interval of 1975 to 1985, which represented policyholders aged 35 to 45 in 2020. The cluster blue was older, on average. This was consistent with the previous discovery that the cluster blue had the most diverse investment and savings plans in force and that the cluster red preferred relatively risky investments.

FIGURE 16: THE BREAKDOWN OF EDUCATION LEVELS OF CUSTOMERS



As for education levels, we found that the clusters blue and red were more educated as there were more customers with postgraduate and undergraduate degrees. Data suggest that relatively risky investment and diverse products were purchased more by highly educated policyholders.

FIGURE 17: THE BREAKDOWN OF OCCUPATION HAZARD LEVELS OF CUSTOMERS



We found also that clusters red and blue had occupations of relatively low risk. It makes sense that occupations of the cluster green and cluster black groups had higher risk as they bought accident and health rider products.

In summary, we found that the MF model with two latent dimensions internally segmented the customers into four groups of distinct features. Those groups of policyholders had different buying patterns and demographic profiles. The MD model not only made more accurate and satisfactory recommendations but also offered the possibility to develop strategic marketing plans.

5.1.2 Restricted Boltzmann Machines

In our tests of the RBM approach, we started with training a Bernoulli RBM with basic *1-step contrastive divergence* to our client's data, and after some fine-tuning,[10] applying techniques such as *persistent contrastive divergence*, *sparsity targets*, and *momentum* the model achieved reasonable convergence. For the 20 product types in our test data, we found that 6-12 hidden dimensions and a sparsity target between 0.25-0.4 would yield reasonable outcomes. We chose 10 as the number of hidden dimensions. In our test sets the RBM model scored about 30%-32% precision (success rate) and recall (coverage) overall.

FIGURE 18: SUCCESS RATES AND COVERAGE RATIOS OF RBM

10 latent dimensions	Popularity	success rate (precision)	coverage ratio (recall)
A&H_1yr_Renewable	33.7	52.3	63.7
A&H_others	15.26	53.7	37.8
A&H_health	8.33	19.8	21.0
Saving_curr1_RP6+	4.77	14.9	18.2
A&H_Cancer	11.48	12.3	26.4
Invest_life_FE	11.73	10.1	24.9
Saving_curr2_RP6+	2.76	12.3	15.0
Prot_curr1_other_6+	4.22	14.1	10.8
Invest_annuity_other	4.26	1.3	24.0
Saving_curr1_RP2-5	1.54	8.0	5.0
Saving_curr2_SP	0.08	1.4	0.9
Invest_life_BE	0.24	16.7	0.5
Overall rate		30.2	32.5

For recommendations excluding prior purchases, the corresponding precision/recall rates were about 12-13%.

The weights of an RBM are meaningful—they represent the strengths of the positive and negative relationships between individual visible nodes (products) and hidden nodes (latent features). If we consider every node in the hidden layer a customer behavior type, its corresponding weights form a vector, and the strongly positive components and strongly negative components represent, respectively, the products those kind of customers tend to buy or not to buy, modulo the effects of individual customer biases and global product biases (or popularities). With an RBM model with two hidden nodes we obtained similar customer types (or latent attributes) as we did with an MF model.

FIGURE 19: THE GRAY-SCALE PLOT OF THE WEIGHT MATRIX OF RBM WITH TWO HIDDEN DIMENSIONS

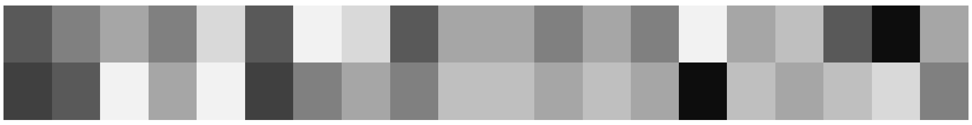


Figure 19 is a gray-scale plot of the real-valued weight matrix we trained for two hidden dimensions. The 20 columns represent 20 product groups, including such products as life, accident & health (A&H), cancer, investment, annuities, and savings. Each of the two rows represent a hidden node, and can be considered a principal customer behavior type. Each bright square represents a strong relationship between the specific node and product, and the very dark squares represent negative relationships. On the top row, for example, the 5th, 7th, 8th and 15th products from left represent:

- [Layer 1] A&H_others, A&H_Cancer, Saving_curr2_RP6+, Invest_**life_FE** (Life insurance with investment components)

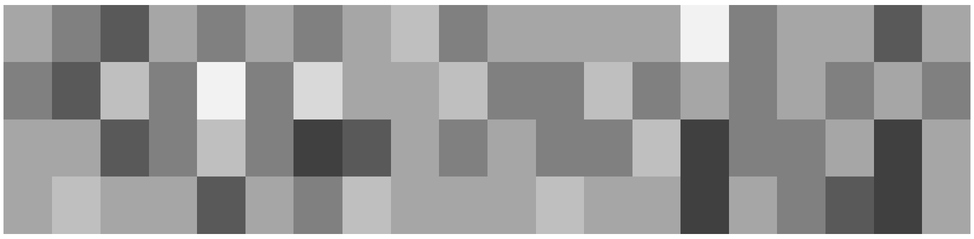
  The last one is a popular category. On the bottom row, the 3rd, 5th and 19th products from left represent products from the categories.
- [Layer 2] A&H_health, A&H_others, **A&H_1-yr_Renewable**,

  The last one is also a popular product and representative of a kind of customer behavior.

Figure 20, on the other hand, represents the weight matrix for an RBM with four hidden dimensions we trained, with the four rows corresponding to the four hidden nodes. The main grouped products we mentioned above for two hidden dimensions are split further:

- [Layer 1] Invest_life_FE
- [Layer 2] A&H_health, A&H_Cancer
- [Layer 3] A&H_1-yr_Renewable
- [Layer 4] Saving_curr2_RP6+

FIGURE 20: THE GRAY-SCALE PLOT OF THE WEIGHT MATRIX OF RBM WITH FOUR HIDDEN DIMENSIONS



In Figure 20, Layer 1, represented by *Invest_life_FE,* and Layer 3, represented by *A&H_1yr_Renewable* are what have been described above. We may interpret the new Layer 2 and Layer 4 as representing those customers who show strong concerns about their personal health and financial well-being, respectively.

5.1.3 Autoencoders

The plain autoencoder, a three-layered neural network, can be considered a nonlinear generalization of MF. Applying autoencoder to our data we saw that more diverse products were recommended, though the explorations were somewhat complementary with MF.

FIGURE 21: SUCCESS RATES AND COVERAGE RATIOS IN AUTOENCODER

	POPULARITY	SUCCESS RATE	COVERAGE RATIO
A&H_1yr_Renewable	33.7	51.83	64.57
A&H_Cancer	11.48	NA	0
Saving_curr1_RP6+	4.77	6.16	5
A&H_others	15.26	55.32	47.42
Invest_life_FE	11.73	NA	0
A&H_health	8.33	17.51	9.13
Prot_curr1_other_6+	4.22	15.56	15.8
A&H_surgery	0.24	NA	0
Invest_annuity_other	4.26	2.33	23.25
Saving_curr2_RP6+	2.76	14.47	11.86
Saving_curr2_RP2-5	0.12	NA	0
Invest_life_BE	0.24	0.65	34.58
Saving_curr1_SP	0.76	NA	0
Prot_curr2_6	0.01	NA	0
Saving_curr2_SP	0.08	NA	0
Saving_curr1_RP2-5	1.54	14.27	15.19
Prot_curr1_term_6+	0.1	NA	0
Retire_curr1_annuity	0.06	2.97	57.72
Invest_annuity_GMXB	0	NA	0
Prot_curr2_>6	0.34	NA	0
Overall ratio		29.45	32.22

5.2 BUNDLE SALES POTENTIAL

The products provided by an insurer are usually not independent but correlated. Customers often bought several products together to meet their needs. Such information is actually digested by CF models and so customized recommendations can be made. It is useful to present such information in a product-oriented point of view.

Bundle sales or cross sales to policyholders may be achieved based on the knowledge that combinations of products are often purchased together. It is natural to apply *association rule mining* for this purpose. In association rule mining, there are a few important statistics. A useful measure is called *support,* which denotes the population of the combination of products among all purchasing records. The *confidence* metric measures how likely purchasing a set of products implies purchasing of another. Therefore, confidence is a conditional probability and such implication between two sets of products is called an association rule. A meaningful association rule between two sets of products is more than a rule of high conditional probability because these two sets of products must be correlated. To measure the degree of correlation, another metric called *lift* is introduced. Lift is a positive number, and the rule is highly correlated if its lift is away from 1.

In insurance, a customer may not buy several products in one go. Therefore, we define the association rule as an implication between two sets of products, where the first set of products contains all the products ever bought, i.e., these products do not need to be purchased in the same day. Considering the recency of the data, we use the purchase records in these two years from 2018-01-01 to 2020-09-30. We show here the interesting rules found by the association rule mining, how likely a policyholder who bought A would also buy B (and C)?

FIGURE 22: SAMPLE OF THE STATISTICALLY RELEVANT ASSOCIATION RULES

After	Would buy		LHS popularity	RHS confidence
A	B	C		
A&H_1yr_Renewable	A&H_others		65	35
A&H_1yr_Renewable	Invest_life_FE		65	17
A&H_1yr_Renewable	A&H_others	Invest_life_FE	65	7
A&H_others	A&H_1yr_Renewable		30	77
A&H_others	A&H_1yr_Renewable	A&H_Cancer	30	32
Invest_life_FE	A&H_1yr_Renewable	A&H_others	23	20
A&H_Cancer	A&H_1yr_Renewable		22	86
Saving_curr1_RP6+	A&H_Cancer		9	28
Saving_curr1_RP6+	Invest_life_FE		9	18
Invest_annuity_other	A&H_1yr_Renewable		8	35
Prot_curr1_other_6+	A&H_1yr_Renewable		8	67

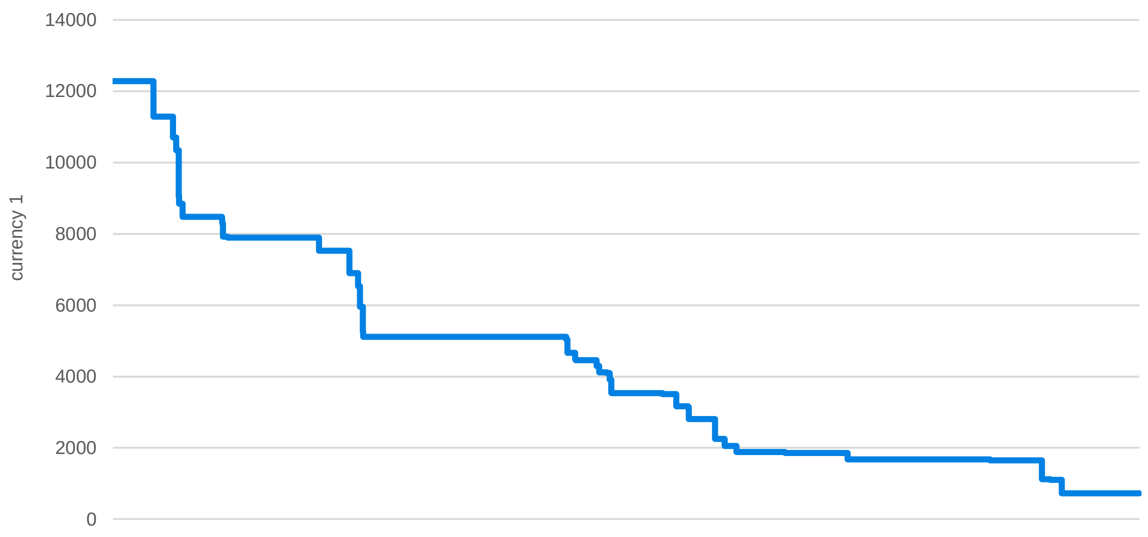
In Figure 22 above several statistically relevant rules are listed. The left-hand side of the association rule is listed in column A. The right-hand side of the rule is listed in column B alone or column B and C depending on how many product types on the RHS of the rule. Figure 22 provides a snapshot of the current sales structure of the company. For example, the most popular rider recently is A&H_1yr_Renewable. If a policyholder is in force with this rider, it is most likely that the rider is associated with A&H_others. There are also more complicated rules. For example, there are 30% population of policyholders purchasing A&H_others, and there are 32% among them also buying A&H_Cancer with A&H_1yr_Renewable rider.

5.3 CUSTOMER VALUE

The purpose of a recommender system is to create leads and generate additional revenue. In this section we estimated the revenue created by using the recommender system built by an MF model with seven latent dimensions. In so doing, we defined the value of the customer as the amount of expected potential additional profit gained from policyholder from additional sales. The reason why we chose MF with seven instead of two latent dimensions is that the latter gives more balanced recommendations; there is no big hierarchy between success rates and coverage ratios and less concentration on popular products.

The expected potential lifetime profit of each customer was estimated by the product of the probability of success (success rates), mean lifetime premium, and *assumed* profit margins of the products. For example, suppose two product types A and B are recommended to a customer. Each recommendation generates revenue potentially. The expected profit generated by recommending product A to the customer is estimated by the premium times the profit margin of product A times the success rate of recommending product A. Summing up the estimated amount of potential profit of recommending product B following the similar line, the value of the customer in question can be estimated. In principle the customer value is a sum of more products if a higher n of top-n recommendations is adopted. However, the probabilities of success drop with an increasing number of products recommended. So, one needs to explore a best recommending strategy of practical implementation.

FIGURE 23: CUSTOMER VALUES IN DESCENDING ORDER



Having obtained the values of each customer, we sorted the customers according to their values in descending order and obtained the above chart. Among all policyholders, there were around 1,000 people in the group with highest values, at around $12,000. We noticed also that policyholders were segmented into different groups of different values represented by the horizontal segments of the curve above.

Such a definition of customer values is *dynamical*. Using a moving window approach, one is able to update the customer values frequently. With customer segmentations phrased in terms of values, the company can explore promotional strategies that maximize benefits. For example, one can develop rewarding plans for those high-value customers and seek promotions to lift the values of the low-value policyholders.

Summing up the values of each customer, one is able to estimate the total potential revenue generated by the recommender system. With 300,000 policyholders, the annual premium in force was around $51 billion. The expected potential new sales premiums created by the recommender system was $4.7 billion. This amount of potential premium translated to $1.3 billion of potential revenue considering the profit margins. This significant amount of potential profit was generated by considering the policyholders as a whole and then carefully taking care of the individual policyholders by making customized recommendations. For the purpose of comparison, we estimated also the potential profit generated by a trivial recommender system which only recommends the most popular product. The number was significantly lowered to $0.1 billion, which demonstrates the power of a recommender system.

5.4 LIMITATIONS OF THE STUDY

Recommender systems have proven to be useful in contexts such as e-commerce and web search engines. They help guide potential customers to the information being sought while benefiting the service provider by increasing potential revenue. Although many different algorithms and approaches for recommender systems have been developed and applied in practice in the past decades, some key challenges to recommender systems remain.

These challenges are briefly described below, although our study will not explore these topics further.

5.4.1 Cold start

CF suffers from a so-called cold-start problem, which refers to a situation when a new customer enters the system, or a new product is added to the catalogue. In such cases, the recommender system is unable to make any recommendation because there is no purchase history. In general, customers (products) are said to be cold if there is limited interaction between products and customers.

The problem may be partially overcome by recommending the most popular products or asking the new customer to rate a given set of products at the beginning as a way to solicit preferences.

5.4.2 Sparsity

Each customer usually rates only a small subset of the available products in most recommender systems. This is particularly so for insurance products. One can be making a huge mistake if one assumes that a customer who has not clicked or rated a product equates disinterest.

We need to ignore the missing values because the more likely explanation is that s/he has not discovered the product yet. This will result in a very sparse rating matrix. For this matter, we can use implicit feedback such as number of clicks and shares to gain more data on the customers.

5.4.3 Scalability

As the number of customers and products grow, the recommender systems are facing a great challenge to handle enormous data in a graceful manner. Despite the large amount of data, the recommender systems need to provide real-time suggestions to keep customers engaged.

With such large-scale datasets, some proposed using clustering, dimension reduction, and approximation mechanisms to deal with this scalability problem and speed up the recommendation algorithms.

5.4.4 Over-specialization

Recommender systems tend to recommend products that are similar to what customers liked in the past. While this approach produces safe recommendations, it only recommends products that are closely related to the customer profile and do not suggest novel products. This may prevent customers from discovering new products that they may actually like had they known about them.

In order to overcome the problem of over-specialization, we can introduce some level of randomness into the recommenders by performing sampling, similarity measures, and dimension reduction.

6. Conclusions and Discussion

6.1 CONCLUSIONS

Partnered with a major local insurer, we tested three approaches to CF—the archetypical building blocks underpinning many recommender systems in use today: MF, autoencoders, and RBMs. We found that these algorithms had similar Success Rates and Coverage Ratios, but differed in the breadth of the products they recommended. With back testing, we found improvements to the naïve *recommend-the-most-popular-products* approach, and calculated the dollar values in improved sales.

Each of these models has a hidden layer in the middle that can be interpreted as embeddings for the customers and the products. We chose the MF approach to explore this more deeply. As a by-product to training the recommender model, customers are segmented by clustering their embeddings in the hidden layer. We found major groupings of clients with different distributions of education levels, occupation classes, and ages in addition to the very distinct buying patterns. Two customer attributes emerged as embedding dimensions from this study: *preference for risky products* and *preferences for cheap A&H products.* Similar patterns emerged in the hidden dimensions for the RBM model.

As a proof of concept, the future values of the client to the insurance company were estimated based on product recommendations and assumed profit margins. Potential values of the policyholders ranged from $2,000 to $12,000. Policyholders were segmented into groups of similar values. Summing up values of the customers, we estimated that the recommender system generated $1.3 billion of potential lifetime profit from 300,000 existing policyholders.

6.2 DISCUSSION

In this report, we considered only cross sales to the existing policyholders. While retaining customers and maximizing the potential revenues from the existing customers are crucial, acquiring new customers is equally important. The lack of ability to make customized recommendations to new customers is the Achilles' heel of the traditional collaborative filtering models. A collaborative filtering model has no idea how to make recommendations to new customers simply because they have no history of past purchases. How to make high-quality personalized recommendations to the new customers in the system according to limited interactions and information is then an issue for future work.

For starters, we may take inspiration from well-known user cases mentioned above such as YouTube, Airbnb, UberEats, or Amazon, whose recommenders are highly intertwined with all parts of the systems, and customer browsing histories are recorded and analyzed in detail to provide hints of user preferences even before any purchase. For Amazon, a click of a product or a keyword typed into the search bar would be a strong signal of interest. Such an experiment, though, would involve redesigning parts of the existing system and testing them online and is out of the scope of this report.

Another issue of paramount importance for practical implementation is how to extend the system to include the agent logistics. Unlike in the case of e-commerce, interactions between customers and products are not in a direct manner. In insurance, there are various selling channels. In particular, the most popular one in most places is still the agent channel. Insurance products are presented to customers via agents and customer backgrounds and needs are gathered through interviews between agents and customers. Therefore, we not only need to consider the pairing between products and customers but also the pairing between products, customers, and agents. This is a three-body problem somewhat similar to the situation of food delivery applications like *Berets*. However, in the case of insurance it is more complicated as there is no clear optimization to be made. In the case of UberEats, the time gaps between repeated purchases serves as a proxy or *implicit feedback* for customer satisfaction. Though such a metric would not always work naively for insurance products, records of customer interactions in the forms of complaints, inquiries, and so on may also enrich system knowledge for individual customers. Moreover, a mature insurance recommender system also needs to deal with the real-time feedback from agents in order to offer personalized recommendations that fit the needs of new customers.

Though we only tested basic collaborative filtering algorithms, most of the working recommender systems we have surveyed are hybrid in nature, each with its own specific setups tailored to the system's specific needs and data availability. It is not our intention to dismiss more traditional approaches such as rule-based systems or content-based filtering. Rather, a finely tuned recommender system may include elements of both to take advantage of all information available. On the other hand, it is well-known that YouTube uses *deep* (multileveled) neural networks for their recommender engine. Attractive as it is, training a deep network typically requires orders of magnitude more data than is currently available to us in this project.

Some existing *direct-to-customer* insurance platforms do use social media, search engines, or mobile apps to place advertisements or generate leads.[17] Such data would be highly irregular but could be a topic for future research.

In our tests, we treated each client as *static* and did not explore the dynamical or contextual behaviors. In the UberEats example, each individual had preferences for different foodstuffs at different times of the day and in different situations, or contexts. Similarly, the same customer at different stages of his or her life would have different insurance needs. In our datasets we saw that, for example, many clients purchased their first Life and Health products within three or four years of each other, but waited typically seven to 15 years to purchase their first annuity. A more detailed study of such contextual information could help improve the predictive accuracies.

During back-testing, we split training data (about 20 years) and testing data (about two years) unevenly. This was partly for convenience, but also corresponded to the time-horizon of typical insurance agents who would hope to know which products the clients they are working with would purchase not too far into the future. We should understand that even if a customer has not made a single purchase in the testing period, it doesn't mean that he or she doesn't want or need the products. Timing, or in more general terms, context, as we discussed above, is also an important factor influencing customer decisions. Moreover, changes in insurance trends happen whenever there is a substantial environmental change. For example, COVID-19 certainly arouses concern about current health insurance protections. New regulations against the existing products may push people buying the affected products. Such considerations suggest that recent purchase records should have higher weights than the old ones.

We chose to score our models with precision (success rate) and recall (coverage). These two quantities are well-known in machine learning standard practice and give more meaningful metrics than plain accuracy for uneven data. For instance, given that most customers purchased only a small number of products out of many, the constant *zero* recommender, which naïvely recommends *no purchase* for all products for all customers, would score high in terms of accuracy. That is not what we want to aim at. Instead, the *precision*, defined as the number of True Positives divided by Total Positives, measures how well the model's guesses fit the truth, irrespective of how many potential purchases were left out. On the other hand, the *recall,* defined as the number of True Positives divided by Total Truths, measures how well the model finds or recovers the potential true purchases, irrespective of how many wrong guesses it makes. These two quantities, taken together, form a better picture of how useful the model is. Usually there is a trade-off, the higher the precision, the lower the recall, and vice-versa, and the modeller would tune and optimize their models to suit their needs.

7. References

1. Ekstrand MD, Riedl JT, Konstan JA. Collaborative filtering recommender systems. Foundations and Trends in Human-Computer Interaction. 2011 Feb 1;4(2):81–173
2. Lops P., de Gemmis M., Semeraro G. (2011) Content-based Recommender Systems: State of the Art and Trends. In: Ricci F., Rokach L., Shapira B., Kantor P. (eds) Recommender Systems Handbook. Springer, Boston, MA
3. Miquel Montaner, Beatriz López, Josep Lluís de la Rosa (2003) A Taxonomy of Recommender Agents on the Internet, Artificial Intelligence Review 19
4. Terveen, L., & Hill, W.C. (2001). Beyond Recommender Systems: Helping People Help Each Other
5. Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," in Computer, vol. 42, no. 8, pp. 30-37, Aug. 2009, doi: 10.1109/MC.2009.263
6. Funk, Simon. "Netflix Update: Try This at Home"
7. R. Salakhutdinov, A. Mnih and G. Hinton, "Restricted Boltzmann machines for collaborative filtering", Proceedings of the 24th international conference on Machine learning, pp. 791-798, 2007
8. Sedhain, S., Menon, A. K., Sanner, S., & Xie, L. (2015). AutoRec. Proceedings of the 24th International Conference on World Wide Web - WWW ’15 Companion. doi:10.1145/2740908.2742726
9. Carreira-Perpiñán, M.Á., & Hinton, G.E. (2005). On Contrastive Divergence Learning. AISTATS
10. Geoff Hinton, "A Practical Guide to Training Restricted Boltzmann Machines, Version 1”, August 2010
11. DigFinGroup (July 8, 2019) Seasonalife’s comparison tool first step to ‘insurance robo’
12. DigFinGroup (November 4, 2019) Will Bowtie make it?
13. StartupInno (April 13, 2020) Bowtie Insurance’s Business Story, Let me tell you myself (in Chinese)
14. BusinessFocus (May 9, 2019) [Disrupting an old industry] Actuary founded start-up after heart-attack, creating the first “virtual insurance” in Hong Kong (in Chinese)
15. Reuters (September 5, 2019) Prudential buys Assurance IQ for $2.35 billion in new tech bet The Economic Times (September 10, 2019) Prudential Financial to acquire Assurance IQ, an insurtech for $2.35bn
16. PR Newswire (July 8, 2020) Picwell Launches Revolutionary Product Aimed at Taking the Guesswork Out of Health Savings Accounts
17. Technical.ly (January 15, 2016) Picwell wants to be the ‘Moneyball of the insurance world’
18. Washington Post (December 8, 2014) Can technology pick the perfect health plan for you?
19. BusinessInsurance (November 22, 2013) Towers Watson expands private exchange with $215M Liazon purchase
20. Press Release (July 30, 2019) Willis Towers Watson completes acquisition of TRANZACT
21. Capital.com (July 13, 2020) GoHealth IPO: everything you need to know and even more
22. Reinsurance News (March 2, 2020) Chaucer eyes more efficient underwriting service with Artificial Labs alliance
23. Digital Insurance Agenda (May 4, 2017) SPIXII: Making insurance simple, accessible and personal for everyone
24. Press Release (2019) Taiwan Life Insurance Company’s “smart helper iKash”, a precision digital marketing tool for visitations (in Chinese)
25. Taipei Times (December 5, 2018) Cathay Financial unveils AI chatbot
26. iThome (December 4, 2018) Cathay Financial’s newly launched AI customer service chatbot “Alpha” understands customers better every day, the key being this group of chatbot tag trainers (in Chinese)
27. Digital Insurance Agenda (October 15, 2017) Ladder: Instant, simple and smart life insurance
28. Medium (January 10, 2018) Announcing the Ladder API
29. PR Newswire (December 12, 2019) Sproutt Launches; Uses Data and AI to Finally Reward Life Insurance Customers Who Live Healthy https://sproutt.com/about-us
30. Netflix Tech Blog (April 6, 2012) Netflix Recommendations: Beyond the 5 stars
31. Carlos A. Gomez-Uribe, Neil Hunt (December 2015) The Netflix Recommender System: Algorithms, Business Value, and Innovation
32. James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet (September 2010) The YouTube Video Recommender System, RecSys2010
33. Renjie Zhou, Samamon Khemmarat, Lixin Gao, The Impact of YouTube Recommendation System on Video Views
34. Prudential Life Insurance Assessment, Kaggle, Nov 2015

CONTACT

Wing Wong
wing.wong@milliman.com

Yun-Tien Lee
yuntien.lee@milliman.com

Shenglan Ko
shenglan.ko@milliman.com

Kwan Yau
kwan.yau@milliman.com