Maximizing Customer Lifetime Value using Stacked Neural Networks: An Insurance Industry Application

***Abstract*—Customer Lifetime Value (CLV) has been widely used as a key performance metric to evaluate retention strategies in insurance industry. On the other hand, current applications for retention include cross-sell using machine learning based recommendation systems. Many current recommendation systems find similar preferences or sequential next item recommendations, but do not maximize CLV. Moreover, these methods may not consider temporal lifetime patterns resulting in a gap between customer lifetime maximization and insurance product recommendation.**

This paper proposes a two-stage neural network architecture. The Stage-I neural network uses a self-attention mechanism and a Metric Learning (CML) to generate product recommendations. The Stage-II neural network uses a neural network-based survival analysis to infer insurance product recommendations that maximize customer lifetime. The proposed stacked neural network model can be used as a generative model to explore different cross-sell scenarios.

The applicability of the proposed recommendation system is evaluated using transactional data from an Australian insurance company. We validated our results against a state of the art self-attention recommendation system, successfully extending its functionality to include lifetime value.

***Index Terms*—Recommender systems, Neural Networks, Customer relationship, Customer Lifetime Value, Self-Attention Mechanism, Customer retention, Insurance product recommendation**

I. Introduction

The growing number of churning customers in the insurance industry has prompted insurers to look for new customer retention strategies to keep and engage their current customers and prevent declining revenues. Efforts in customer retention have a larger effect on profits than actively acquiring new customers. It is common for insurers to lose up to 15% of their clients each year. Reducing defections by just 5%, would increase the average company's growth rate by more than 50% [1]. One approach to solve the problem involves estimating customer value to identify which customer relationships will increase and which will decrease in value or revenue for insurers. For this aim, Customer Lifetime Value (CLV), which is the present value of all future profits coming from a customer over its lifetime with a company, serves as a valid metric to evaluate customer value. Customer Lifetime Value has become a topic of interest in the scientific community as an option to address the problem of customer retention and churn prediction [2] [3].

CLV models rely on customer income at every stage of their life cycle to make good estimations, using an array of techniques from survival analysis [4], [5], [6], [7] and data mining [3], [8]. In [2], a Pareto/NBD model is used to predict CLV. The first component of the model predicts the number of future transactions. Then, another component provides an estimated profit for each of the predicted future transactions. After that, CLV is computed with the outputs of these components. These methods focus on estimating CLV rather than optimizing it.

Recommender systems (RSs), which have been very profitable for the digital business, can play a bigger role in the insurance industry specifically. Currently some insurance companies recommend policies to a client for cross-sell or up-sell opportunities with the help of recommender systems. A large variety of methods are used to obtain recommendations such as Collaborative Filtering (CF) [9], Content filtering [10] and hybrids [11]. RSs consider user preferences, product data and transactional data to suggest next best items for purchase.

In recent years, attention-based recommender systems have been proposed extensively [12], [13], [14]. Shoujin et al. [15] propose an attention-based transaction embedding model (ATEM) to recommend ads for web sessions. In [16] authors propose an attention-based user behavior modeling framework called ATRank for recommendation tasks. In [17] and [18], self-attention mechanisms are used to infer the item-item relationship from users' historical interactions for the retail industry.

Considerable progress has been made in deep learning techniques applied to recommendation tasks. One common approach is the Convolutional Neural Network (CNN) that can be used to extract features from text, video, and audio datasets [19] [20] [21]. Ansell et al. [22], apply life-stage segmentation and survival analysis based on Cox's linear regression to identify cross-sell opportunities in 10,979 UK customers of a large insurance company. They show that it is possible to identify which customers are more likely to re-purchase and to predict the time frame in which they will re-purchase.

Other techniques that aim to optimize CLV rely on Reinforcement Learning to do it. In [23], the authors propose a framework that uses Reinforcement Learning (RL) to make Personalized Ad Recommendations (PAR). Hallak et al. [24] propose an architecture that uses RL for recommendation systems, that enables easy modular hands-free LTV optimization using matrix factorization methods.

In this paper, we propose a recommender system that maximizes CLV for an insurance company, which outputs risk and survival curves for the top n recommended products for each customer. The key contributions of our work are the following:

1) A method that captures both product recommendations and CLV optimization for the insurance industry. It considers product features to infer an increase or decrease of CLV, easing the exploration of different products that might extend or reduce lifetime and increase or decrease profits.

2) The method captures the non-linear relationship between recommended products and Customer Lifetime Value, resulting in more realistic predictions.

The rest of this paper is structured as follows. Section II shows the data processing steps. Section III describes Stage I of the method, discussing the Self-Attention Recommender system. Section IV details Stage II where a CLV model via non-linear survival analysis is used. Section V shows the proposed method to maximize CLV via recommended system. Section VI goes on to show Experimental Results. Section VII presents the Conclusion, and finally the Section Appendix shows key mathematical ideas in the proposal.

II. Data Processing

We will now show how the insurance dataset is cleaned and transformed for use in our recommender system. Consider a set of users and items denoted by $\mathcal{U} = \{u_1, \ldots, u_m\}$ and $\mathcal{S} = \{s_1, \ldots, s_n\}$. Let $\boldsymbol{X} = \boldsymbol{X}_p \cup \boldsymbol{X}_u$ be the dataset that contains item features and user features, where $\boldsymbol{X}_p \in \mathcal{R}^{(n\times m)\times d}$, $\boldsymbol{X}_u \in \mathcal{R}^{(n\times m)\times f_u}$ and f_u represents the number of user features. Let o_{end} be the cutoff time for our dataset $\mathcal{X}$. Each user $u_i \in \mathcal{U}$ has a subscription interval denoted by $I_{s_j}^{u_i} = [t_{s_j}^{u_i}\ o_{s_j}^{u_i}]$ if the user has unsubscribed at o_{end} or $I_{s_j}^{u_i} = [t_{s_j}^{u_i}\ o_{end}]$ in the case he is still subscribed at o_{end}. The subscription and unsubscription time of user u_i to item s_j is denoted by $t_{s_i}^{u_i}$ and $o_{s_j}^{u_i}$ respectively. The time period of the subscription is $|I_{s_j}^{u_i}| = o_{end}\cdot(1-e_{s_j}^{u_i}) + o_{s_j}^{u_i}\cdot e_{s_j}^{u_i} - t_{s_j}^{u_i}$. The parameter $e_{s_j}^{u_i}$ is the subscription status for user u_i to item s_j. The values $e_{s_j}^{u_i}$ will be 1 in the case it is subscribed and 0 if the user has no active subscription at time o_{end}.

A purchase history of each user u_i is represented by the 3-tuple $\gamma_{u_i} = (I_{s_j}^{u_i}, \mathcal{S}_{u_i}, \boldsymbol{X}_{u_i})$ where $\mathcal{S}_{u_i} = \{s_1^i, \ldots, s_{|\mathcal{S}_{u_i}|}^i\}$ represents a sequence of items in the order that the user u_i purchased the items before o_{end}. The *purchase history matrix* $\boldsymbol{X}_{u_i} \in \mathcal{R}^{l\times d}$ (obtained from $\boldsymbol{X}_p$) has data containing summary features of the user u_i l-th purchase up to time O_{end}. The first purchase $\boldsymbol{x}_{u_i}^1 = [x_{1,1}, \ldots, x_{1,d}]$ is a d-dimensional embedding vector that has d features of an item in set $\mathcal{S}$. The *purchase history matrix* is built from all the l d-dimensional embedding vectors:

$$\boldsymbol{X}_{u_i} = \begin{bmatrix} \boldsymbol{x}_{u_i}^1 \\ \boldsymbol{x}_{u_i}^2 \\ \vdots \\ \boldsymbol{x}_{u_i}^l \end{bmatrix}. \tag{1}$$

This matrix is the input to the Recommender System described next.

III. Stage I: Self-Attention Recommender System

The recommender system uses a self-attention model to deal with sequential data and a preference model that uses Collaborative Metric Learning (CML) to tackle users' item preferences [25]. The objective is to predict the items that the user u_i will purchase next given their purchase history coming from matrix $\boldsymbol{X}_{u_i}$.

A. Self-Attention Mechanism

The Self-Attention model maps a query and a set of key-value pairs to an output. The query, keys, values, and output are all vectors [26]. This allows us to calculate the global dependencies between users and items.

To construct the Self-Attention model, we first project the purchase history matrix $\boldsymbol{X}_{u_i}$ to the spaces $\mathcal{F}(\boldsymbol{X}_{u_i})$ and $\mathcal{G}(\boldsymbol{X}_{u_i})$, where $\mathcal{F}(\boldsymbol{X}_{u_i}) = \boldsymbol{X}_{u_i}\boldsymbol{W}_{\mathcal{F}}$, $\mathcal{G}(\boldsymbol{X}_{u_i}) = \boldsymbol{X}_{u_i}\boldsymbol{W}_{\mathcal{G}}$ and matrices $\boldsymbol{W}_{\mathcal{F}}, \boldsymbol{W}_{\mathcal{G}} \in \mathcal{R}^{d\times d}$ become the learned weights for query and key respectively, which are implemented as 1x1 convolutions. We calculate the dot products of the query with all keys in $\boldsymbol{\Pi} = \mathcal{F}(\boldsymbol{X}_{u_i})\mathcal{G}(\boldsymbol{X}_{u_i})^T$, where $\boldsymbol{\Pi} = (\pi_{i,j}) \in \mathcal{R}^{d\times d}$. Then, the attention matrix $\boldsymbol{\Psi}_{u_i} = (\psi_{i,j}) \in \mathcal{R}^{l\times l}$ is computed, and now contains the similarities between the l items of user u_i. The similarity between the i-th item and the j-th item is denoted by $\psi_{i,j}$ and obtained by computing the softmax of the attention scores $\pi_{i,j}$ as follows: $\psi_{i,j} = softmax(\pi_{i,j})$. The output $\boldsymbol{\zeta_{u_i}} = (\zeta_1, \ldots, \zeta_j, \ldots, \zeta_l) \in \mathcal{R}^l$ of the Self-Attention module is obtained from the mean of the product of the attention matrix $\boldsymbol{A} = (a_{i,j}) \in \mathcal{R}^{l\times l}$ and the purchase history matrix (i.e., $\boldsymbol{A} = \boldsymbol{\Psi}_{u_i}\boldsymbol{X}_{u_i} \in \mathcal{R}^{l\times l}$) [25],

$$\zeta_j = \frac{1}{l}\sum_{i=1}^{l} a_{i,j}. \tag{2}$$

Following the transformation of $\boldsymbol{X}_{u_i}$, we compute a geometric sequence of timescales so we can add sinusoidals of different frequencies to the input [26]. The time embedding function $\Gamma(t, 2i)$ consists of two sinusoidal signals defined as follows, where t is the time step, and i the dimension: $\Gamma(t, 2i) = sin(t/10000^{2i/\alpha})$ and $\Gamma(t, 2i+1) = cos(t/10000^{2i/\alpha})$.

B. Preference model

We encode the user-item interaction matrix to latent factors. Let the users' and items' latent factors be the matrices $\boldsymbol{U}_i \in \mathcal{R}^{m\times d}$ and $\boldsymbol{I}_j \in \mathcal{R}^{n\times d}$. To obtain the minimum distance between users and items a scoring function is used, which finds the Euclidean distance between each user and item interaction:

$$Score(\boldsymbol{U}_i, \boldsymbol{I}_j) = ||\boldsymbol{U}_i - \boldsymbol{I}_j||_2^2, \tag{3}$$

where $||\cdot||_2^2$ is the L2 norm of the matrix $\boldsymbol{U}_i - \boldsymbol{I}_j$. If user u_i likes item s_j the distance will be small and large otherwise.

C. Objective function

Let $\mathcal{S}_{u_i}^{l+1}$ be the set of k predicted items s_{l+1}^k for user u_i that will be recommended after time o_{end}. The cost function used to train the model minimizes Euclidean distance as in [25], is computed as follows:

$$\boldsymbol{y}_{u_i}^{l+1} = ||\boldsymbol{U}_i - \boldsymbol{I}_{l+1}||_2^2 + ||\boldsymbol{\zeta}_{u_i} - \boldsymbol{x}_{u_i}^{l+1}||_2^2. \tag{4}$$

Note that $\boldsymbol{I}_{l+1}$ and $\boldsymbol{x}_{u_i}^{l+1}$ are the embedding vectors for the next item $s_{l+1}^k \in \mathcal{S}_{u_i}^{l+1}$. Thus the purchase history matrix for the next

$l+1$ purchase is denoted by $\boldsymbol{X}_{u_i} = [\boldsymbol{X}_{u_i}; \boldsymbol{x}_{u_i}^{l+1}]$. In order to predict a set of next items, the pairwise ranking loss is minimized to learn the model parameters [27]. The pairwise ranking loss is defined as in [25],

$$\mathcal{L}(\Phi) = \sum_{(u_i,s_j)\in\mathcal{S}^+} \sum_{(u_i,s_k)\notin\mathcal{S}^+} \max\left(0, \boldsymbol{y}_{u_i}^j + \Delta - \boldsymbol{y}_{u_i}^k\right) + \lambda||\boldsymbol{\Phi}||_2^2, \tag{5}$$

where $\boldsymbol{\Phi} = \{\boldsymbol{X}, \boldsymbol{I}_j, \boldsymbol{U}_i, \boldsymbol{W}_{\mathcal{F}}, \boldsymbol{W}_{\mathcal{G}}\}$ represents the model parameters, and $\mathcal{S}^+$ denotes the set of next k items to recommend to user u_i. The set of items the user does not prefer is denoted by $\mathcal{S}^-$ ($\mathcal{S}^- = \mathcal{S} - \mathcal{S}^+$), Δ represents the difference between the preferred items and unpreferred items. The parameter λ is the regularization parameter. We use L_2 regularization to control the model complexity.

IV. Stage II: Customer Lifetime Value using non-linear Survival Analysis

A. Customer Lifetime Value

The CLV function CLV_{u_i} captures all the purchases that a customer u_i will make, for the n items that the company sells. As in the model proposed in [4], we calculate customer u_i CLV before cutoff o_{end} from the sum of the net cash flows $\mathcal{C}_{s_j}^{u_i}(t)$ (i.e., the total gains minus the total costs) obtained per purchased item s_j and discounted at an assumed constant rate d at time t. The discount factor d is usually set based on business knowledge. Given these parameters, we can compute the CLV function as follows:

$$CLV_{u_i} = \sum_{j=1}^{l} \mathcal{C}_{s_j}^{u_i}(t+j) \cdot (1+d)^{-j}. \tag{6}$$

The net cash flow $\mathcal{C}_{s_j}^{u_i}(t)$ can be estimated as follows:

$$\mathcal{C}_{s_j}^{u_i}(t) = \varepsilon_j \cdot S_{s_j}^{u_i}(t), \tag{7}$$

where ε_j corresponds to the marginal profit for product s_j. The total value coming from an active customer is obtained from Equation (6) which allows us to calculate CLV for customer u_i at time t for the q products. Equation (6) is more straight-forward, the challenge is estimating the $S_{s_j}^{u_i}(t)$ in a reasonable way. In order to estimate $S_{s_j}^{u_i}(t)$ survival analysis is used.

B. Survival Analysis

Survival analysis is a heavily studied topic in statistics. Survival Analysis estimates the time until an event occurs, also called time to event. It comprises two elements: time effect and individual effect. The first is described as a survival function shared among the whole population. The second describes the difference of individuals from a base point in terms of the covariates. Combining these two functions makes it possible to describe quantitatively the probability of an event taking place at time t [28].

1) Survival Hazard Function: The survival hazard function is used to represent the probability that time of the event of interest occurs later than a specified time t. The survival function is represented by $S(t) = P(T \geq t)$ and is a non-negative monotonically decreasing function with $S(0) = 1$. At the beginning, 100% of the observed subjects survive meaning none of the events of interest have occurred. The Hazard function $h(t)$ represents the likelihood that the event occurs at time t given that no event has occurred before time t. Formally, the Hazard function is defined as $h(t) = \lim_{\Delta t\to 0} \frac{F(t+\Delta t)-F(t)}{\Delta t \cdot S(t)} = \frac{f(t)}{S(t)}$, where $F(t) = 1 - S(t)$ is the cumulative distribution function, which represents the probability that the event of interest occurs earlier than t; and $f(t)$ is the density function $f(t) = F'(t)$. Note that, $f(t) = -S'(t)$. Similar to $S(t)$, the Hazard function $h(t)$ is a non-negative function and can have a variety of shapes. The cumulative Hazard function $H(t)$ is expressed as $H(t) = \int_0^t h(u)du$. Thus, the survival function can be rewritten as $h(t) = -\frac{d}{dt}\left[\ln S(t)\right]$ and $S(t) = exp(-H(t))$ respectively. To estimate the survival hazard the Cox model is widely used due to its efficiency in linear regression analysis of survival data.

2) Neural network for survival analysis: In order to capture non-linear relationships of items and customers we propose a non-linear neural network based survival analysis. The network predicts the covariates of the Hazard rate parameterized by the weights θ. The hidden layers of the network consist of fully connected layers [7]. The output of the network is a single node with linear activation that estimates the log risk function in the Cox model. The network is trained by setting the objective function to the average negative log partial likelihood of $h_\theta(\boldsymbol{x}_{u_i}^p)$ with L_2 regularization:

$$l(\theta) = -\frac{1}{n_e} \sum_{q:e_q=1} \left(h_\theta(\boldsymbol{x}_{u_i}^q) - log \sum_{p\in\mathcal{R}(t_{s_q}^{u_i})} e^{h_\theta(\boldsymbol{x}_{u_i}^p)} \right) + \lambda \cdot ||\theta||_2^2, \tag{8}$$

where n_e is the number of users with an observable event, and the set $\mathcal{R}(t) = \{q : t_{s_q}^{u_i} \geq t\}$ is the set of users still at risk of failure at time t. The parameter λ is the $\mathcal{L}_2$ regularization parameter. In order to find the weights of the network, the gradient descended method is used to minimized Eq. (8).

V. Maximizing CLV with Recommender Systems

In Section III the set $\mathcal{S}_{u_i}^{l+1}$ of recommended items was obtained. In this section we propose a final recommendation for user u_i that maximizes CLV (the best item $s_{l+1}^q \in \mathcal{S}_{u_i}^{l+1}$ that minimizes the risk function). After estimating the hazard functions for the items $s_j \in \mathcal{S}_{u_i}^{l+1}$ based on the neural network, the survival functions $S_{s_j}^{u_i}(t)$ are obtained.

Fig. 1 depicts the stacked neural network architecture of the recommender system. It follows two stages. Stage I is the self-attention recommender system. Stage II is the customer lifetime value using non-linear survival analysis. The output of Stage I is the set $\mathcal{S}_{u_i}^{l+1}$ of the k predicted items, $\mathcal{S}_{u_i}^{l+1}$ is used as an input to Stage II and it calculates the maximum CLV_{u_i} function.

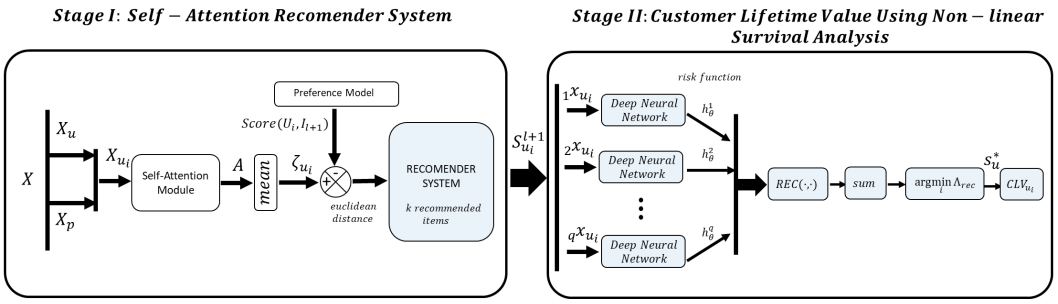

Fig. 1. Proposed stacked neural network. Stage I: Self-attention recommender system. Stage II: Customer lifetime value using non-linear survival.

There are two ways of arriving at the final recommendation. In the first method we estimate a set of CLV functions CLV_{rec} using Eq. (6) for the set $\mathcal{S}_{u_i}^{l+1}$, and find the argmax of CLV_{rec} to obtain the final recommended item $s_{u_i}^*$. The second approach uses an anti symmetric matrix REC to represent the relationship between the different items in the set $\mathcal{S}_{u_i}^{l+1}$ and obtain the final recommendation item $s_{u_i}^*$.

To simplify the notation, we use s_j instead of $s_{l+1}^j \in \mathcal{S}_{u_i}^{l+1}$, which corresponds to one item of the q recommended items at the $l+1$ purchase. For each item $s_j \in \mathcal{S}_{u_i}^{l+1}$ the purchase history matrix $\boldsymbol{X}_{(u_i,s_j)} = [\boldsymbol{X}_{u_i}; \boldsymbol{x}_{(u_i,s_j)}^{l+1}]$ is calculated. Then, $\boldsymbol{x}_{(u_i,s_j)}^{l+1}$ is given as an input to the network. The Hazard function obtained by the network is:

$$h_{u_i}(t; \boldsymbol{X}_{(u_i,s_j)} | s_j \in \mathcal{S}_{u_i}^{l+1}) = \lambda_0 \cdot e^{h_\theta^j(\boldsymbol{X}_{(u_i,s_j)})}, \tag{9}$$

where $h_\theta^j(X_{(u_i,s_j)})$ represents the risk function for user u_i when they purchase the item s_j. The network predicts the risk $h_\theta^j(\boldsymbol{X}_{(u_i,s_j)})$ for each product $s_j \in \mathcal{S}_{u_i}^{l+1}$ that was recommended in Stage I in Section III to user u_i. Thus, based on the assumption that each user has the same baseline Hazard function λ_0, it is possible to take the log of the Hazards ratio in order to compute the risk of each item s_j. Let us define the recommender function $rec_{i,j}^u(x)$ for user u_i as:

$$\begin{aligned} rec_{i,j}^u = \quad & log\left(\frac{h_u(t;\boldsymbol{X}_{(u,s_i)} | s_i \in \mathcal{S}_u^{l+1})}{h_u(t;\boldsymbol{X}_{(u,s_j)} | s_j \in \mathcal{S}_u^{l+1})}\right) = \quad log\left(\frac{\lambda_0 \cdot e^{h_\theta^i(\boldsymbol{X}_{(u,s_i)})}}{\lambda_0 \cdot e^{h_\theta^j(\boldsymbol{X}_{(u,s_j)})}}\right) \\ = \quad & h_\theta^i(\boldsymbol{X}_{(u,s_j)}) - h_\theta^j(\boldsymbol{X}_{(u,s_j)}). \end{aligned} \tag{10}$$

The recommender function can be used to get the relationship between an item $s_i \in \mathcal{S}_{u_i}^{l+1}$ with another item $s_j \in \mathcal{S}_{u_i}^{l+1}$. When the function $rec_{i,j}^u$ is positive, it means that the recommended item s_i leads to a higher risk of death than item s_j, so the item that should be recommended is s_j.

Given the set $\mathcal{S}_u^{l+1}$, the recommender function $rec_{i,j}^u(x)$ for each $s_i, s_j \in \mathcal{S}_{u_i}^{l+1}$ is computed as in Eq. (10). Then, the recommended matrix $REC_u \in \mathcal{R}^{k \times k}$ is constructed as follows:

$$REC_u = \begin{bmatrix} rec_{1,1}^u & rec_{1,2}^u & \dots & rec_{1,k}^u \\ \vdots & \vdots & \vdots & \vdots \\ rec_{k-1,1}^u & rec_{k-1,2}^u & \dots & rec_{k-1,k}^u \\ rec_{k,1}^u & rec_{m,2}^u & \dots & rec_{k,k}^u \end{bmatrix}. \tag{11}$$

The last matrix represents the relationship between the different items in the set $\mathcal{S}_u^{l+1}$. REC_u is an anti-symmetric matrix since it represents the difference of each risk $h_\theta^i({}_i\boldsymbol{X}_u)$ and $h_\theta^j({}_j\boldsymbol{X}_u)$ (for notation and simplicity h_θ^i, h_θ^j). REC_u will be used to obtain the final recommendation $s*_u$ that has the lowest risk. For this aim, we search over all the elements in the matrix. For example, for each element $rec_{i,j}^u$ in the i-th row, if all of the values in that row are negative it means that the item s_i has the lowest risk. However, if the value $rec_{i,j}^u$ is positive, then the s_j gets a lower risk than s_i, meaning the search continues to the j-row and so on. Using a searching algorithm s_u^* could be estimated. However, the time complexity of a searching the matrix is too high.

To estimate s_u^*, we identify the i-th row that contains all the negative values. Then, the sum of each row is computed in $\lambda_{rec_i} = \sum_{j=1}^k rec_{i,j}^u$. Thus, the vector $\Lambda_{rec} = [\lambda_{rec_1}, \dots, \lambda_{rec_k}]^T$ is constructed. Finally, the $arg\ min$ of Λ_{rec} is calculated and the result is s_u^*. This idea is shown formally in the proposition 1.

$$s_u^* = arg\min_i \{\Lambda_{rec}\}. \tag{12}$$

VI. Experimental Results

In this section we investigate the experimental results from our proposed stacked neural network. The dataset used belongs to a large Australian insurance company consisting of small-and-medium enterprises (SME) with transactions starting in 2007 and ending in 2019 that include 18 different items. The dataset had 19,174 customers and 173,127 transactions.

We used our stacked neural network model to obtain the item that maximize CLV. The top 5 recommended items, is obtained in stage I based on the self-attention recommender system. Then, Stage II computes the CLV using non-linear survival analysis and also obtains the recommended item that maximize CLV. The results are provided in Table VI. In the case of $user_{6455}$ the item that maximizes CLV the most is $item_7$, which was option three of the Self-Attention Model.

Fig. 4 and Fig. 2 show the survival function for users $user_{6455}$ and $user_{11426}$ for their current item and our 5 recommended items. Fig. 3 shows the top 10 recommended items based on defection risk. We can observe that the $item_1$ was the most recommended product. Fig. 5 describes the top 10 recommended products that maximize CLV in this case the $item_3$ was recommended the most. Fig. 6 clusters customers based on defection risk per

current product (item). Here $item_1$ concentrates a high number of customers with low risk followed by $item_8$ and $item_5$.

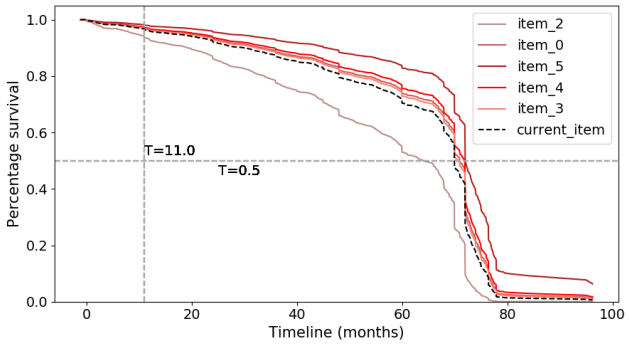

Fig. 2. User 11426's survival curves, the recommended product at the first stage is item_2, the recommended product based on defection risk is item_5 and the final recommended product that maximizes CLV is item_3.

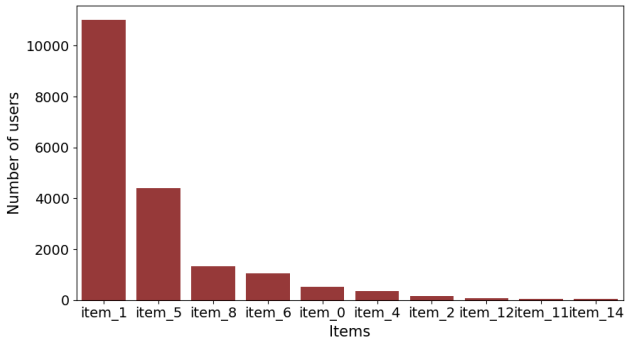

Fig. 3. Top 10 items recommended based on defection risk.

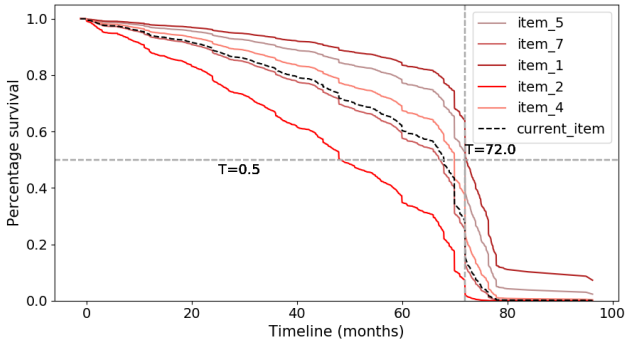

Fig. 4. User 6455's survival curves. The recommended product at the first stage is item 5, the recommended product based on the defection risk is item_1 and the final recommended product that maximizes CLV is item_7.

VII. Conclusions

The system described was tested with data from a real insurance company. It was observed that the survival curves changed when varying products on the customers we simulated. The results show insurers can proactively focus their marketing efforts on more specific items that do increase customer lifetime value and divest on the ones whose cross-sell opportunities do not increase it.

This technique manages to model the non-linear relationships of user's preferences through the neural network survival method,

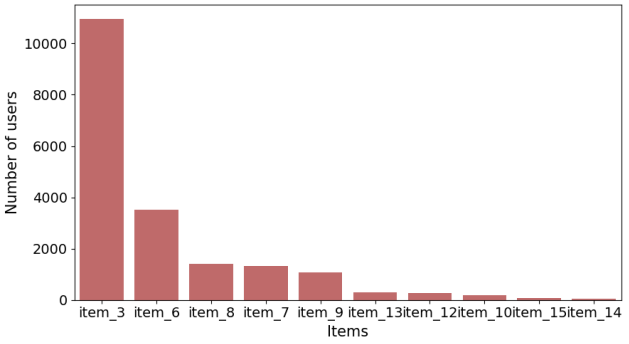

Fig. 5. Top 10 items recommended based on CLV maximization.

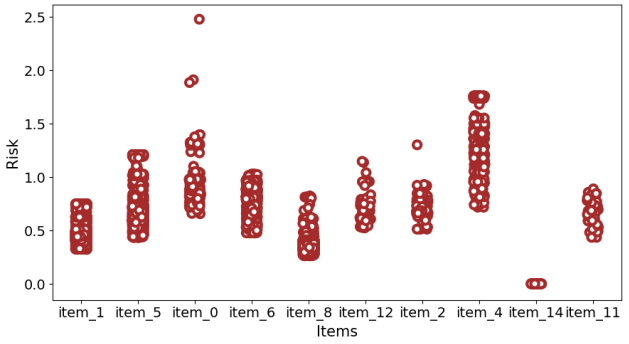

Fig. 6. Customers clustered based on defection risk per current product (item).

providing an edge when compared to other linear survival analysis methods in the literature. Another success was being able to factor in product features which most survival analysis methods do not consider in their time to event predictions.

Appendix

Proposition 1. *Consider the anti-symmetric matrix REC_U that satisfies $REC_u = -REC_u^T$ (in component notation $rec_{i,j}^u = -rec_{j,i}^u$). If all the elements $rec_{i,j}^u$ $(j = 1, \ldots, k)$ in the i-th row of REC_u are negative or zero values, then the following equation holds:*

$$\sum_{j=1}^{k} rec_{i,j}^u < \sum_{j=1}^{k} rec_{l,j}^u \; \forall i \neq l. \tag{13}$$

Proof. Moreover, if the elements in the i-th row are negative (i.e., $rec_{i,j}^u = h_\theta^i - h_\theta^j \leq 0 \;\; \forall j = 1, \ldots, k$) for , then the sum of the elements are also negative (i.e., $\sum_{j=1}^{k} \left(h_\theta^i - h_\theta^j \right) < 0$).

In order to prove that $\sum_{j=1}^{k} rec_{i,j}^u < \sum_{j=1}^{k} rec_{l,j}^u \;\; \forall i \neq l$, let us assume that there exists a row l such that $\sum_{j=1}^{k} \left(h_\theta^i - h_\theta^j \right) < \sum_{j=1}^{k} \left(h_\theta^l - h_\theta^j \right)$. Then the following equations hold:

$$k h_\theta^l - \sum_{j=1 \forall j \neq \{i,l\}}^{k-2} h_\theta^j - h_\theta^i - h_\theta^l \;\; < \;\; \sum_{j=1}^{k} \left(h_\theta^l - h_\theta^j \right) \tag{14}$$

Stacked Neural Networks			Self-Attention Model				
ID	MaxSurv	MaxCLV	top1	top2	top3	top4	top5
$user_{6455}$	$item_1$	$item_7$	$item_5$	$item_7$	$item_1$	$item_2$	$item_4$
$user_{11426}$	$item_5$	$item_3$	$item_2$	$item_0$	$item_5$	$item_4$	$item_3$
$user_2$	$item_1$	$item_6$	$item_5$	$item_4$	$item_6$	$item_0$	$item_1$
$user_{6859}$	$item_6$	$item_3$	$item_6$	$item_7$	$item_0$	$item_3$	$item_4$
$user_{10969}$	$item_4$	$item_3$	$item_0$	$item_7$	$item_6$	$item_4$	$item_3$

TABLE I
COMPARISON OF TWO APPROACHES: PROPOSED STACKED NEURAL NETWORK VS. SELF-ATTENTION MODEL.

$$(k-1)h_\theta^l - \sum_{j=1\forall j\neq\{i,l\}}^{k-2} h_\theta^j - h_\theta^l + \left(h_\theta^l - h_\theta^i\right) < \sum_{j=1}^{k}\left(h_\theta^l - h_\theta^j\right) \tag{15}$$

Next, let us rewrite $\sum_{j=1}^{k}\left(h_\theta^i - h_\theta^j\right)$ as follows: $\sum_{j=1}^{k}\left(h_\theta^l - h_\theta^j\right) = (k-1)\,h_\theta^i - \sum_{j=1\forall j\neq\{i,l\}}^{k-2} h_\theta^j - h_\theta^i + \left(h_\theta^i - h_\theta^l\right)$.

The last equation provides the decomposition of risks h_θ^i and h_θ^l. Thus, solving for $\sum_{j=1\forall j\neq\{i,l\}}^{k-2} h_\theta^j$ we have:

$$\begin{aligned}\sum_{j=1\forall j\neq i,l}^{k-2} h_\theta^j \;=\; & -\sum_{j=1}^{k}\left(h_\theta^l - h_\theta^j\right) + (k-1)\cdot h_\theta^i \\ & -h_\theta^i + \left(h_\theta^i - h_\theta^l\right)\end{aligned} \tag{16}$$

By substituting Eq. (16) in Eq. (14), the following equation is obtained: $(k-1)\cdot h_\theta^l + \sum_{j=1}^{k}\left(h_\theta^l - h_\theta^j\right) - (k-1)\,h_\theta^i + h_\theta^i - \left(h_\theta^i - h_\theta^l\right) - h_\theta^l + \left(h_\theta^l - h_\theta^i\right) < \sum_{j=1}^{k}\left(h_\theta^l - h_\theta^j\right)$.

Therefore, $k\cdot\left(h_\theta^l - h_\theta^i\right) < 0$. Since $h_\theta^i - h_\theta^j \leq 0 \;\; \forall j = 1,\ldots,k$, thus for any $j = l$, the last equation must satisfy that $h_\theta^i \leq h_\theta^l$. Therefore the last equation is a contradiction and $\sum_{j=1}^{k} rec_{i,j}^u < \sum_{j=1}^{k} rec_{l,j}^u \;\forall i \neq l$ is proved.

□

REFERENCES

[1] M. Lewis and N. Slack, *Operations management: critical perspectives on business and management*. Taylor & Francis US, 2003, vol. 1.

[2] N. Glady, B. Baesens, and C. Croux, "A modified pareto/nbd approach for predicting customer lifetime value," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2062–2071, 2009.

[3] B. P. Chamberlain, A. Cardoso, C. H. Liu, R. Pagliari, and M. P. Deisenroth, "Customer lifetime value prediction using embeddings," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 1753–1762.

[4] S. Gupta and D. R. Lehmann, "Customers as assets," *Journal of interactive Marketing*, vol. 17, no. 1, pp. 9–24, 2003.

[5] D. Jain and S. S. Singh, "Customer lifetime value research in marketing: A review and future directions," *Journal of interactive marketing*, vol. 16, no. 2, p. 34, 2002.

[6] E. W. Ngai, L. Xiu, and D. C. Chau, "Application of data mining techniques in customer relationship management: A literature review and classification," *Expert systems with applications*, vol. 36, no. 2, pp. 2592–2602, 2009.

[7] A, "A," *a*, vol. 1, no. 1, p. 1, 2000.

[8] P. S. Fader, B. G. Hardie, and K. L. Lee, ""counting your customers" the easy way: An alternative to the pareto/nbd model," *Marketing science*, vol. 24, no. 2, pp. 275–284, 2005.

[9] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, 2009.

[10] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*. Springer, 2007, pp. 325–341.

[11] R. Burke, "Hybrid recommender systems: Survey and experiments," *User modeling and user-adapted interaction*, vol. 12, no. 4, pp. 331–370, 2002.

[12] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," *arXiv preprint arXiv:1703.03130*, 2017.

[13] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, 2016, pp. 1480–1489.

[14] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," *arXiv preprint arXiv:1606.01933*, 2016.

[15] S. Wang, L. Hu, L. Cao, X. Huang, D. Lian, and W. Liu, "Attention-based transactional context embedding for next-item recommendation," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[16] C. Zhou, J. Bai, J. Song, X. Liu, Z. Zhao, X. Chen, and J. Gao, "Atrank: An attention-based user behavior modeling framework for recommendation," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[17] S. Zhang, Y. Tay, L. Yao, and A. Sun, "Dynamic intention-aware recommendation with self-attention," *arXiv preprint arXiv:1808.06414*, 2018.

[18] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," *arXiv preprint arXiv:1805.08318*, 2018.

[19] X. Chen, Y. Zhang, Q. Ai, H. Xu, J. Yan, and Z. Qin, "Personalized key frame recommendation," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 315–324.

[20] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, "Outer product-based neural collaborative filtering," *arXiv preprint arXiv:1808.03912*, 2018.

[21] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Advances in neural information processing systems*, 2013, pp. 2643–2651.

[22] J. Ansell, T. Harrison, and T. Archibald, "Identifying cross-selling opportunities, using lifestyle segmentation and survival analysis," *Marketing Intelligence & Planning*, vol. 25, no. 4, pp. 394–410, 2007.

[23] G. Theocharous, P. S. Thomas, and M. Ghavamzadeh, "Personalized ad recommendation systems for life-time value optimization with guarantees," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[24] A. Hallak, Y. Mansour, and E. Yom-Tov, "Automatic representation for lifetime value recommender systems," *arXiv preprint arXiv:1702.07125*, 2017.

[25] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 197–206.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[27] N. Usunier, D. Buffoni, and P. Gallinari, "Ranking with ordered weighted pairwise classification," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 1057–1064.

[28] J. P. Klein and M. L. Moeschberger, *Survival analysis: techniques for censored and truncated data*. Springer Science & Business Media, 2006.