ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES

MSc THESIS

İbrahim Erdem KALKAN

A CROSS SELLING RECOMMENDER SYSTEM BASED ON RECURRENT NEURAL NETWORKS FOR ONLINE SHOPPING

DEPARTMENT OF INDUSTRIAL ENGINEERING

ADANA-2022

ABSTRACT

MSc THESIS

A CROSS SELLING RECOMMENDER SYSTEM BASED ON RECURRENT NEURAL NETWORKS FOR ONLINE SHOPPING

İbrahim Erdem KALKAN

ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES DEPARTMENT OF INDUSTRIAL ENGINEERING

Supervisor : Assoc. Prof. Dr. Cenk ŞAHİN
Year: 2022, Pages: 72
Jury : Prof. Dr. Rızvan EROL
: Prof. Dr. Turgay İBRİKÇİ
: Assoc. Prof. Dr. Cenk ŞAHİN

Recommender systems are considered to be capable of prediction of what is the next product to buy for a specific customer. For an effective cross-selling framework it is crucial to identify which customers are more suitable than others to target a product for the retail industry. This thesis proposes a hybrid model which implements recurrent neural network architectures with self-attention mechanism processing implicit feedback. Furthermore, the proposed design is capable of handling both sequential and non-sequential features, which correspond to purchase behavior and non-behavioral customer specific information respectively. This study represents an alternative solution to a well-known business problem: improving cross-selling effectiveness by estimating customers' likelihood for which products or services to buy next time. A recommender system which works on additional data configurations is the core concept of the framework. This study is verified with an online shopping dataset, and it is illustrated that the concatenation of relevant features provides additional information to the model, and it is obtained approximately 13% of improvement in evaluation metrics.

Keywords: Marketing Management, Recurrent Neural Networks, Recommender Systems, Cross-Selling

ÖZ

YÜKSEK LİSANS TEZİ

ÇEVRİMİÇİ ALIŞVERİŞ İÇİN ÖZYİNELEMELİ YAPAY SİNİR AĞLARI TABANLI BİR ÇAPRAZ SATIŞ ÖNERİ SİSTEMİ

İbrahim Erdem KALKAN

ÇUKUROVA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

Danışman : Doç. Dr. Cenk ŞAHİN
Yıl: 2022, Sayfa: 72
Jüri : Prof. Dr. Rızvan EROL
: Prof. Dr. Turgay İBRİKÇİ
: Doç. Dr. Cenk ŞAHİN

Öneri sistemlerinin, belirli bir müşterinin satın almayı planladığı sıradaki ürünü tahmin etmede belli bir potansiyele sahip olduğu kabul edilebilir. Perakende sektöründe, etkili bir çapraz satış çerçevesi için, belli bir ürün hedef olarak seçildiğinde, hangi müşterilerin pazarlama için daha uygun olacağını tahmin etmek hayati bir öneme sahiptir. Bu tez, öz-dikkat mekanizması ve özyinelemeli sinir ağlarından oluşan, örtük geri bildirimi işleyen hibrit bir öneri sistemi mimarisi önermektedir. Bununla beraber, önerilen kurgu, sırasıyla davranışsal ve müşteriye özel bilgileri temsil eden, zincirsel ve zincirsel olmayan veri özelliklerini alabilecek bir kapasiteye sahip olacaktır. Bu çalışma, iyi bilinen bir iş problemi olarak, müşterilerin sırada hangi ürün veya hizmeti satın alacaklarının tahmin etmek suretiyle çapraz satışta verimliliğin arttırılması problemine alternatif bir çözüm önermektedir. Bu çerçevenin çekirdeği, ilave veri özelliklerinin farklı düzenlemelerini içeren bir öneri sistemidir. Çalışma, bir çevrimiçi satış datası kullanılarak doğrulanmış, uygun özelliklerin bir araya getirilmesinin modele ilave bilgi sağlayabileceği gösterilmiş ve ölçüm sonuçlarında %13 gibi bir gelişme elde edilmiştir.

Anahtar Kelimeler: Pazarlama Yönetimi, Özyinelemeli Sinir Ağları, Öneri Sistemleri, Çapraz Satış

EXTENDED ABSTRACT

Cross-selling has been defined as additional sales of related or sometimes unrelated items to a previously purchased item (Kamakura, 2008). The crucial point of cross-selling is extracting and evaluating more eligible customers for a target product. For instance, if an online-retailer wants to increase the sales in a specific product, it should attempt to mine marketable customers within its existing portfolio.

Statistical models such as multi-label classification and RS algorithms can be conducted to identify cross-selling opportunities (Bogaert et al., 2019). Customer segmentation is considered as useful for cross-selling. It partitions customers into groups and products purchased by other customers in the same group can be utilized for cross-selling (Zhang et al., 2021). Both customer segmentation and multi-label classification studies exploit both customer specific data such as age, gender etc. and transaction data. Furthermore, segmentation using only transaction data suffers from data sparsity (Baer & Chakraborty, 2013). With this study, a RS which uses implicit feedback combined with a brief of customer specific features is implemented.

Heuristic-based and model-based techniques are two main tools to implement RS. Whereas, heuristic-based techniques commonly include nearest neighbor approaches with different similarity measures, model-based methods commonly are implemented with data mining techniques such as ANN, Bayesian classifiers, and clustering.

Among all methods, the usage of ANN has gained acceleration recently. ANN based RS have attracted significant attention by overcoming obstacles of conventional models and achieving high recommendation quality (Zhang & Yao, 2019). RNN is one of the complicated neural network architectures with different solutions such as LSTM. RNN is considered as capable of analyzing time series data such as stock prices, or daily temperature. of fixed-sized inputs, generally, it works on sequences of arbitrary lengths. Since it could receive sentences, documents, or audio samples as input, it is known as an extremely useful tool for the tasks like

automatic translation, speech-to-text, or prediction of next phrases in a text for a natural language processing (NLP) system (Géron, 2017, ch.14).

Despite its promising theoretical background, a simple RNN has a crucial drawback in terms of attaining long-term dependencies in practice. This issue is known as vanishing gradient problem, which is an effect that is similar to what is observed with deepening feed-forward networks (Chollet, 2018, ch.6). LSTM (Hochreiter & Schmidhuber, 1997) is one of the distinctive architectures to handle this problem.

Attention mechanism is an important part of the proposed model. It has been proposed in the field of image processing with a purpose of focusing on certain feature information during model training. The self-attention or intra-attention mechanism is regarded as an improvement in attention which focuses on internal links of features and reduces the external data dependency (Li et al., 2020), and computes a representation vector of input sequence (Vaswani et al., 2017). Whereas the general attention mechanism seeks to discover the important parts of the sequence relating to the network output, self-attention seeks to find the important portions of the sequence that relate to each other (Katrompas & Metsis, 2022).

A neural network design including LSTM block with an attention layer can be capable of handling both implicit feedback and customer specific data. Variety of RS applications featured by a RNN take into account previous user-item interactions, such as an ordered list of consumed items (Carta, 2021; Devooght & Bersini, 2017; Donkers et al., 2017; Hidasi & Karatzoglou, 2018; Hidasi et al., 2016; Smirnova & Vasile, 2017; Tan et al., 2016; Wu et al., 2016). This thesis proposes an integrated form of attention mechanism with contextual features from the model of Carta, (2021), using additional contextual features from the model of
Smirnova & Vasile, (2017) and using non-sequential customer specific features. Therefore the scientific contribution of this thesis could be considered twofold: 1) presenting a framework to marketing management literature due to its analytical approach to cross-selling in online-shopping, and 2) presenting an improved version

of a RS using sequence to sequence learning.

The proposed model has been implemented with Python and open source libraries, applied and verified with a publicly available online shopping data set. The experiments have been run at open source cloud servers and recorded promising evaluation metrics. Compared to the model which includes only product id as feature, an approximately 13% improvement has been obtained.

GENİŞLETİLMİŞ ÖZET

Çapraz-satış, daha önce satın alınmış ürünlere ilave olarak uygun bir ürünün, hatta bazen uygun olmayan bir ürünün bile, satılması olarak tanımlanabilmektedir (Kamakura, 2008). Çapraz satışta en kritik nokta, hedeflenen bir ürün için uygun müşterilerin ortaya çıkarılması ve değerlendirilmesidir. Örneğin bir çevrimiçi perakendeci belli bir üründe satışlarını yükseltmek isteyebilir, bunun için ilk olarak, mevcut müşteri portföyü içerisinden pazarlama yapmaya uygun müşteriler araştırılmalıdır.

Çapraz satış fırsatlarını değerlendirmek için çok etiketli sınıflandırma ve öneri sistemleri gibi istatistiksel yöntemler uygulamaya konulabilir (Bogaert et al., 2019). Müşteri bölümlemenin çapraz satış için uygun olduğu dikkate alınabilir. Yöntem, müşterileri gruplara ayırmakta ve her bir grupta satın alınmış ürünleri, aynı grup içerisinde satın alma yapmamış müşterilere pazarlanabilir ürün olarak sunmaktadır (Zhang et al., 2021). Hem müşteri bölümleme hem çok etiketli sınıflandırma çalışmaları, yaş, cinsiyet gibi müşteriye özel veriyi ve işlem verisini kullanmaktadır. İlaveten, yalnızca işlem verisini kullanan müşteri bölümleme çalışmaları data seyrekliği gibi bir problemi yaşamaktadırlar (Baer & Chakraborty, 2013). Bu çalışma ile az miktarda müşteriye özel veriyle harmanlanmış, örtülü geri bildirimi kullanan bir öneri sistemi uygulaması yapılacaktır.

Sezgisel-tabanlı ve model-tabanlı teknikler öneri sistemi uygulaması için iki temel tekniktir. Sezgisel-tabanlı teknikler genel olarak; değişik benzerlik ölçütleri kullanan en yakın komşu yaklaşımlarından oluşurken; model-tabanlı teknikler genelde yapay sinir ağları, Bayesyen sınıflandırıcılar ve kümeleme gibi veri madenciliği yöntemlerini kullanmaktadır.

Tüm yöntemler içerisinde yapay sinir ağları uygulamaları son yıllarda bir ivme kazanmıştır. Yapay sinir ağı tabanlı öneri sistemleri geleneksel modellerde karşılaşılan engellerin üstesinden gelmesi ve öneri kalitesinin yüksek olmasıyla önemli bir dikkat cezbetmiştir (Zhang & Yao, 2019). Özyinelemeli sinir ağı, uzun

kısa-dönem hafıza gibi değişik çözümleriyle karmaşık sinir ağı mimarilerinden birisidir. Hisse senedi fiyatları, günlük sıcaklıklar gibi zaman serileri analizinde kullanılabilme kapasitesine sahip olduğu düşünülmektedir. Sabit büyüklükler yerine değişken uzunluklu veri üzerinde çalışabilmektedir. Cümleleri dökümanları, ses örneklerini girdi olarak kullanabilme özelliği olduğu için, otomatik çeviri yapmak, konuşmaları metne çevirmek ya da doğal dil işleme sistemlerindeki takip eden ifadeleri tahmin etmek gibi işlerde oldukça faydalı bir araç olarak bilinmektedir (Géron, 2017, ch.14).

Umut vadeden bu teorik arka planına rağmen basit özyinelemeli sinir ağlarının uzun dönem bağımlılıkları yakalayabilme anlamında kritik bir eksikliği bulunmaktadır. "Vanishing gradient" olarak bilinen bu olgu, gittikçe derinleşen yapay sinir ağlarında gözlenen duruma oldukça benzerdir (Chollet, 2018, ch.6). Uzun kısa-dönem hafıza (Hochreiter & Schmidhuber, 1997) bu problemin çözümüne yönelik özel bir mimaridir.

Dikkat mekanizması önerilen modelin oldukça önemli bir bölümüdür. Fotoğraf işlemede, model eğitimi esnasında belirli özelliklere odaklanabilme amacıyla önerilmiştir. İçsel bağlantılara veya özelliklere odaklanarak dışsal veri bağımlılığı azaltan, öz-dikkat ya da iç-dikkat mekanizmaları, dikkat hadisesinde bir ilerleme olarak kabul edilebilir (Li et al., 2020). Girdi zincirsel verisini temsil etmek üzere bir vektör meydana getirir (Vaswani et al., 2017). Genel dikkat mekanizması, çıktıdaki zincirin önemli parçalarını ortaya çıkarmaya bakarken, öz-dikkat zincirin birbiriyle ilişkilerinde önemli olabilecek bölümlerini bulmaya çalışır.

Özyinelemeli yapay sinir ağıyla desteklenmiş çeşitli öneri sistemi uygulamaları, kullanılan ürünlerin zamansal listesi gibi önceki müşteri ürün ilişkilerini dikkate almaktadır (Carta, 2021; Devooght & Bersini, 2017; Donkers et al., 2017; Hidasi & Karatzoglou, 2018; Hidasi et al., 2016; Smirnova & Vasile, 2017; Tan et al., 2016; Wu et al., 2016).

Bu tez, bağlamsal özellikler ve dikkat mekanizmasını kullanan Carta, (2021) modeli ile; Smirnova & Vasile, (2017) modelinde yer alan ilave bağlamsal

özelliklerin birleştirilmesi ve zincirsel olmayan müşteriye özel datanın ilave edilmesi şeklinde bir birleştirilmiş model önermektedir. Bu nedenle bu tezin bilimsel katkısı: 1) çevrimiçi ticarette çapraz satış için önerdiği analitik yöntemi nedeniyle pazarlama yönetimi literatürüne yeni bir çerçeve sunmak ve 2) mevcut öneri sistemleri modellerinin geliştirilmiş versiyonlarını sunmak şeklinde iki yönlü olarak düşünülebilir.

Önerilen model Python ve açık kaynaklı kütüphaneleri kullanılarak geliştirilmiş ve genel kullanıma açık bir çevrimiçi alışveriş veri seti ile uygulanmıştır. Deneysel çalışmalar bulut sunucularda yapılmış ve gelişme vaadeden değerlendirme sonuçları kaydetmiştir. Yalnızca ürün anahtarlarını kullanan model ile kıyaslandığında %13 gibi bir gelişme elde edilmiştir.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Assoc. Prof. Dr. Cenk ŞAHİN, who guided me throughout this project, for his supervision, encouragement, patience and immense knowledge and for leading my interests to research and scholarship.

I would also like to show gratitude to my committee, including Prof. Dr. Rızvan EROL and Prof. Dr. Turgay İBRİKÇİ, for their patience, and precious advice.

My sincere thanks go to my wife Sultan and my son Hakan Erdem for their encouragement, patience and support through the research.

TABLE OF CONTENTS **PAGES**

LIST OF TABLES PAGES

LIST OF FIGURES

ABBREVIATIONS

: Number of
ADAM : Adaptive Momentum Estimation
ANN : Artificial Neural Networks
Avg. : Average
CF : Collaborative Filtering
CRM : Customer Relationship Management
CV : Cross-Validation
DL : Deep Learning
Emb. : Embedding
GPU : Graphics Processing Unit
HR : Hit Ratio
LSTM : Long Short-Term Memory
MCCV : Monte Carlo Cross-Validation
MF : Matrix Factorization
NDCG : Normalized Discounted Cumulative Gain
NLP : Natural Language Processing
NPTB : Next-Product-to-Buy
Prob : Probability
RFM : Recency, Frequency, and Monetary
RNN : Recurrent Neural Networks
RS : Recommender System

1. INTRODUCTION

Technological improvement and digitization in business have given rise to rapid transformations in marketing. For instance, bio-metrics, smart card, and e-commerce applications flourished rapidly with interactive decision systems (Akçura & Srinivasan, 2005). As a result of that transformation, developing efficient CRM strategies has become a crucial part of marketing. Decision makers should be capable of evaluating all available data, and developing sustainable data-driven models in order to cope with growing competition. It is common knowledge that, since acquiring new customers is much more expensive, it is better practice to enhance relationships with existing customers (Reinartz & Kumar, 2003; Kamakura et al., 2003). Cross-selling, which is focused on strengthening existing relationships, is defined as one of the CRM tools (Kamakura et al., 2003). One can define the business problem as how sales could be boosted with deepening existing contacts.

Cross-selling has been defined as additional sales of related or sometimes unrelated items to a previously purchased item (Kamakura, 2008). One of the challenges of cross-selling is to know which item to target to which customer. In order to identify optimal cross-selling opportunities, marketing strategists first need to be able to identify whether an existing customer is likely to make a subsequent purchase and which product they will buy (Ansel & Archibald, 2007). Typically a company has several candidate products. Due to the communication, or marketing expenses, and also because it is too time-consuming and ineffective, it is unfeasible to target all of the products to each customer. Furthermore, the company may not want to turn off the customer by flooding him or her with too many offers (Knott et al., 2002). Examining who is likely to purchase within the next months is one of the essential parts of effective resource allocation for marketing management (Martinez et al., 2020). The crucial point of cross-selling is extracting and evaluating more eligible customers for a target product. For instance, if an online-retailer wants to increase the sales in a specific product, it should first attempt to mine marketable

customers within its existing portfolio.

During technological transformation, cross-selling must utilize all the analytical tools to investigate customers' past purchasing behavior, identify customer similarities and potential cross-selling opportunities at each contact with the customer, rather than relying on a sales or services representative to decide whether to cross-sell and which item to offer (Kamakura, 2008). Statistical models such as multi-label classification algorithms can be conducted to identify cross-selling opportunities. By using these algorithms, it can be predicted which products a customer will buy next (Bogaert et al., 2019). Customer demographics, or some other relational data are required to implement a multi-label classification. Customer segmentation is considered as useful for cross-selling. It partitions customers into groups that maximize customers' similarities within a group and their dissimilarities among groups based on customers' attributes used in the segmentation process. Products purchased by other customers in the same group can be utilized for cross-selling (Zhang et al., 2021). One can implement a customer segmentation exploiting both customer specific data such as age, gender, etc. and transaction data. Furthermore, segmentation using only transaction data suffers from data sparsity. For an efficient segmentation it is required to add customer specific data (Baer & Chakraborty, 2013). On the other hand collecting such kinds of detailed data brings about customer reactions. Many customers are concerned that companies may abuse the information they collect. Therefore, any attempt to collect customer specific data without permission could give rise to unwanted results (Akçura & Srinivasan, 2005).

One other useful framework to conduct cross-selling is called recommender system (RS). The task of RS is extracting from past transactions or user preferences data, the information about users' possible future likes and interests (Lü et al., 2012). RS could provide users such personalized recommendations that fit their taste and needs. They exploit different types of user feedback. Most convenient is the high quality explicit feedback, which includes explicit input by users regarding their interest in products (Hu et al., 2008). Rating data is the most convenient example of

explicit feedback. Since systems which use such sort of data demand more user effort, time, and attention, for some industries, especially online retail, explicit feedback is often hard to collect in sufficient amounts. There are additional difficulties with explicit feedback. First, since users usually give high ratings to items they think they should consume, it tends to be highly biased. Second, users tend to rate only a small fraction of the products they purchase (Geuens et al., 2017; Verstrepen et al., 2017). Therefore, explicit feedback does not reflect users' real interests. To cope with aforementioned difficulties, attention is increasingly shifting towards systems which engage with implicit feedback likewise items bought, videos watched, songs listened to, books lent from a library, ads clicked on, etc. Data consisting of positive-only purchase history is a typical form of implicit feedback (Verstrepen et al., 2017).

It is possible to find quite rich literature regarding a variety of techniques that could be employed to implement a recommender system. Adomavicius & Tuzhilin (2005) have put forth a comprehensive framework about techniques. According to them, heuristic-based and model-based techniques are two main titles. Whereas, heuristic-based techniques commonly include nearest neighbor approaches with different similarity measures, model-based methods commonly are implemented with data mining techniques such as artificial neural networks (ANN), Bayesian classifiers, and clustering. Among all methods, the usage of ANN has gained acceleration recently. Deep learning (DL) based (Since DL is a subset of machine learning which is concerned with neural networks, in this thesis ANN and DL are used with the same meaning.) RS have attracted significant attention by overcoming obstacles of conventional models and achieving high recommendation quality (Zhang & Yao, 2019). Furthermore, from the perspective of RNN, applications which are employed with a special RNN architecture, such as long short-term memory (LSTM), applied to a collaborative filtering (CF) problem could yield quite competitive results compared to standard nearest neighbors and matrix factorization (MF) methods (Devooght & Bersini, 2017).

Main purpose in this thesis is presenting a solution to predict which customers are more suitable than others in order to cross-sell a product as a target for the retail industry. In terms of solution, this study proposes an integrated version of existing models which implements recurrent neural network (RNN) architecture processing implicit feedback. The scientific contribution of this thesis could be considered twofold: 1) presenting a framework to marketing management literature due to its analytical approach to cross-selling in online-shopping, and 2) presenting an improved version of a RS using sequence to sequence learning. This study illustrates that the concatenation of additional features such as contextual or item-based, supplies useful information to the models, and improves the evaluation records.

The following section of this study informs about related works of both cross-selling applications using different data mining tools and RS developed with deep learning. The theoretical background of RS with RNN and proposed architecture of model configuration are presented in the third section. With the fourth section, the data-set and conducted experiments are explained thoroughly. The consequences, limitations and future studies are discussed in the fifth section.

2. RELATED STUDIES

The previous and related studies are considered in two sub-categories. First, the literature is examined and presented with respect to cross-selling applications, and then RS applications with their different methods.

2.1. Cross-Selling Applications

From the perspective of cross-selling, one can notice a tendency to use data-driven models to implement cross-selling applications in the early 2000's. Kamakura et al. (2003), is not the first, but quite inspiring, which provides a model and its extended versions which are developed to implement cross-selling based on customer transaction data. They use financial transactions and customer specific data comprising 22 different financial products which belong to 5,550 different customers.

Knott et al., (2002), handle cross-selling as next-product-to-buy (NPTB) model. They propose an NPTB framework to predict which product a particular customer is most likely to buy. According to their presentation, data compilation, model selection, model evaluation and scoring and targeting customers are four major steps for an NPTB framework. Their formulation is simply as follows:

$$A = The\ event\ that\ the\ customer\ purchases\ over\ a\ certain\ period.$$
$$B = The\ event\ that\ the\ customer\ next\ buys\ a\ specific\ product.$$
$$Prob(A\ and\ B) = Prob(A) * Prob(B|A)$$

Where $Prob(B \mid A)$ is the NPTB probability, and $Prob(A)$ is the purchase incidence prediction. If a customer will be targeted, both $Prob(A)$ and $Prob(B|A)$must be obtained reasonably sufficient. They compare four statistical models with a heuristic field test. The models are implemented with banking data for 14 retail products and 270,842 customers. Their results illustrate that such statistical models, regardless of

what sort of model, can help companies focus on customer retention rather than acquisition.

Thuring et al. (2012), implement a method for target customer selection using historical purchase behavior data-set which is supplied by an European insurance company. They analyze customer data and develop a model for predicting a score for each customer with respect to a product. The scores represent customer behavior for a not owned product based on historical and behavioral information about other owned products. A company could select a target group for a marketing campaign based on the predicted scores, and offer a specific product to this group. The company can evaluate feedback of cross-sale campaigns to be analyzed to refine the model. They also illustrate the cross-selling work-flow in the financial sector.

Bogaert et al. (2019), have a different aspect with their evaluation purpose. They use both statistical models and RS approach to compare their effectiveness, and result that state-of-the-art multi-label classification techniques, such as “adaboost” and “random forest” are slightly better than RS approach. Their data-set comprises 96,602 unique customers which possess 83 different financial products.

One can notice that repurchase prediction models are not only integral parts of CRM, but also highly related with cross-selling applications. These kinds of models exploit customer transactions data and derive a large feature set from the previous purchases to predict the next purchase. Martinez et al. (2020) derive 276 predictors to evaluate the state-of-the-art machine learning algorithms and obtain its best with “gradient tree boosting”. Chou et al. (2021), propose a new approach which overwhelms a RNN architecture's results with approximately 100 derived predictors. One can also find predictive models which are associated with DL. Salehinejad & Rahnamayan (2016), propose RNN as an efficient technique to predict recency, frequency, and monetary (RFM) [1] variables of given customers.

[1] Recency: the number of days since the last purchase date. Frequency: the number of all purchases. Monetary: the total amount of money spent on all purchases.

Handling similarities in terms of both users and items is one other point of view to identify cross-selling opportunities. Moon & Russell (2012), propose a statistically-based CF model to obtain purchase probability of customers, based on the customer preference similarity stemming from prior purchase behavior. They clarify their model by using a sample of 4,000 customers from an insurance company's database. The sample comprises both product ownership history and postal codes data to identify the similarities of customers by their spatial information. As a result of their model comparison, they obtain individuals who are near one another share similar product preferences.

Regarding similarity-based techniques, Zhang et al. (2021), implement a cross-selling model with measuring customer similarity employing a community detection model. They break customers into significant clusters, and regard products purchased in a cluster as targeting products for a customer in the same cluster, if it is not purchased by the customer. Ansel & Archibald (2007), establish clusters of customers in terms of their specifications such as age, gender to identify their repurchase probability.

2.1. RS Applications

On the extent of RS, MF is one the most convenient methods (Tan et al., 2016). MF, which is a popular technique derived from latent factor analysis, mainly relies on low dimensional decomposition of a sparse matrix consisting of the item preferences of the users (Li et al., 2015). One can find a comprehensive literature about different versions of MF (Koren et al., 2009), such as non-negative (Lee & Seung, 2000), probabilistic MF (Salakhutdinov & Mnih, 2007), and its Bayesian versions (Salakhutdinov & Mnih, 2008). He et al. (2017), define a DL inference of MF with embedding layers and propose a multi-layer perceptron approach of RS.

On the aspect of DL-based RS, Zhang & Yao (2019), represent a comprehensive research. The usage of "Restricted Boltzmann Machines" (Salakhutdinov et al., 2007) could be regarded as the most inspiring study. From

multi-layer perceptrons, to more sophisticated building blocks, a variety of neural networks could be implemented to achieve an efficient RS. As a DL architecture, most of the RNN implementations emphasize time awareness with the user-item interactions. Therefore, user interactions such as ratings, transactions, turns into a time-ordered-sequence. RNN applications usually take into account previous user-item interactions, such as an ordered list of consumed items, aiming to predict the next item a target user will likely consume (Donkers et al., 2017).

In terms of handling sequential and session-based data, RNN could be regarded as an excellent tool. Since, a session is regarded as interactions during a certain time interval between user and system, session-based RS should be capable of handling the interactions in real-time. Hidasi et al. (2016), apply the RNN structure to the RS and they criticize the MF shortcomings on session-based data. Without user profile tracking, they use previous item sequences in sessions to future prediction. Their study argues that using one-hot encoding on item sequences always gains better results than using embedding vectors. They propose some modifications on standard RNN, for instance, they use a ranking loss function in their model. Furthermore, a session-parallel mini batch approach is utilized in order to alleviate some problems stemming from session-based data. They conduct their experiments on two large e-commerce data-sets and obtain their approach is successful on common baselines.

Tan et al. (2016), propose a data augmentation method with their session-based RS application. They use embedding drop-out which is dropping some clicks randomly from the session. Therefore, they managed to avoid some noise clicks, and reduce the overfitting. Pre-trained models are utilized for adaptation of the temporal changes on items. They use a model which is first trained on whole data to initialize another model. On the other hand, they also try to alleviate another drawback which is about the output sizes. The study states that when output sizes get larger, a hierarchical softmax function does not work well on the output. They use output embeddings as the next item instead. Their results are reported with the same data-

set and same loss functions with Hidasi et al. (2016) study. As a result they obtain deployable forms RS with RNN.

Wu et al., (2016), represents an industrial framework which employs a RS that could refine recommendations in real time. Their system runs at an e-commerce website, and exploits page browsing history. Since web-page browsing is a dynamic data source, an online updated RS is appropriate. The sequences of pages are handled in their system with a sliding window approach. They test their system with a data-set that has 232,326 records, 27,985 sessions, and 37,667 unique users who bought 1,584 different items, and outperforms a CF system.

Hidasi & Karatzoglou (2018), demonstrate their new design of ranking loss functions and new negative output sampling system. Mini batches are designed in their new approach as a manner of introducing a bit of negative examples to the model in each iteration. On the other hand, they argue that categorical cross-entropy loss and softmax operation on it shows an instability. To target that drawback they improve previously demonstrated (Hidasi et al., 2016) loss functions with a “max” operation with a brief computational cost.

Devooght & Bersini (2017), illustrate a competitive version of CF powered by LSTM which outperforms nearest neighbor and MF approaches. They break the prediction into two as 1) short-term prediction which concerns only the next item to be consumed, and 2) long-term prediction of the item which is consumed once eventually. On the other hand, they also focus on coverages which mean the ability of the model of distinct item recommendation. They use popular data-sets which have explicit feedback, and a brief of user and item specific features. They also test the effects of the combination of additional features with a one-hot encoding form. They obtain LSTM especially good in terms of short-term prediction and item coverage.

Smirnova & Vasile (2017), offer different combinations of contextual information as sequential form with item sequences, and obtain more effective models. They exploit two different transaction data-sets consisting of session-based

clicks. They draw attention beyond item history: “Besides the order of items, additional information about user-item interaction is frequently available, such as type of the interaction, the time gaps between events and time of day of interaction”. It is possible to see practical benefits of appropriate contextual information with another application in Carta, (2021). A self-attention mechanism is seen as an integral part of the studied model. Furthermore, the model avoids softmax transformations of outputs with a categorical cross-entropy loss.

3. THEORY & METHOD

In this section the theoretical background of the proposed model is explained, and its components are demonstrated. However, a mathematical definition of the problem, which is handled in section 3.1, needs to be presented before. Section 3.2. explains the proposed models components each of whose related theoretical requirements are briefly demonstrated.

3.1. Problem Definition

As highlighted in previous studies (Bogaert et al., 2019), RS could be regarded as a useful tool in an effective marketing framework. According to the simple illustration of Figure 3.1, since RS outputs a prediction frame of items that are recommended for each customer, either previously purchased or not, it could be regarded as both target item set for selected customers or target customer set for a certain item.

This thesis represents a solution to a well-known business problem: improving cross-selling effectiveness by estimating customers' likelihood for which products or services to buy next time. The proposed framework exploits a basic transaction data set like an abstract presentation at Table 3.1, and mathematical representation of the recommendation problem is mostly adapted from the study of Martinez et al. (2020). Every shopping of each customer has a record as a row with a timestamp in the transaction data set.

Suppose that an online retailer's customer portfolio has a total number of U customers represented by their IDs (See Table 3.1, Customer ID). The customer ($u \in \{1, \dots, U\}$) has a purchase history recorded within a certain period, $i_{u,Nu}$ where N_u represents transactions for a customer u.

Table 3.1. Simple form of transaction data

Customer ID	Transaction ID	Timestamp	Item
u	n	$t_{u,n}$	$i_{u,n}$
1	*1*	$t_{1,1}$	$i_{1,1}$
…	…	…	…
1	N_1	t_{1,N_1}	i_{1,N_1}
2	*1*	$t_{2,1}$	$i_{2,1}$
…	…	…	…
U	1	$t_{U,1}$	$i_{U,1}$
…	…	…	…
U	N_u	t_{U,N_u}	i_{U,N_u}

Each purchase has a purchase time, t_u, and an item ID, i. All transactions are assumed ordered chronologically; $t_{u,i1} \leq t_{u,i2}\ for\ all\ i_1 \leq i_2$. Given an historical transaction data, $\{(t_{u,n}, i_{u,n}) | n = 1, \ldots, N_u\}$ for every customer $u \in \{1, \ldots, U\}$; the problem is defined as the prediction of the item ID purchased in the next transaction, $N_u + 1$ chosen from the item catalog, $i \in \{i_{U,Nu}\}$. The problem could be constrained with time frames. That is, given a purchase history for a user, between the timestamp, t_a and t_b, the prediction of item ID which will be purchased by that user within a certain time frame after t_b.

All kinds of data elements mentioned so far could be handled as sequential due to its chronologically ordered form. However, since data containing user features do not have any time attributes, it is considered as static features. In addition to Table 3.1, different data sets consisting of customer features and item features could be possible as side information sources.

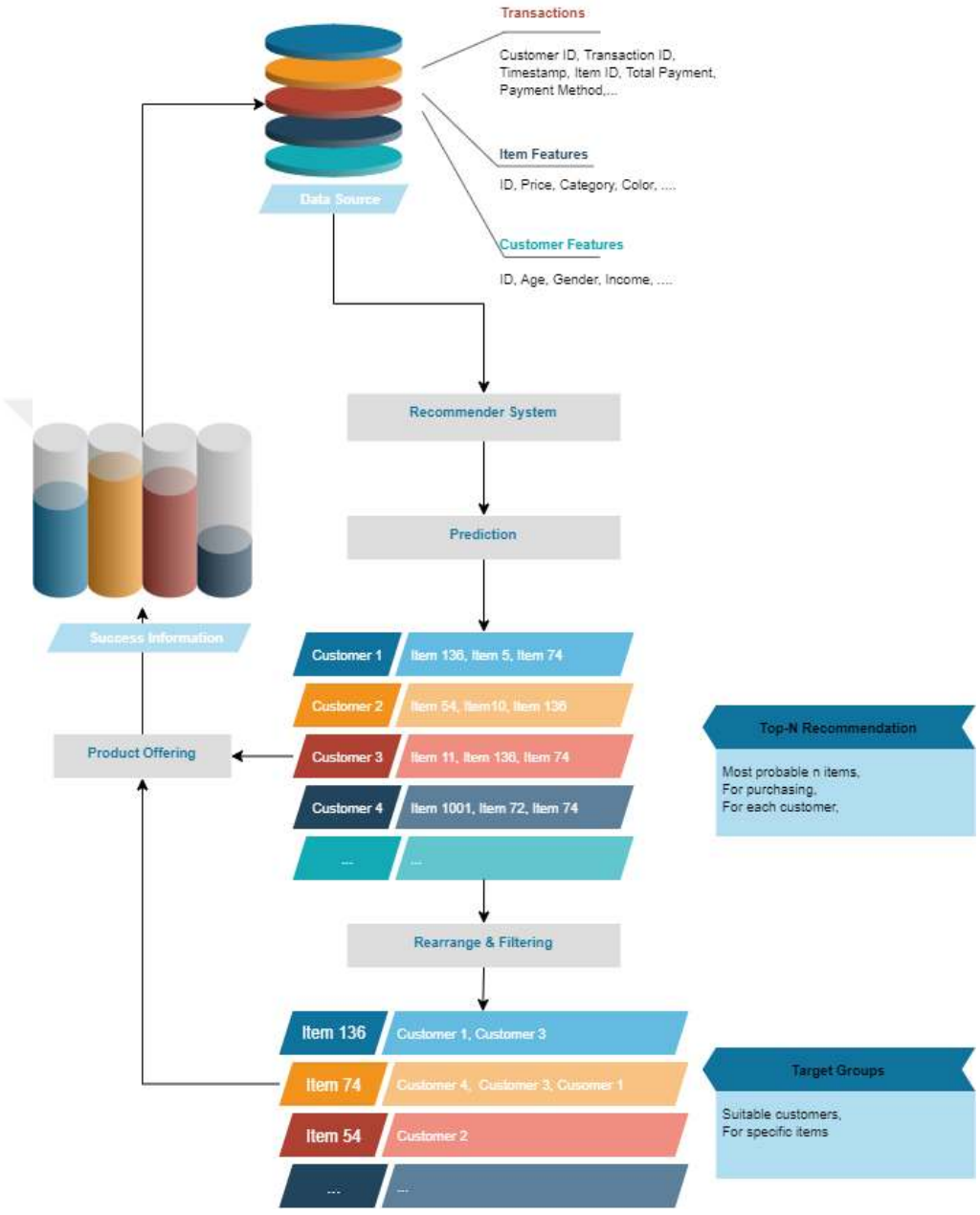

Figure 3.1. Cross Selling Framework

3.2. Proposed RS Configuration

A RS which employs a RNN with different side information configurations is proposed as the core concept of this study. In this subsection, the proposed model is introduced. Briefly, the proposed model is an integrated form of attention mechanism with contextual features from the model of Carta, (2021), using additional contextual features from the model of Smirnova & Vasile, (2017) and using non-sequential customer specific features. It is built up of RNN (LSTM) and fully connected ANN layers with a self-attention layer, and additional features are integrated in order to obtain more satisfactory results. Figure 3.2. illustrates the full connections of the components of the model.

3.2.1. Input Setup

Item-ids of Figure 3.2. refer to customer's purchase history, and other item features are prepared separately as additional features like contextual features and customer features described below.

As mentioned earlier, RSs which exploit implicit feedback are the main point of interest of this thesis. Therefore, transactions of an online retailer and data relevant to a particular purchase, are sufficient for modeling tasks. It is used in this study such an approach which is similar to Devooght & Bersini's (2017), explanation in terms of considering each item as a word, the catalog of items as the full vocabulary, and the purchase history of each user as input sequences.

The possible contributions of contextual information are clearly explained by Smirnova & Vasile (2017). It also proposes concatenation of additional contextual features to the main sequence of purchase history. If available, extracting more contextual features such as amount of payment, or payment method and adding them to previously proposed, especially time-related features, could record a significant improvement.

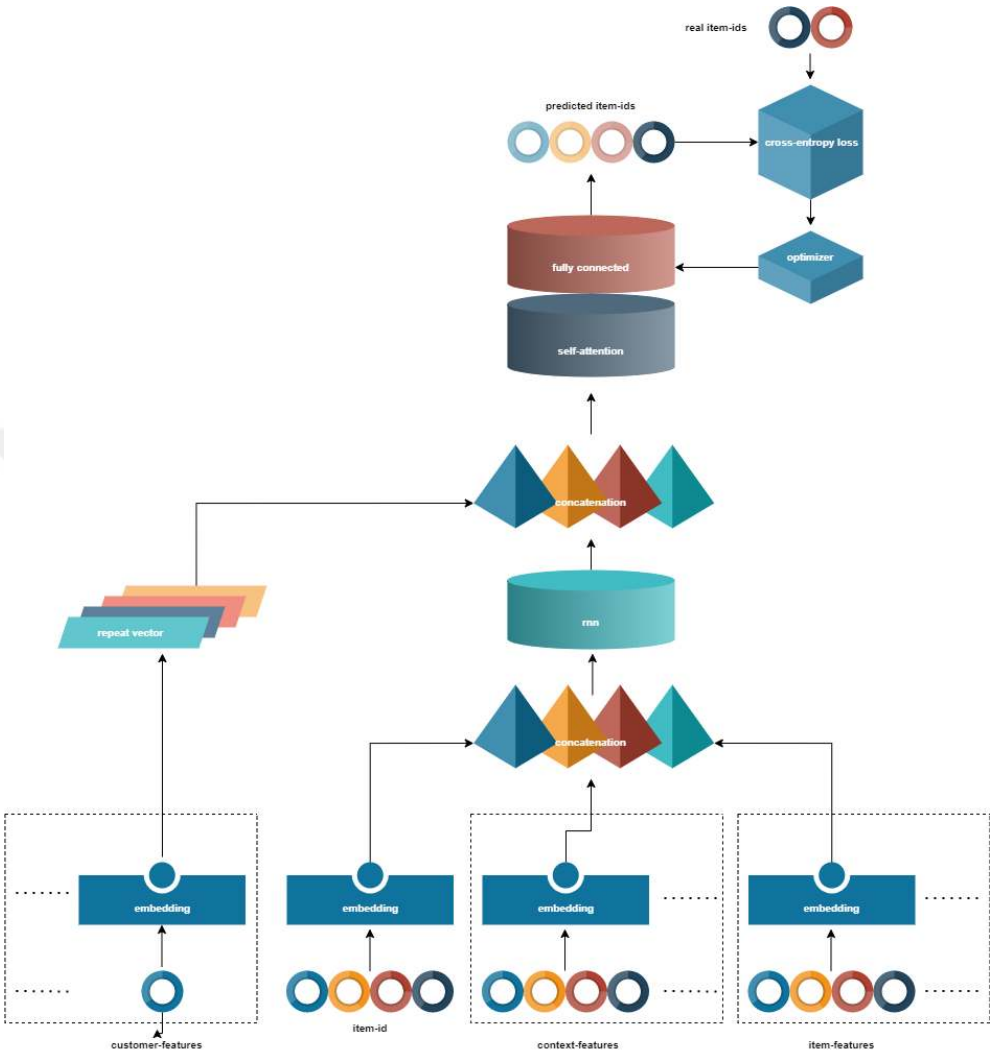

Figure 3.2. Model Schema

Even though it is sometimes difficult to collect customer specific variables (Akçura & Srinivasan, 2005), if available, customer features could be capable of building more personalized recommendations. As mentioned previously, such kinds of features are not changeable in short time ranges, therefore they are not handled as timely ordered. Customer features are included in the model by repeating them as a

number of time-steps. This issue is represented in the Figure 3.2. by repeat vector[2]. On the other hand, in most cases storing item features such as category, price, are easier. For this reason, in this study it is also proposed using eligible item features additionally.

Every kind of feature is transformed into sequences associated with a customer's transactions and used as input vectors. Besides, most of the models handle inputs in the form of the one-hot encoded vectors (Devooght & Bersini, 2017; Tan et al., 2016; Hidasi & Karatzoglou, 2018; Donkers et al., 2017). Instead of one-hot encoded inputs, embedding layers are used aiming to get 3 dimensional representations of inputs regarding not only purchase history but also additional features.

3.2.2. RNN for RS

A simple RNN is a type of neural network which has an internal loop (Chollet, 2018, ch.6). It is capable of analyzing time series data such as stock prices, or daily temperature. Instead of fixed-sized inputs, generally, it works on sequences of arbitrary lengths. Since it could receive sentences, documents, or audio samples as input, it is known as an extremely useful tool for the tasks like automatic translation, speech-to-text, or prediction of next phrases in a text for a natural language processing (NLP) system (Géron, 2017, ch.14). The main characteristic of a simple RNN is the existence of an internal hidden state (h) in recurrent units, which is updated with the following function:

$$h_{(t)} = g(Wx_{(t)} + Uh_{(t-1)}) \quad (1)$$

Where g represents an activation function, $x_{(t)}$ is the input at time step t, $h_{(t-1)}$ is the previous state, and $h_{(t)}$ is the current state which is a function of current

[2] See: https://keras.io/api/layers/reshaping_layers/repeat_vector/

input and previous state. W and U are weight matrices for input and hidden state respectively.

Despite its promising theoretical background, a simple RNN has a crucial drawback in terms of attaining long-term dependencies in practice. This issue is known as vanishing gradient problem, which is an effect that is similar to what is observed with deepening feed-forward networks (Chollet, 2018, ch.6). LSTM (Hochreiter & Schmidhuber, 1997) is one of the distinctive architectures to handle this problem (See: Figure 3.3 and 3.4).

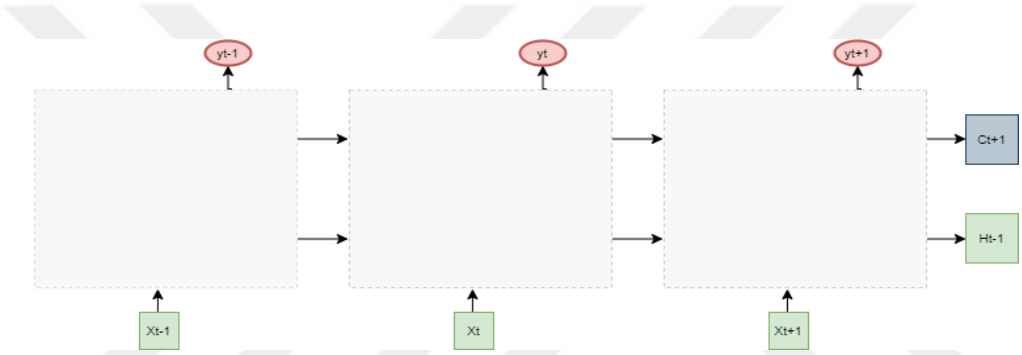

Figure 3.3. Sequence View of LSTM Cells

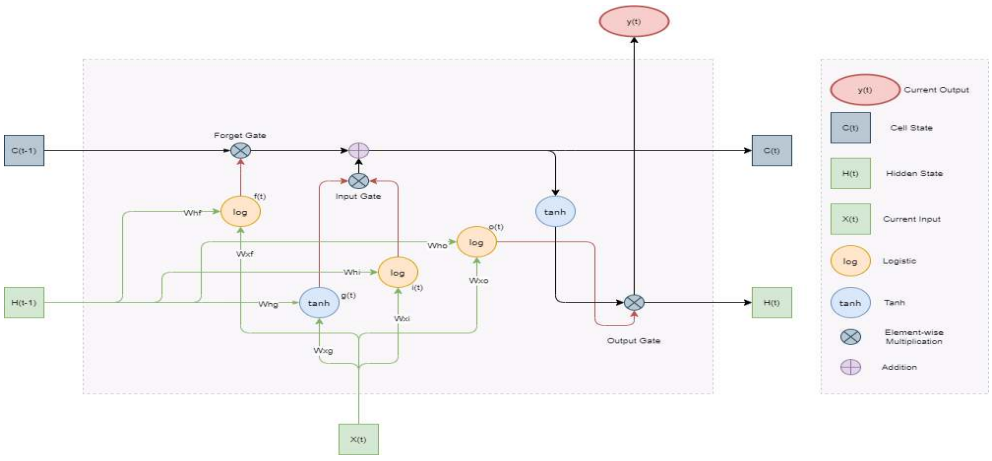

Figure 3.4. Detailed View of an LSTM Cell

LSTM includes 4 layers, $f_{(t)}, g_{(t)}, i_{(t)}, o_{(t)}$ each of which is designed for different tasks. As it is seen in Figure 3.4. long-term memory cell, $C_{(t-1)}$ first goes through a forget gate, drops some memories, and adds some new memories with the addition operation. New memories are selected with different input and hidden state operations. Therefore, at each time-step, some memories are dropped and some memories are added (Géron, 2017, ch.14). Calculations for those three layers:

$$f_{(t)} = \sigma({W_{xf}}^T . x_{(t)} + {W_{hf}}^T . h_{(t-1)} + b_f \ \ ) \tag{2}$$

$$g_{(t)} = tanh({W_{xg}}^T . x_{(t)} + {W_{hg}}^T . h_{(t-1)} + b_g) \tag{3}$$

$$i_{(t)} = \sigma({W_{xi}}^T . x_{(t)} + {W_{hi}}^T . h_{(t-1)} + b_i) \tag{4}$$

After the addition operation, the long-term state is copied and passed through the $tanh$ function, and then the result is filtered by the output gate. By this way the cell generates the hidden state, $h_{(t)}$ and current time-step's output y_t which are equal to each other (Géron, 2017, ch.14). Calculations for current output and memory states:

$$o_{(t)} = \sigma({W_{xo}}^T . x_{(t)} + {W_{ho}}^T . h_{(t-1)} + b_o) \tag{5}$$

$$C_{(t)} = (f_{(t)} \otimes C_{(t-1)} + i_{(t)} \otimes g_{(t)}) \tag{6}$$

$$y_{(t)} = h_{(t)} = (o_{(t)} \otimes \ tanh(C_{(t)})) \tag{7}$$

Where $W_{xf}, W_{xg}, W_{xi}, W_{xo}$ are the weight matrices for the connection between input and 4 layers. $W_{hf}, W_{hg}, W_{hi}, W_{ho}$ are the weight matrices for the connection between the previous hidden state $h_{(t-1)}$ and 4 layers. b_f, b_g, b_i, b_o are the bias terms for each of the four layers. Compared to the simple RNN, LSTM uses additive memory updates and separates the memory, C_t from the hidden state, h_t which interacts with the environment when making predictions (Cheng et al., 2016).

For targeting RS, if items consumed in time-steps are regarded as inputs, the output of RNN will be a probability distribution over the next item as the next element of the input sequence (Hidasi et al., 2016). It is a ranking over all the items as the next item which could be purchased. For a particular user, first-k items which have a top score from output are recommended.

3.2.3. Self-Attention

Attention mechanism has been proposed in the field of image processing with a purpose of focusing on certain feature information during model training. The self-attention or intra-attention mechanism is regarded as an improvement in attention which focuses on internal links of features and reduces the external data dependency (Li et al., 2020), and computes a representation vector of input sequence (Vaswani et al., 2017). Whereas the general attention mechanism seeks to discover the important parts of the sequence relating to the network output, self-attention seeks to find the important portions of the sequence that relate to each other (Katrompas & Metsis, 2022).

Self-attention mechanism has been used in Cheng et al., (2016), a NLP study. Their two sequence model works with an encoder-decoder design, and thanks to an attention layer, computes the relation between current word and previous words in a sequence. Figure 3.5, adapted from their study, illustrates the intra-attention mechanism. Let $C_{t-1} = (c_1 \ldots, c_{t-1})$ denotes the current memory tape, and $H_{t-1} = (h_1, \ldots, h_{t-1})$ the previous hidden tape. At time step t, the relation between x_t and

$x_1 \dots x_{t-1}$:

$$a_{i,t} = v^T tanh(W_h h_i + W_x x_t + W_{\hat{h}} \hat{h}_{t-1}) \quad (8)$$

$$s_{i,t} = softmax(a_{i,t}) \quad (9)$$

$$\left[\hat{h}_t \ \hat{c}_t\right]^T = \sum_{i=1}^{t-1} s_{i,t} . [h_i \ c_i]^T \quad (10)$$

Where, $v, W_h, W_x, W_{\hat{h}}$ are new weight matrices. Equations 2-7 are recalculated using the terms, $\hat{h}_t, \hat{c}_t$.

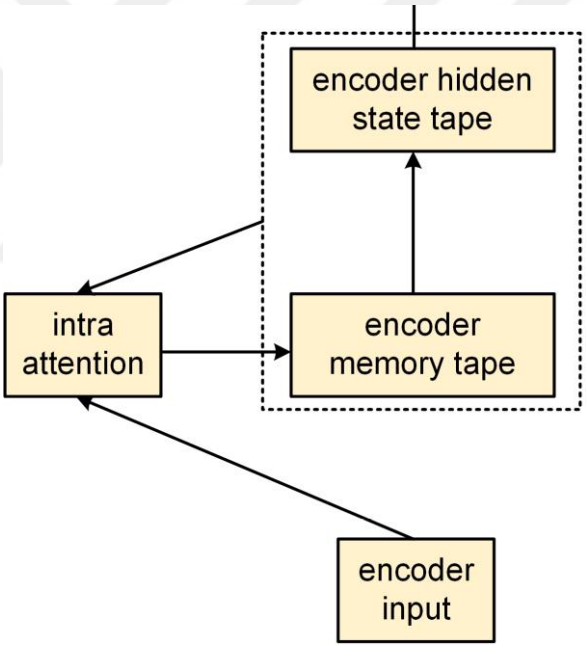

Figure 3.5. Intra-Attention from an Encoder-Decoder Structure (Cheng et al., 2016)

Even though self-attention is a convenient tool in natural language and image processing it could be easily integrated into such a kind of purchase behavior handling task. Regarding each item is a word as if it is in a NLP model, self-attention

could be useful with calculating the intra-relationship weights of items if they exist, and the model could be capable of weighting inputs before making a classification. Therefore, intra-dependencies of inputs could be handled by paying them appropriate attention.

3.2.4. Fully Connected Layers

This part of the model contains fully connected neural network layers (See: Figure 3.3.). The relationship between inputs and outputs of the model appears clearly in this section. The form of training sequence is one time-step sliding of the input sequence of items. The proposed input and output configuration can be formulated as a sequence-based prediction problem that for any given purchase sequence for a customer: $(t_{u,n}, i_{u,n}) \mid n = 1, \dots N_u - 1$ (See: Section 3.1.), the output could be obtained as: $(t_{u,n}, i_{u,n}) \mid n = 2, \dots, N_u$. The last layer produces the predictions with a form of time-steps and catalog size of items, and the last time-step of the output sequence contains predictions over all items.

In some aspects such model's outputs are regarded as classification-based outputs (Tan et al., 2016), as another sequence. The aim of such a kind of two sequence modeling is to capture intra-dependencies between items (See: Section 3.2.3). Therefore, a categorical cross entropy loss to be minimized is appropriate for such kind of classification task:

$$L = -\sum_{t=1}^{T} \sum_{m=1}^{M} y_m \log p(m) \tag{11}$$

where T is the number of time-steps, M is the number of items, y_m is the true item and $p(m)$ is the probability distribution over items.

4. EXPERIMENTS

For verification purposes of RS, the proposed configuration is implemented and verified thanks to open source Python libraries. Keras[3] and Tensorflow[4] are used for DL implementation. All notebooks and experiment results are available publicly[5]. For all GPU-based experiments, Kaggle[6] platform, with 1 free *Tesla P100-PCIE-16GB GPU* is used. After data collection step, the workflow to be stick to is represented in the Figure 4.1

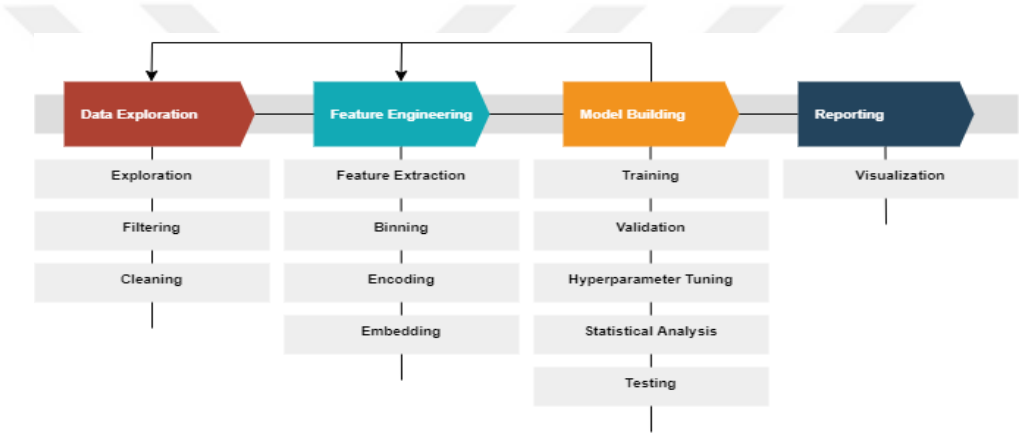

Figure 4.1. Workflow for Experiments

4.1. Data Exploration

To run experiments, a data-set containing purchase history, available variables for extracting contextual, item, and customer features is required (In this part the terms item and product are used interchangeably). An online-shopping transactions data-set which is publicly available[7] collected by different vendors is used. The data-set contains records of about half a million orders with their date-

[3] See: https://keras.io
[4] See: https://tensorflow.org
[5] See: https://github.com/ibrahimerdem/thesis_repo
[6] See: https://kaggle.com
[7] Available at: https://kaggle.com/zusmani/pakistans-largest-ecommerce-dataset

stamps, payment methods, related products with their categories, prices and ordered quantities, and some other attributes from July-2016 to August-2018. For each customer transaction there is one record, and each customer occurs in as many observations as she purchases products (In this part the terms user and customer are used interchangeably). In terms of only completed orders, it has 233,685 observations.

First operation on the data-set is filtering completed transactions, and filtering relevant and available columns some of which are candidate features for modeling. All renamed columns to be processed whose item names and categories are encoded with an integer number, and their brief descriptions are presented in Table 4.1. (Appendix A, Table A.1. represents the first 25 records of the data-set) After the first filtering the data-set contains the information about only purchase events. As a result of analysis of the data-set, no missing values are obtained.

Table 4.1. Filtering available and relevant columns of the data-set.

Column Name	Description
t_id	Unique number for each transaction
t_date	The date of transaction day
c_id	Unique number for each customer
c_since	The date of registration day of a customer
item_name	The definitive name of products purchased
item_category	Categories of each product
price	Price value of each product at their transaction day
amount	The amount of each transaction

As summarized in Table 4.2 and 4.3, it is obtained that most of the customers in the data-set have only 1 purchase. On the other hand some products are observed rare. To overcome this kind of sparsity, two filters are applied. First, 2 and less times

purchased products are eliminated, and then customers who have 2 or more times purchase are selected. This filtering guarantees that the data-set contains customers who purchase more than 1. On the other hand, there could still exist rare items after the second filtering in the data-set. (For distributions of customers and products see: Appendix B. Figure B.1 and B.2.)

Table 4.2. Summary statistics for distribution of purchase numbers of customers.

mean	std.	min	25%	50%	75%	max
3.48	16.82	1.0	1.0	1.0	3.0	1657.0

Table 4.3. Summary statistics for distribution of purchasing numbers of products

mean	std.	min	25%	50%	75%	max
5.2	26.95	1.0	1.0	1.0	3.0	2863.0

There are some product categories which occurred very few times in the data-set out of 16 distinct product categories. Table 4.4 represents the summary statistics for category frequency. To improve data quality, additional data processing steps are applied, and a possible solution which records better metrics is experimented.

Table 4.4. Summary statistics for distribution of number of product categories.

mean	std.	min	25%	50%	75%	max
14605.2	12695.5	910.0	5108.0	10240.0	20715.2	41658.0

First, 3 product categories, which occur at a low frequency, are detected, and a frequency number which is 2,500 for item categories is tried, a new category named "others" is created, the elements whose occurrences are under frequency number are grouped in the new category. (For category frequencies after processing steps see:

Appendix B. Figure B.6.) The data composition after all processing steps is presented in Table 4.5.

Table 4.5. Data composition after processing steps.

#Transactions	150,075
#Customers	25,562
#Products	12,861
#Categories	13

4.2. Feature Engineering

Attributes which could be taken as sequential are prepared as contextual and item features as additional features. Derived contextual features are explained below:

- *recency*: the number of days between the purchasing date and the modeling date for each purchased item. Its values are categorized between 1-100. Appendix B. Figure B.4. represents the distribution after the categorization step of it
- *month*: the month number of each purchased item, as an integer between 1 to 12.
- *dayofweek*: the weekly-basis day number of each purchased item, as an integer between 1 to 7.
- *payment*: the total amount of money paid for each transaction. It is derived by multiplication of price and amount. Its values are categorized between 1-1000. Appendix B. Figure B.5. represents the distribution after the categorization step of it.

Some product features are transformed and used, apart from product ids:

- *category*: main product category encoded with integers between 1-13. Appendix B. Figure B.6. represents the distribution after the categorization step of it.
- *price*: price attribute of each item. Its values are categorized between 1-500. Appendix B. Figure B.3. represents the distribution after the categorization step of it.

On the other hand, integrating static data-features are also exploited to capture user profiling opportunities. For this purpose, the sign up date attribute is used to make a feature which includes the day number between modeling date and sign up date as *customer_since*.

For the purpose of training, first, all discrete features are encoded, and applied discretization for continuous features such as *payment* and *recency*. All customer purchase events are put into a time order. Using product ids and their corresponding additional features, sequences are generated with encoded features. Since all customers do not make equal amount of transactions, a pre-padding method for sequence generation and a masking[8] are applied for training episodes.

Since the main problem relies on prediction of the next item to be consumed (See: Section 3.2.4), one time-step sliding input sequences are used as output sequences similar to text generation methods[9].

For the model training phase, the last purchase event of each customer is held as true labels, and previous events are kept as inputs. All sequences belonging to the purchase events except the last one are fed into the models as inputs and one-time step sliding forms of those inputs' only item ids are considered as outputs. Therefore, the last last item id of the output is calculated as the relevant item.

[8] See: https://tensorflow.org/guide/keras/masking_and_padding
[9] See: https://tensorflow.org/text/tutorials/text_generation

4.3. Model Evaluation

Models are evaluated based on two separate scenarios. First one is the Monte Carlo Cross-Validation (MCCV) technique, and second is performance measuring on a test data-set held out before the training episode. Approximately %10 of the data set is set aside as the testing purpose, and the rest for modeling. There are two kinds of model sets that are used for evaluation: 1) baselines and 2) designed models.

4.3.1. Baselines

Competitor models are compared with 4 baselines:

- *Popularity*: A system which uses most frequent *k* products from the training data as a recommended list.
- *Last-random:* This baseline chooses at most *k* products from each customer's last purchased products. If less than *k* products are available it chooses all products with random order.
- *Last-ordered:* This baseline chooses at most *k* products with an order reverse to purchasing order. If less than *k* products are available it chooses all products.
- *User-user:* A collaborative filtering model with a simple implementation based on user similarities. This baseline implements a sparse matrix called utility or user-item matrix whose cells are filled with 0 and 1. Based on those values, user similarity scores are calculated between each of them. Top *k* items are selected to recommend to a user, from the top *n* similar neighbors' purchases.

4.3.2. Experimented Models

4 different data configuration with same LSTM architecture are designed to evaluate:

- *Product-only (Model 1):* Model which takes only sequences consisting of embedding form of product ids as input.
- *Model 1 + Contextual Features (Model 2):* Model which receives concatenation of inputs of *Model 1* and one-hot-encoded *recency*, *month*, *dayofweek* and *payment* vectors as input (See: Section 4.2).
- *Model 2 + Other Item Features (Model 3):* Model which receives concatenation of inputs of *Model 2* and one hot encoded *price*, and *category* vectors as input.
- *Model 3 + Profile Features (Model 4):* Model which adds a repeat vector of one-hot-encoded *customer-since* values to the output of *Model 3's* LSTM outputs.

4.3.3. Metrics

To assess the performance of experimented models, *hit-ratio (HR)* is conducted as the main evaluation metric. According to (He et al., 2015), if a product from the test set occurs in the top-k recommended list, it is counted as a hit. Thus, *HR(k)* is as calculated as:

$$HR(k) = Number\ of\ hits\ /\ Ground-truth\ test\ set \tag{12}$$

One other popular metric, which assigns higher importance to results at top ranks, scoring successively lower ranks with marginal fractional utility, *Normalized Discounted Cumulative Gain (NDCG)* (He et al., 2015) is adopted:

$$NDCG(k) = Z_k \sum_{n=1}^{k} \quad (2^{r_i} - 1)\ /\ log_2(i+1) \tag{13}$$

where Z_k is the normalizer to ensure the perfect ranking has a value of 1 and r_i is the graded relevance of the item at position i. If the recommended item is in the

test set $r_i = 1$, and 0 otherwise.

Metrics are calculated by handling purchased product ids as true labels. True labels are the last purchased product id. Besides, recommendation lists with the top, $k = 20$ probabilities of corresponding products are generated from the final output.

4.3.4. Cross-Validation

Instead of splitting data into k-fold subsamples without replacing, a repeated random subsampling validation technique, which is also known as MCCV (Xu & Liang, 2001), is conducted. Since relatively large data-sets the probability of reselecting an observation is relatively low, the iterative random splitting of data (with replacement) as training and validation is considered as appropriate for this study. Figure 4.2 illustrates the sampling and iteration structure of MCCV.

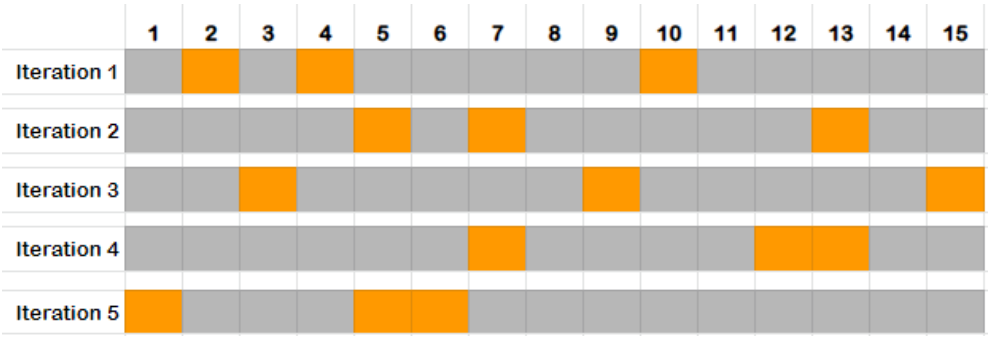

Figure 4.2. Monte Carlo Cross-Validation (Kumar, 2020)

The number of iteration of splitting depends on the size of the data set and study. In this study, MCCV is conducted 5 times randomly splitting the modeling part of the data set as approximately 80% for training and 20% for validation. For validation phase, validation set's last item ids are used as true labels to calculate the validation metrics. On the other hand, one can realize the validation procedure with regards to a predetermined modeling date (See: Section 4.3.5) and true labels from Figure 4.3.

All CV experiments are run with an early stopping callback which is set to 3 as patience on validation scores. For all epochs, models calculate a loss metric on validation set and if they do not obtain any improvement in successive 3 epochs, they stop the running.

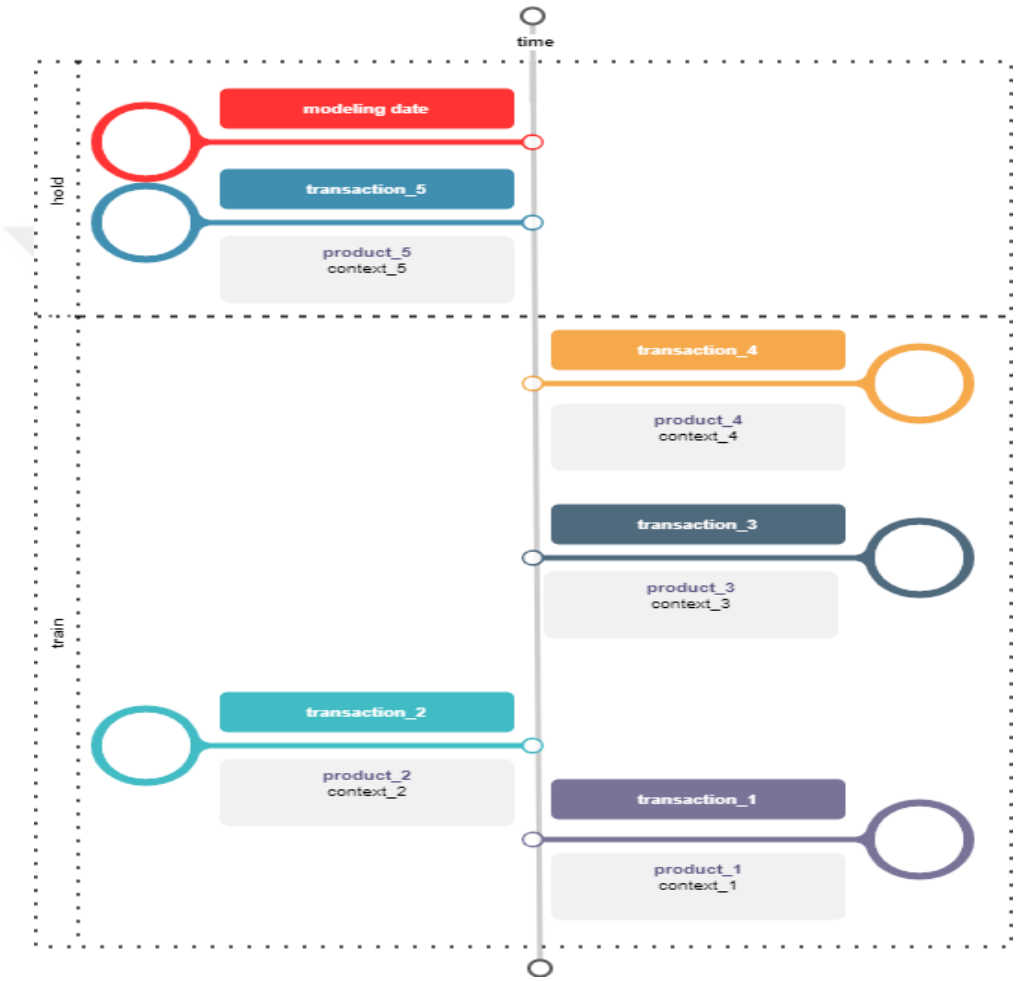

Figure 4.3. Model Validation Procedure

4.3.5. Testing

The testing procedure implemented for observing the evaluation metrics on a previously unseen data set by models. Since the testing could only be conducted on 1 test set, whose summary statistics for purchase number could be observed in

Table 4.6, random sampling errors could not be handled appropriately. There is no different procedure conducted from Figure 4.3, in the testing phase.

Table 4.6. Summary statistics for purchase numbers of customers of the test set.

mean	std.	min	25%	50%	75%	max
23.5	61.8	2.0	4.0	8.0	21.0	855.0

4.3.6. Hyper-Parameter Tuning

Tuning of hyper-parameters are conducted with CV, thanks to Hparams[10] library easily. Applying grid search, some hyperparameters are fixed by pre-testing experiments, due to the high amount of possible parameter combinations. No significant improvement is obtained by changing the learning rate and optimizer which are fixed as 0.0001 and *ADAM*, respectively. First part of experiments are conducted with 0.6 and 0.8 of dropout rates, and dropout rate is fixed with 0.8. Since non-binary outputs are obtained, a *sparse categorical cross-entropy* is used as loss function. On the other hand, there is also no significant improvement to adding extra layers to the models.

Embedding unit size of item ids and RNN unit (or LSTM unit) size are searched for their optimal values in terms of evaluation metrics. One other parameter considered as important is the number of time-steps. A trial and error manner is conducted to converge an optimal number. Experiments are run with *256, 512, 1024,* for embedding size, *400, 600, 800, 1000* for RNN units, and *8, 10, 12* for number of time steps.

User-user is a collaborative filtering model based on customer behavior similarities as neighbors. This model is run with a fixed cosine similarity metric, and different number of neighbor settings, *10, 50, 100, 150* as a single hyper-parameter.

[10] See: https:/tensorflow.org/tensorboard/hyperparameter_tuning_with_hparams

4.4. Results

Results are released in terms of different scenarios. The effect of embedding units, RNN units and number of time-steps could be tracked as result of experiments as well as CV and testing, in terms of two evaluation metrics.

4.4.1. Effect of Hyperparameters

Table 4.7 shows the best hyperparameter configurations of the 4 competing models, in terms of minimum values of CV averages of categorical cross entropy loss. There is no statistical sensitivity analysis conducted between hyperparameter records. Table 4.8 shows the number of parameters for models at their best configurations, which refers to the complexity of the models.

Table 4.7. Hyperparameter tuning results.

	Emb. size	RNN unit	#Time steps	Loss	HR(20)	NDCG(20)
Product-only	512	800	12	0.7984	0.2754	0.2094
Model 2	512	400	12	0.7843	0.3029	0.2259
Model 3	1024	600	12	**0.7634**	0.3169	0.2332
Model 4	512	400	12	**0.7666**	0.3184	0.2330

As mentioned earlier item ids are represented with an embedding mapping of their number encodings. Figure 4.4 illustrates the effects of embedding size on categorical cross entropy loss. Besides, since item catalog size is above 10,000, it is expected that the embedding size affects metrics positively in general. Figures 4.5 and 4.6 illustrate the mentioned impact in terms of CV averages, where the number of time steps is fixed with 12 and RNN units 600.

Table 4.8. Number of parameters of models at their best configuration.

	#Trainable parameters	#Non-trainable parameters
Product-only	21,756,697	1,600
Model 2	15,461,159	4,062
Model 3	29,547,937	6,512
Model 4	17,619,885	5,088

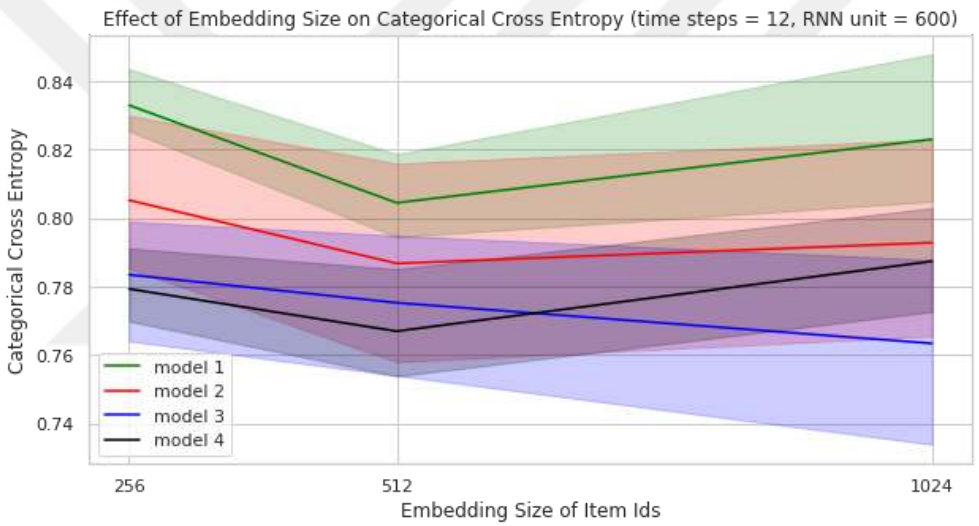

Figure 4.4.Effect of Embedding Size on Categorical Cross Entropy Loss

Model 4 seems to have a different response than others to changes in embedding size with fixed RNN units. In terms of studied analysis, the optimal values of embedding size for relatively simple models such as *Model 1 and 2* are obtained higher than for *Model 3 and 4.*

Figure 4.5. Effect of Embedding Size on HR(20)

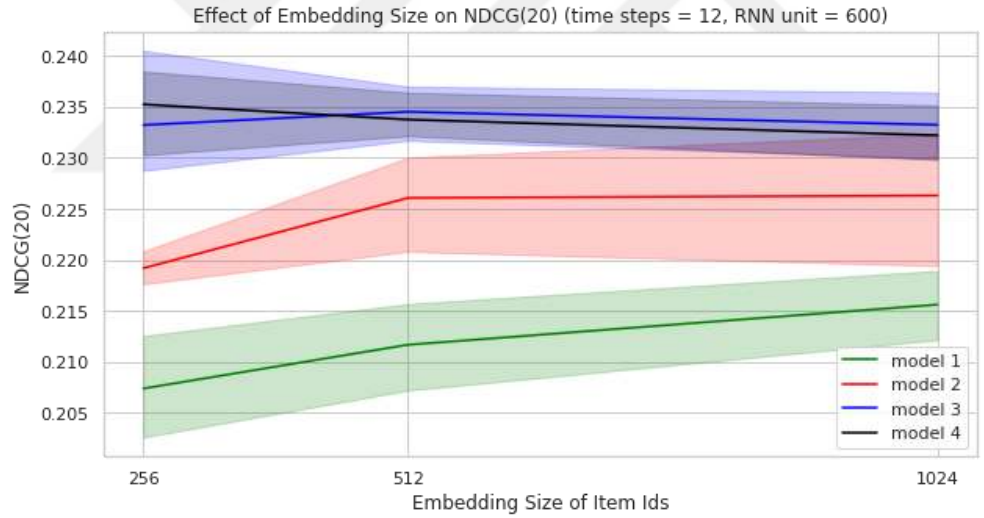

Figure 4.6. Effect of Embedding Size on NDCG(20)

RNN unit size refers to the number of the hidden units of the LSTM block. There is no predetermined optimal value for RNN units, and one can track the impacts of the number of hidden units for this study. Figures 4.7, 4.8 and 4.9 illustrate briefly the impact of the experimented value when other parameters are

fixed with 12 for number of time steps, and 512 for embedding size.

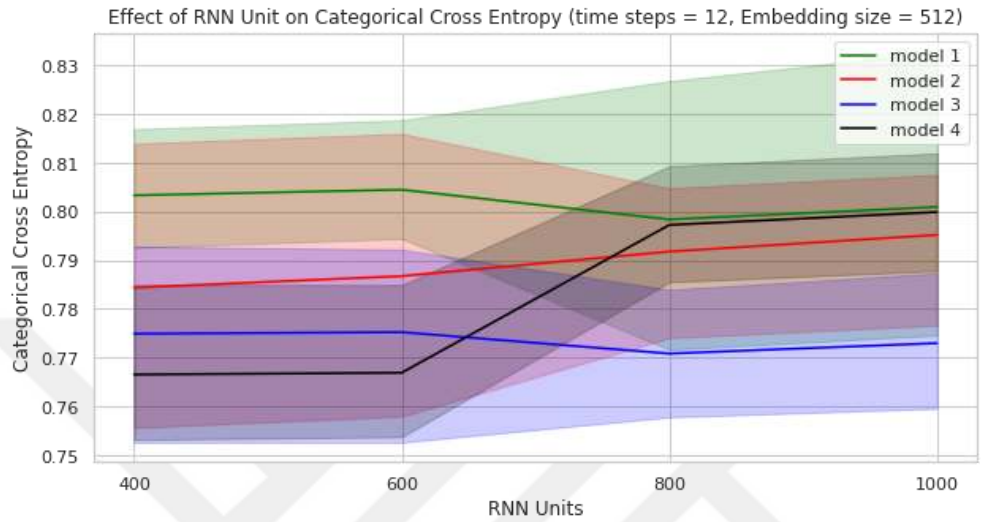

Figure 4.7. Effect of RNN Unit on Categorical Cross Entropy Loss

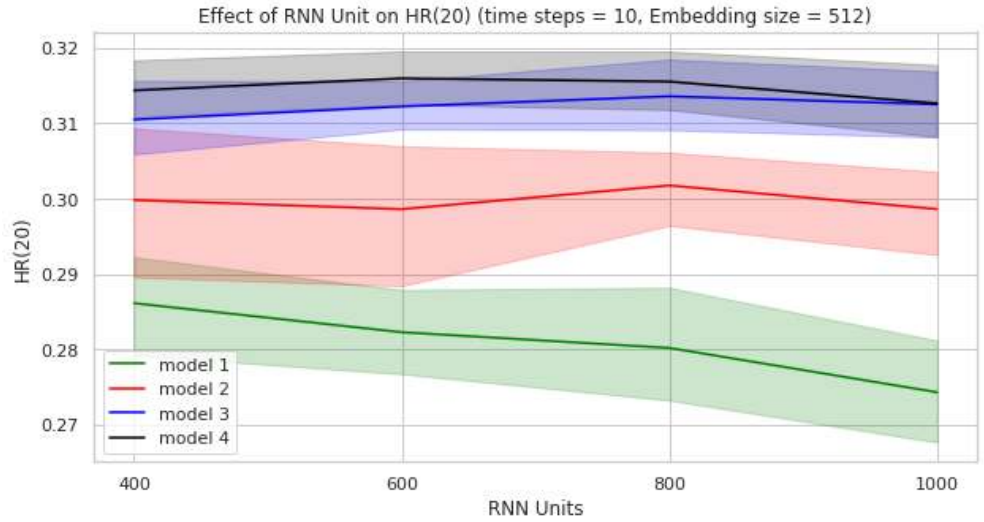

Figure 4.8. Effect of RNN Unit on HR (20)

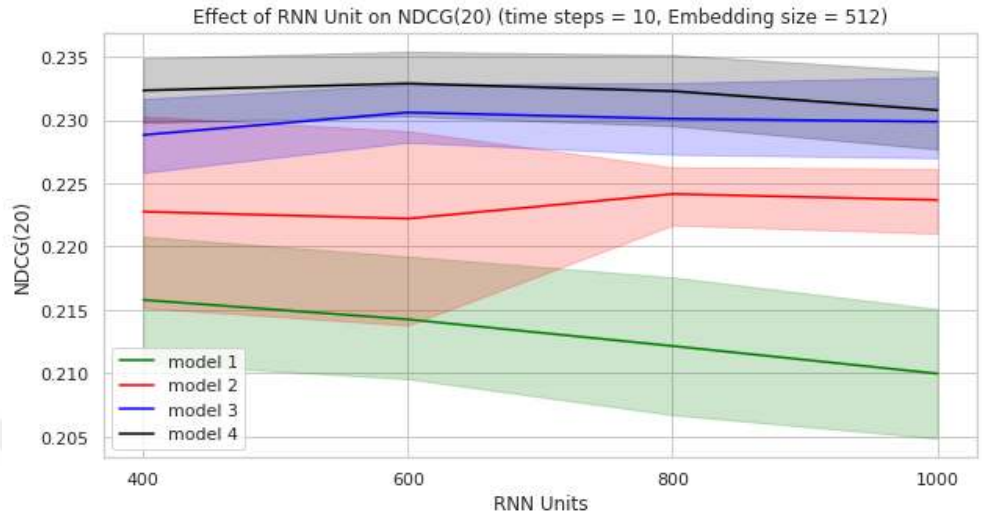

Figure 4.9. Effect of RNN Unit on NDCG (20)

It is obtained that the optimal RNN size for relatively simple models such as *Model 1 and 2* is smaller than the number for *Model 3 and 4* in this study. Furthermore, for *Model 4* relatively higher RNN units seem to affect loss minimization negatively.

Number of time steps refers to the historical depth of purchases for a particular customer. As mentioned earlier, since number of time steps is one of the experimental values, and is treated as a hyperparameter for this study, one can track the impacts of them on metrics and loss function. From Figure 4.9 one can deduct that the number of time steps have a positive effect on loss function at least in the experimented time range. Models 3 and 4 seem to be affected positively from the number of time steps increases in terms of evaluation metrics (See: Figures 4.10 and 4.11). However, for Model 1, looking only at evaluation metrics one cannot have a clear inference about the impacts of the number of time steps.

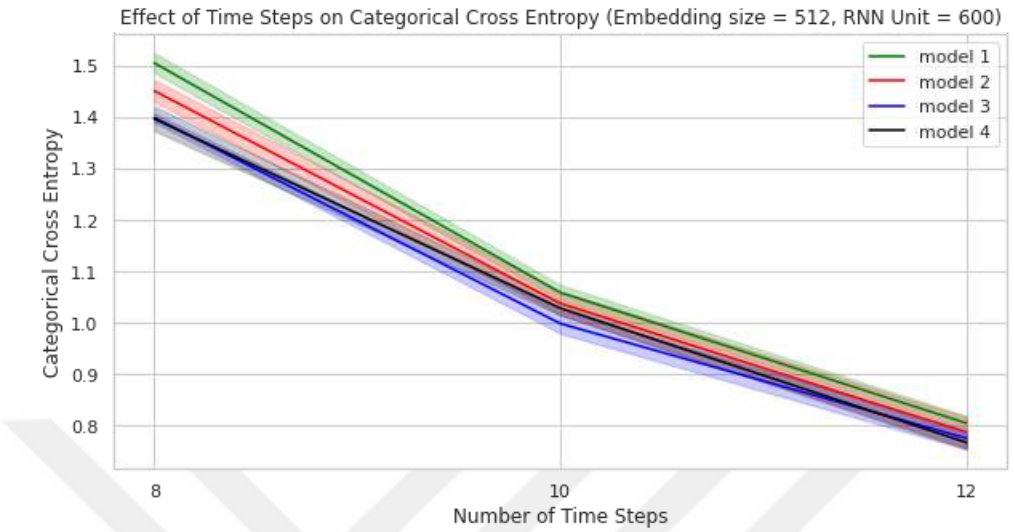

Figure 4.10. Effect of Number of Time Steps Categorical Cross Entropy Loss

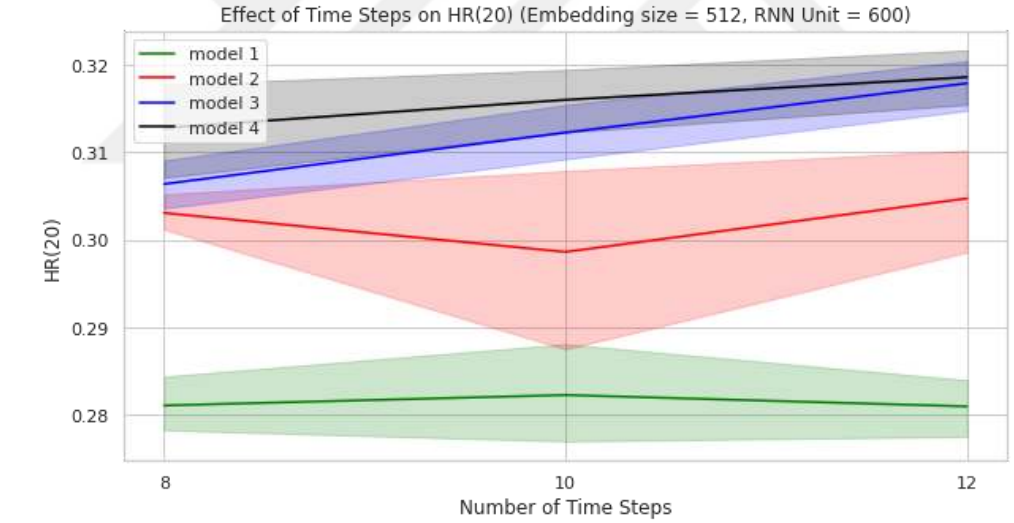

Figure 4.11. Effect of Number of Time Steps on HR(20)

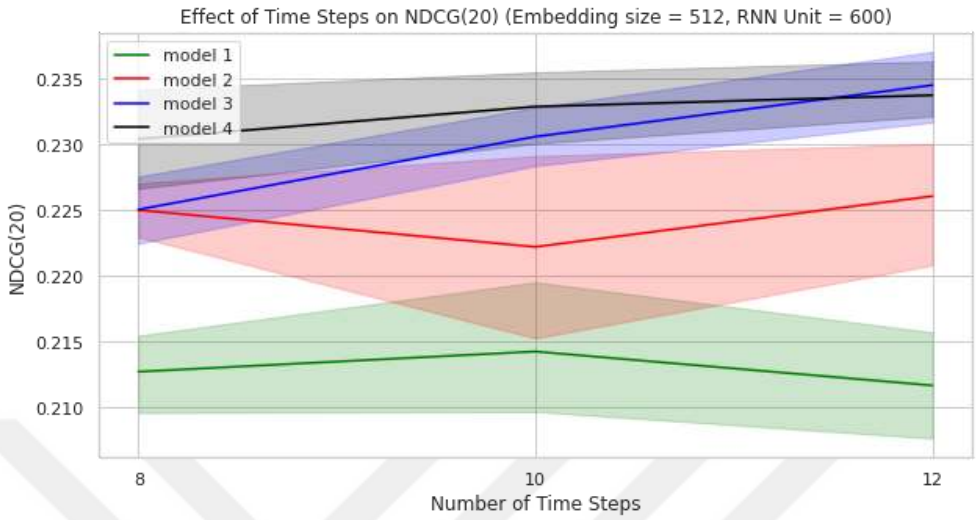

Figure 4.12. Effect of Number of Time Steps on NDCG(20)

4.4.2. Results on Validation Set

After random splitting of the data as training and validation with a proportion of approximately 20% - 80%, the average number of customers of validation sets for all models at their best configuration are obtained as in Table 4.9. Furthermore, models' average number of epochs where they stop early are also represented in Table 4.9.

The average CV results for all models at their best parameter configuration are shown in Table 4.10. *Model 3* and *Model 4,* which are the most complex and the more sequence exploiting models, outperforms base-lines and other models in terms of both two metrics, having very close records to each other. For this data-set, it is obtained approximately 12% in *HR(20)* and 10% in *NDCG(20)* of improvement compared to the *Product-only* model.

Table 4.9. Average number of customers of validation sets and average number of early stopping epochs at models' best configurations.

	#customers	**#epochs**
Popularity	4,790,8	-
Last-random	4.729,4	-
Last-ordered	4,759.6	-
User-user	4,787.8	-
Product-only	4,781.8	38.8
Model 2	4,775.0	41.6
Model 3	4,773.0	32.0
Model 4	4.777,6	26.6

Table 4.10. Average CV results at models' best configurations.

	HR(20)	**NDCG(20)**
Popularity	0.0577	0.0329
Last-random	0.2007	0.1926
Last-ordered	0.1932	0.1694
User-user	0.2816	0.2134
Product-only	0.2754	0.2094
Model 2	0.3030	0.2260
Model 3	**0.3169**	**0.2332**
Model 4	**0.3184**	**0.2330**

4.4.3. Robustness

Statistical significance is obtained as a result of a single-way ANOVA of experiment results and between models with a pairwise manner indeed. As a statistical inference technique, ANOVA is one of the appropriate procedures for

testing the equality of several means (Montgomery, 2013, ch.3). Since this thesis has an argument which is mainly rely on to investigate the impacts of additional sequential features for an RS problem, the single factor, or treatment is considered as feature configuration of 4 models, and *user-similarity* is used as a base model. Table 4.11 and 4.12 shows the 5 random splitting results of models at their best hyper-parameter configuration in terms of *HR(20)* and *NDCG(20)* respectively. Furthermore Figures 4.10 and 4.11 illustrate those results as boxplots.

Table 4.11. CV experiment results for HR(20).

	Number of Random Splits						
	1	**2**	**3**	**4**	**5**	**sum**	**ave**
User-use	0.2830	0.2836	0.2816	0.2772	0.2826	1.4082	0.2816
Model 1	0.2711	0.2764	0.2816	0.2713	0.2765	1.3769	0.2754
Model 2	0.3063	0.3104	0.3068	0.3039	0.2873	1.5147	0.3030
Model 3	0.3142	0.3112	0.3143	0.3164	0.3282	1.5844	0.3169
Model 4	0.3158	0.3154	0.3232	0.3204	0.3172	1.5919	0.3184

Since models are randomized with MCCV procedure, the random effects of data splitting on CV means could be handled appropriately by single-way ANOVA with equation 14, under the assumptions of normal distributed, and have a constant variance of errors for all level of treatments:

$$y_{ij} = \mu + \tau_i + \epsilon_{ij} \tag{14}$$

Where $i = \{1, 2, 3, 4\}$, number of models, $j = \{1, 2, 3, 4, 5\}$ number of splits, y_{ij} is result for model *i*, split *j*, μ is overall mean and τ_i is the effect of the modeling. ϵ_{ij} represents the random error stemming from data splitting.

Table 4.12. CV experiment results for NDCG(20).

	Number of Random Splits						
	1	**2**	**3**	**4**	**5**	**sum**	**ave**
User-use	0.2134	0.2163	0.2134	0.2077	0.2162	1.0670	0.2134
Model 1	0.2047	0.2094	0.2156	0.2087	0.2085	1.0469	0.209
Model 2	0.2299	0.2323	0.2275	0.2265	0.2134	1.1296	0.2259
Model 3	0.2316	0.2279	0.2330	0.2345	0.2392	1.1661	0.2332
Model 4	0.2289	0.2302	0.2410	0.2321	0.2327	1.1649	0.2330

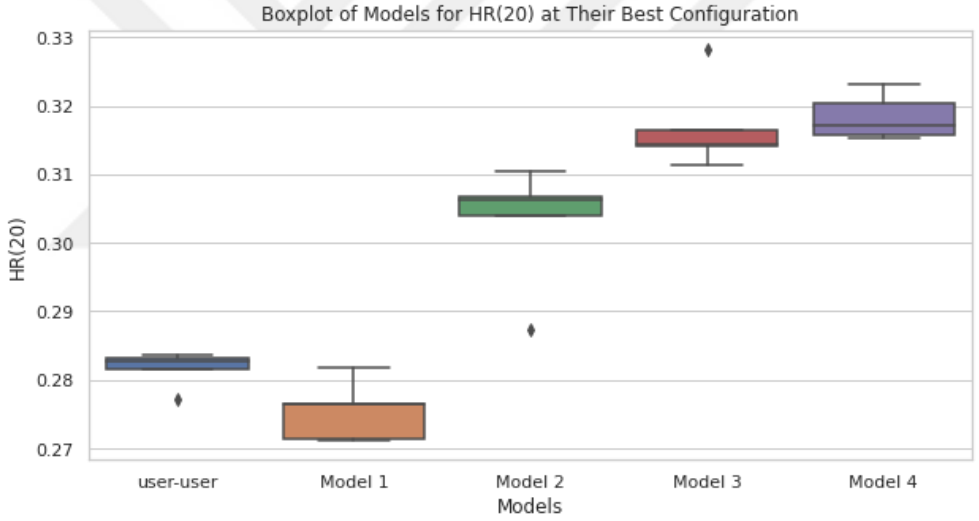

Figure 4 13.. Boxplot of HR(20)

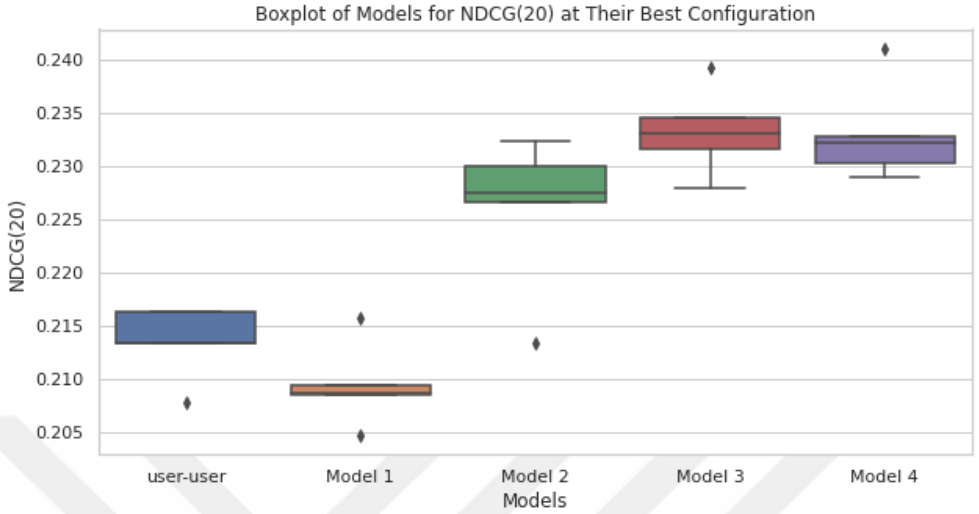

Figure 4.14. Boxplot of NDCG(20)

The hypothesis to be tested for differences between the averages of models results is stated as the equalities of CV means against the difference of at least one of them:

$H_0: \tau_1 = \tau_2 = \tau_3 = \tau_4$

$H_1: \tau_i \neq 0$ *for at least one i*

Overall calculations are obtained by appropriate Python modules, Statsmodel and Bioinfokit, for only *HR(20)* as a response variable. Results show that the modeling effect on the CV means are significant for a significance level, $\alpha = 0.05$. All of the results could be obtained in Appendix C, for 4 modeled *(Model-1 to 4)* and 5 modeled (*plus User-similarity*) systems (See: Table C1 and C3). Besides, it could be obtained that the assumptions are satisfied clearly (See: Figures C1-4).

On the extent of the pairwise comparisons of models Tukey's HSD appears as one of the alternatives (Montgomery, 2013, ch.3). For all pair combinations, test results are calculated and represented in Appendix C. According to pairwise

comparison of CV means, stated as a hypothesis test below, with a significance level of, $\alpha = 0.05$, all pair differences are considered as statistically significant except the pairs of *User-similarity - Model 1* and *Model 3 - Model 4* (See: Table C2-C4).

$$H_0: \mu_i = \mu_j$$
$$H_1: \mu_i \neq \mu_j \text{ for all } i \neq j$$

One can infer that there are no significant differences between the *Model 3 and Models 4* as well as between *Model 1* and the other baseline. On the other hand, other proposed models such as *Model 4 and Model* 3 are significantly superior to all other models.

4.4.4. Results on Test Set

Testing data set is approximately %10 of the data set which is set aside at the beginning of the modeling process. There is no sampling opportunity for this task, therefore no statistical sensitivity could be conducted. Since the test set consists of previously unseen observations, these results could be considered as an estimation of generalization performance of models. However, on the other hand, because of lack of randomness with sampling, it could not be taken as a model selection criterion itself.

Models are run for testing phase with their minimum loss generating configurations in CV section. Each model's performance is tracked during 100 epochs on the test set. Performance metrics are calculated for every epoch, and their graphs are demonstrated in Appendix D (See: Figures D.1, D.2, D.3). Models' best epoch performances are summarized in Table 4.13.

Table 4.13. Test results at models' best configurations, at best epochs.

	HR(20)	**NDCG(20)**
Popularity	0.0467	0.0301
Last-random	0.1904	0.1828
Last-ordered	0.1946	0.1697
User-user	0.2811	0.2113
Product-only	0.2803	0.2087
Model 2	0.2724	0.2029
Model 3	**0.3001**	**0.2147**
Model 4	0.2879	0.2038

Seemingly whereas all RNN models suffer from overfitting, baselines record nearly their validation performance. According to graphs *Model 2* and *4* are the most overfitting models. On the other hand Model 3 outperforms the others with its generalization performance.

5. CONCLUSIONS

In this thesis, one business problem is emphasized as how progressing becomes possible with an existing customer portfolio. In cross-selling aspects, analytical prediction tools are argued as possible solutions. Even though RS applications with RNN are not considered as novel, clearly, it is possible to gain improvement.

Model 3 and *Model 4* both consume contextual and item features as inputs in forms of sequences. *Model 4* additionally uses day values from sign up date to modeling date per user (See: Section 4.3.2), which is not a sequential feature. Results reflect that additional features generally provide an improvement. On the other hand, generating least loss does not always guarantee the best performance, for instance parameters are set with models' least loss value, but best metrics are not recorded at the same configuration. This issue mostly stems from the design of the loss function. In other words performance metrics are not directly proposed models' objectives. As a future research topic ranking loss functions (Hidasi & Karatzoglou, 2018) could be integrated.

Prediction of next products to buy usually becomes a hard task in terms of CRM. One more conclusion is that RS implementation seems profitable for an online retailer which has a deeper catalog. For instance, the best model generates 50 more targetable predictions from the similarity model's on the scale of the test set, 2636 customers. A recommendation list could be usable, if the goal is to find the most likely products each customer will purchase next, and marketable product predictions could be easily transformed to marketable customers by filtering a particular product.

In this study, it is presented that previously proposed RNN models could be extendable by providing additional sequences with a little run-time cost. Besides, looking at CV results, using a non-sequential feature seems promising. To this end, the proposed modeling approach is capable of user profiling. For instance, if

collectible, attributes such as customer age, gender, income level could be integrated to the best model.

Seemingly the attention mechanism provides a powerful aspect to RNN-LSTM models. Therefore integrating contextual and additional item information with an appropriate self-attention mechanism could extract intra dependencies of products.

Like other RS implementations, using RNN could not directly overcome some limitations such as infrequency of both products and customers. Models do not work well with such sparsity. Therefore, customers who do not have enough purchase history, cannot be included in this framework.

One other limitation of the study is known as the session problem. It is obtained that, for some customers, purchases occur at the same time-step. Therefore, there is no sequence for such kinds of customers. In this study that problem is ignored and an arbitrary order for those records.

RNN itself is a valuable tool in this scope. On the other hand, as a future work it could be studied the same issue with the tools from the domain of natural language processing such as *transformers*, or same modeling with session-parallel mini-batches.

REFERENCES

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 734-749. https://doi.org/10.1109/TKDE.2005.99

Akçura, M. T., & Srinivasan, K. (2005). Customer Intimacy and Cross-Selling Strategy. *Management Science*, *51*(6), 1007-1012. https://doi.org/10.1287/mnsc.1050.0390

Ansel, J., & Archibald, T. (2007). Identifying cross-selling opportunities, using lifestyle segmentation and survival analysis. *Marketing Intelligence and Planning*, *25*(4), 394-410. http://dx.doi.org/10.1108/02634500710754619

Baer, D., & Chakraborty, G. (2013). Product Affinity Segmentation Using the Doughnut Clustering Approach. *Proceedings of the SAS Global Forum 2013 Conference*.

Bogaert, M., Lootens, J., Van den Poel, D., & Ballings, M. (2019). Evaluating multi-label classifiers and recommender systems in the financial service sector. *European Journal of Operational Research*, *279*(2), 620-634. https://doi.org/10.1016/j.ejor.2019.05.037

Carta, A. (2021, February 17). *Building a RNN Recommendation Engine with TensorFlow | by Alfonso CARTA | Decathlon Technology*. Medium. Retrieved June 2, 2022, from https://medium.com/decathlontechnology/building-a-rnn-recommendation-engine-with-tensorflow-505644aa9ff3

Cheng, J., Dong, L., & Lapata, M. (2016). Long Short-Term Memory Networks for Machine Reading. *EMNLP*, 551-562.

Chollet, F. (2018). *Deep Learning with Python*. Manning.

Chou, P., Chuang, H.-C., Chou, Y.-C., & Liang, T.-P. (2021). Predictive analytics for customer repurchase: Interdisciplinary integration of buy till you die

modeling and machine learning. *European Journal of Operational Research*, *296*(2022), 635-651. https://doi.org/10.1016/j.ejor.2021.04.021

Devooght, R., & Bersini, H. (2017). Collaborative Filtering with Recurrent Neural Networks.

Donkers, T., Benedikt, L., & Ziegler, J. (2017). Sequential User-based Recurrent Neural Network Recommendations. *Proceedings of RecSys'17*, 27-31. https://doi.org/10.1145/3109859.3109877

Géron, A. (2017). *Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* (first ed.). O'Reilly Media.

Geuens, S., Coussement, K., & De Bock, K. W. (2017). A framework for configuring collaborative filtering-based recommendations derived from purchase data. *European Journal of Operational Research*, *265*(1), 208-218. http://dx.doi.org/10.1016/j.ejor.2017.07.005

He, X., Chen, T., Kan, M.-Y., & Chen, X. (2015). TriRank: Review-Aware Explainable Recommendation by Modeling Aspects. *CIKM '15: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 1661-1670. https://doi.org/10.1145/2806416.2806504

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. (2017). Neural Collaborative Filtering. *Proceedings of the 26th International Conference on World Wide Web*, 173-182. http://dx.doi.org/10.1145/3038912.3052569

Hidasi, B., & Karatzoglou, A. (2018). Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. *The 27th ACM International Conference on Information and Knowledge Management (CIKM'18)*, 843-852. https://doi.org/10.1145/3269206.3271761

Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2016). Session-based Recommendations with Recurrent Neural Networks.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural*

Computation, *9*(8), 1735-1780.

Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative Filtering for Implicit Feedback Datasets. *Eighth IEEE International Conference on Data Mining*, 263-272. http://dx.doi.org/10.1106/ICDM.2008.22

Kamakura, W. A. (2008). Cross-Selling:Offering the Right Product to the Right Customer at the Right Time. *Journal of Relationship Marketing*, *6*(3-4), 41-58. http://dx.doi.org/10.1300/J366v06n03_03

Kamakura, W. A., Wedel, M., Rosa, F., & Mazzon, J. A. (2003). Cross-selling through database marketing: a mixed data factor analyzer for data augmentation and prediction. *International Journal of Research in Marketing*, *20*, 45-65. https://doi.org/10.1016/S0167-8116(02)00121-0

Katrompas, A., & Metsis, V. (2022). Enhancing LSTM Models with Self-attention and Stateful Training. *Intelligent Systems and Applications*, 2017-235. https://doi.org/10.1007/978-3-030-82193-7_14

Knott, A., Hayes, A., & Neslin, S. A. (2002). Next-product-to-buy models for cross-selling applications. *Journal of Interactive Marketing*, *16*(3), 59-75. https://doi.org/10.1002/dir.10038

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer*, *42*(8), 30-37. https://doi.org/10.1109/MC.2009.263

Kumar, S. (2020, September 13). *Understanding 8 types of Cross-Validation | by Satyam Kumar*. Towards Data Science. Retrieved June 15, 2022, from https://towardsdatascience.com/understanding-8-types-of-cross-validation-80c935a4976d

Lee, D. D., & Seung, H. S. (2000). Algorithms for non-negative matrix factorization. *Proceedings of the 13th International Conference on Neural Information Processing Systems*, 535-541. https://dl.acm.org/doi/10.5555/3008751.3008829

Li, S., Kawale, J., & Fu, Y. (2015). Deep Collaborative Filtering via Marginalized

Denoising Auto-encoder. *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM '15)*, 811-820. https://doi.org/10.1145/2806416.2806527

Li, W., Qi, F., Tang, M., & Yu, Z. (2020). Bidirectional LSTM with self-attention mechanism and multi-channel features for sentiment classification. *Neurocomputing*, *387*, 63-77. https://doi.org/10.1016/j.neucom.2020.01.006

Lü, L., Medo, M., Yeung, C., Zhang, Y., Zhang, Z., & Zhou, T. (2012). Recommender systems. *Physics Reports*, *519*, 1-49. http://dx.doi.org/10.1016/j.physrep.2012.02.006

Martinez, A., Schmuck, C., Pereverzyev, S., Pirker, C., & Haltmeier, M. (2020). A machine learning framework for customer purchase prediction in the non-contractual setting. *European Journal of Operational Research*, *281*(3), 588-596. https://doi.org/10.1016/j.ejor.2018.04.034

Montgomery, D. C. (2013). *Design and Analysis of Experiments* (Eighth ed.). Wiley.

Moon, S., & Russell, G. J. (2012). Predicting Product Purchase from Inferred Customer Similarity. *Management Science*, *54*(1), 71-82. https://doi.org/10.1287/mnsc.1070.0760

Reinartz, W. J., & Kumar, V. (2003). The impact of customer relationship characteristics on profitable lifetime duration. *Journal of Marketing*, *67*, 77-99. https://doi.org/10.1509/jmkg.67.1.77.18589

Salakhutdinov, R., & Mnih, A. (2007). Probabilistic matrix factorization. *Proceedings of the 20th International Conference on Neural Information Processing System*, 1257-1264. https://dl.acm.org/doi/10.5555/2981562.2981720

_____, (2008). Bayesian probabilistic matrix factorization using markov chain monte carlo. *Proceedings of the 25th international conference on Machine learning*, 880-887. https://doi.org/10.1145/1390156.1390267

Salakhutdinov, R., Mnih, A., & Hinton, G. (2007). Restricted Boltzmann machines

for collaborative filtering. *Proceedings of the 24th international conference on Machine learning*, 791-798. https://doi.org/10.1145/1273496.1273596

Salehinejad, H., & Rahnamayan, S. (2016). Collaborative Filtering for Implicit Feedback Datasets. *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 1-6. https://doi.org/10.1109/SSCI.2016.7849921.

Smirnova, E., & Vasile, F. (2017). Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks. *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*, 2-9. https://doi.org/10.1145/3125486.3125488

Tan, Y. K., Xu, X., & Liu, Y. (2016). Improved Recurrent Neural Networks for Session-based Recommendations. *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 17-22. https://dx.doi.org/10.1145/2988450.2988452

Thuring, F., Nielsen, J. P., Guillen, M., & Bolance, C. (2012). Selecting prospects for cross-selling financial products using multivariate credibility. *Expert Systems with Applications*, *39*, 8809-8816. https://doi.org/10.1016/j.eswa.2012.02.011

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *31st Conference on Neural Information Processing Systems (NIPS 2017)*. https://doi.org/10.48550/arXiv.1706.03762

Verstrepen, K., Bhaduriy, K., Cule, B., & Goethals, B. (2017). Collaborative Filtering for Binary, Positiveonly Data. *ACM SIGKDD Explorations Newsletter*, *19*(1), 1-21. https://doi.org/10.1145/3137597.3137599

Wu, S., Ren, W., Yu, C., Chen, G., Zhang, D., & Zu, J. (2016). Personal recommendation using deep recurrent neural networks in NetEase. *IEEE 32nd International Conference on Data Engineering (ICDE)*, 1218-1229. https://doi.org/10.1109/ICDE.2016.7498326

Xu, Q.-S., & Liang, Y.-Z. (2001). Monte Carlo cross validation. *Chemometrics and*

Intelligent Laboratory Systems, *56*(1), 1-11. https://doi.org/10.1016/S0169-7439(00)00122-2

Zhang, L., Priestley, J., De Maio, J., Ni, S., & Tian, X. (2021). Measuring Customer Similarity and Identifying Cross-Selling Products by Community Detection. *Big Data*, *9*(2), 132-143. http://dx.doi.org/10.1089/big.2020.0044

Zhang, S., & Yao, L. (2019). Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Computing Surveys*, *52*(1), 1-38. https://doi.org/10.1145/3285029

CURRICULUM VITAE

İbrahim Erdem KALKAN received her Bachelor of Science (B.S) degree in Management Engineering Department from İstanbul Technical University in 2008. He worked for a state-bank with different departments such as internal control, data and reporting from 2010 to 2020. He started his Master of Science (MSc) education in the Industrial Engineering Department of Çukurova University in 2019. He has been working as a data scientist in the research and development department of an IT company since the beginning of 2022. His works are focused mainly on robotic process mining, computer vision in robotic processes, neural networks applications in the domain of banking and finance.

APPENDICES

APPENDIX A

Table A.1. First 25 records of the data-set

t_id	t_date	c_id	price	amount	c_since	item_id	category
211135	2016-07-01	4	360	1	2016-07-01	1	1
211138	2016-07-01	7	360	1	2016-07-01	1	1
211145	2016-07-01	11	120	1	2016-07-01	2	2
211149	2016-07-01	13	420	1	2016-07-01	3	3
211150	2016-07-01	13	360	1	2016-07-01	4	3
211151	2016-07-01	13	490	1	2016-07-01	5	1
211155	2016-07-01	15	149	1	2016-07-01	6	4
211162	2016-07-01	16	500	1	2016-07-01	7	5
211163	2016-07-01	16	100	5	2016-07-01	8	5
211166	2016-07-01	19	450	1	2016-07-01	9	4
211168	2016-07-01	20	20999	1	2016-07-01	10	6
211169	2016-07-01	20	360	1	2016-07-01	1	1
211175	2016-07-01	22	90	1	2016-07-01	11	3
211193	2016-07-01	30	425	1	2016-07-01	12	3
211198	2016-07-01	34	510	1	2016-07-01	14	3
211199	2016-07-01	34	325	1	2016-07-01	15	3
211200	2016-07-01	35	300	1	2016-07-01	16	1
211203	2016-07-01	37	350	2	2016-07-01	17	3
211212	2016-07-01	35	360	1	2016-07-01	1	1
211217	2016-07-01	43	360	1	2016-07-01	1	1
211220	2016-07-01	13	760	1	2016-07-01	18	3
211221	2016-07-01	13	435	1	2016-07-01	19	3

APPENDIX B

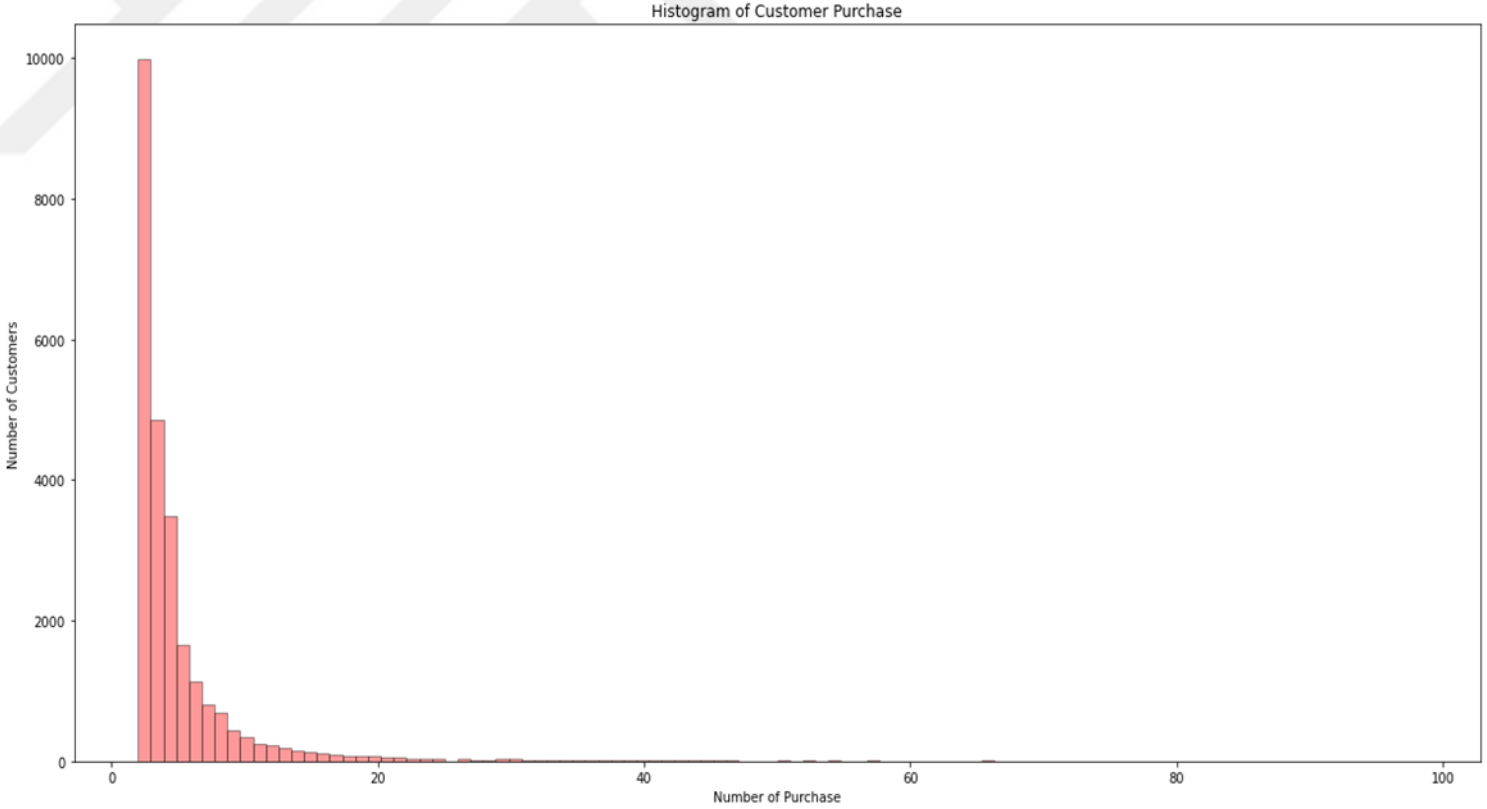

Figure B.1. Distribution of Purchase Number of Customers (for first 100 frequency)

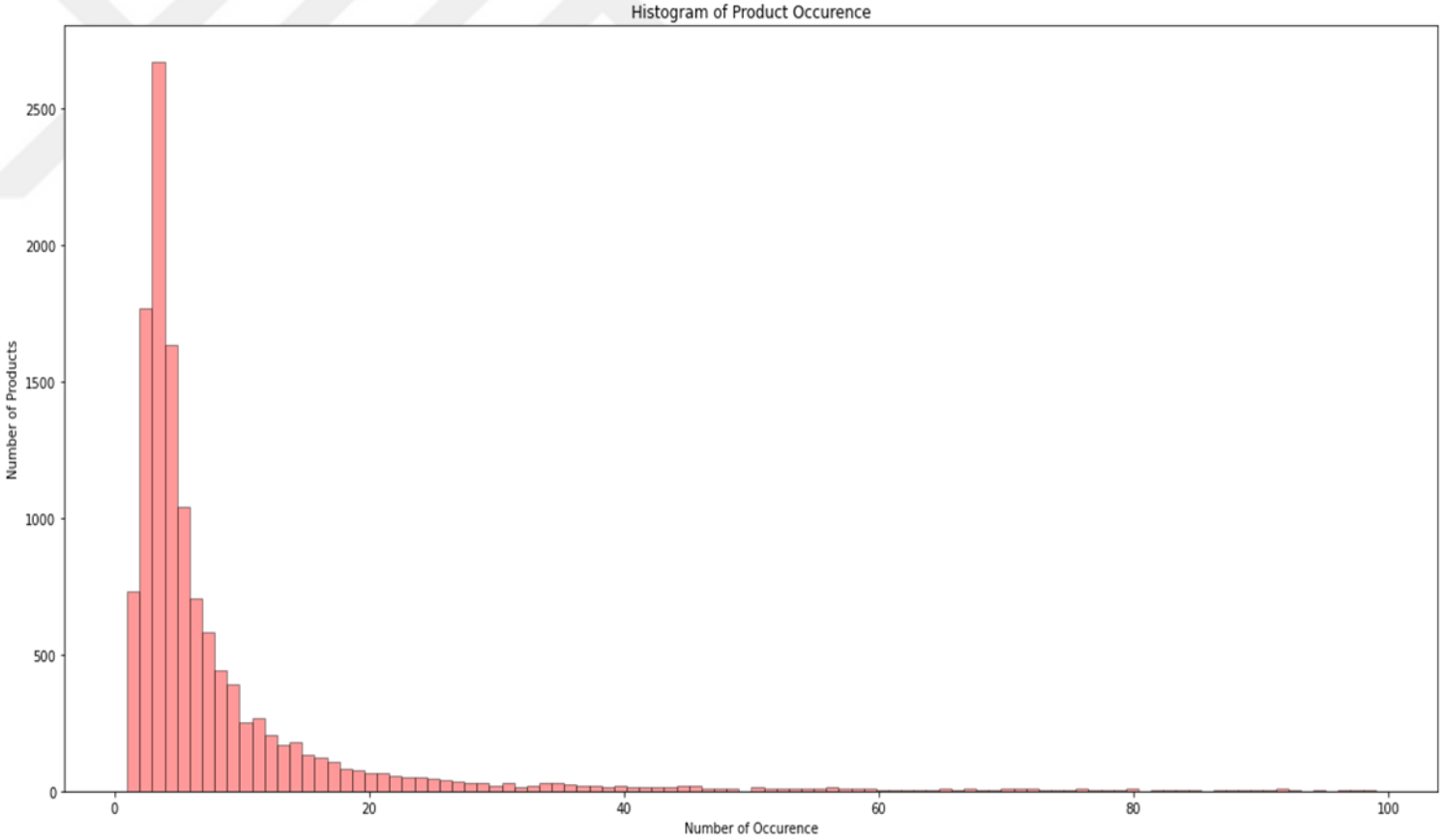

Figure B.2. Distribution of Number of Occurrence of Products (for first 100 occurrence)

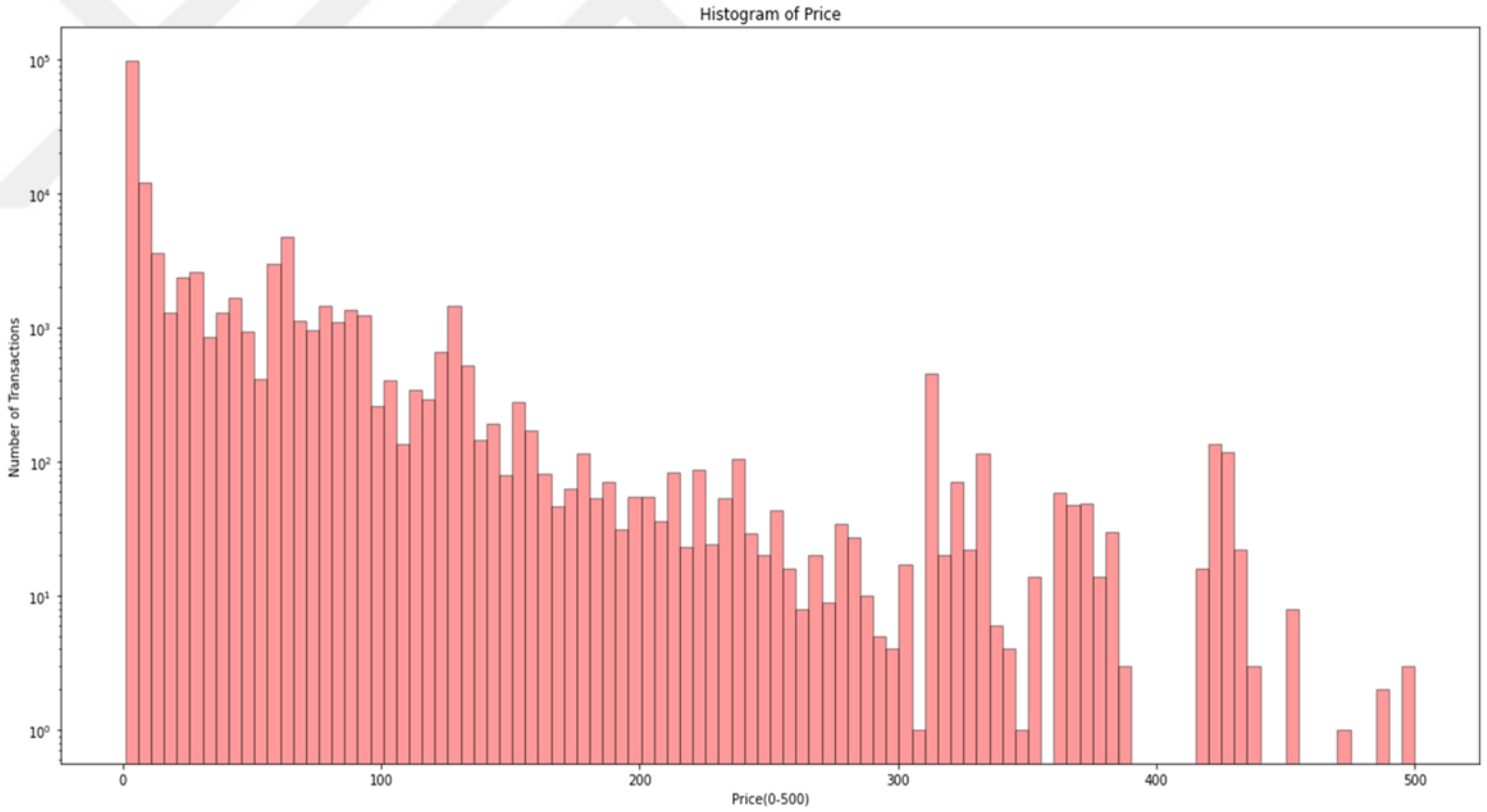

Figure B.3. Distribution of Price After Categorization

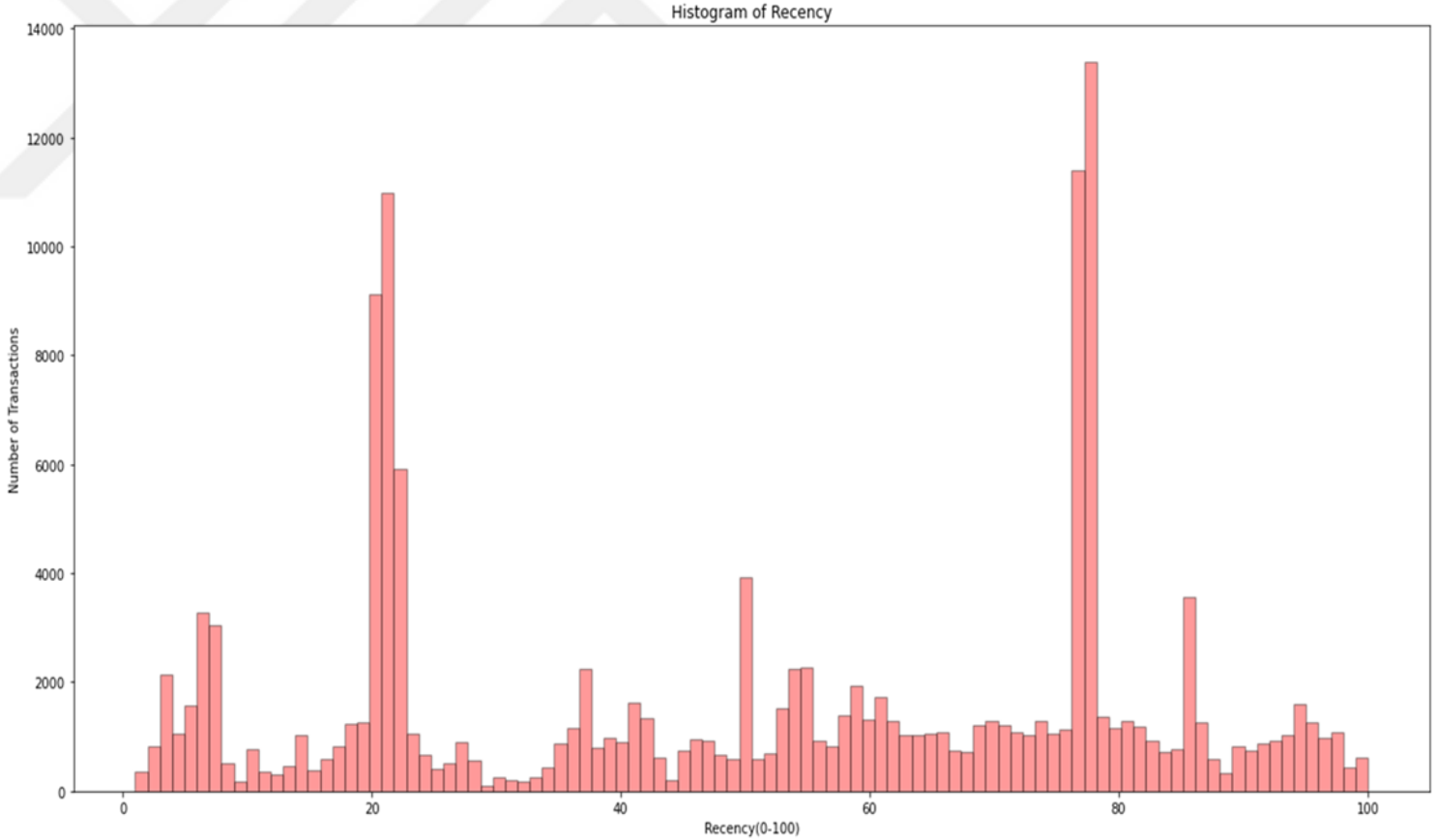

Figure B.4. Distribution of Recency after Categorization

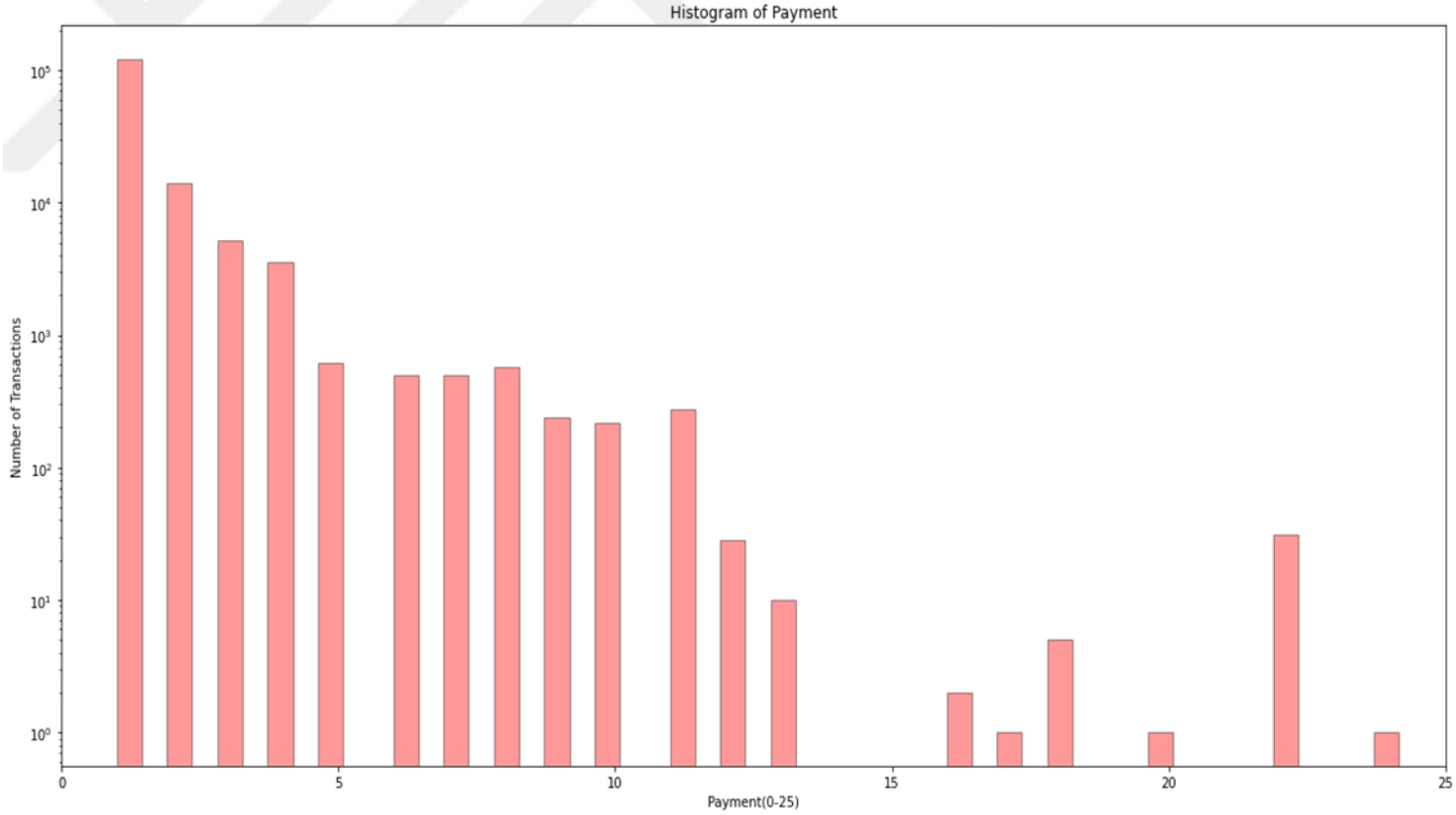

Figure B.5. Distribution of Payment after Categorization

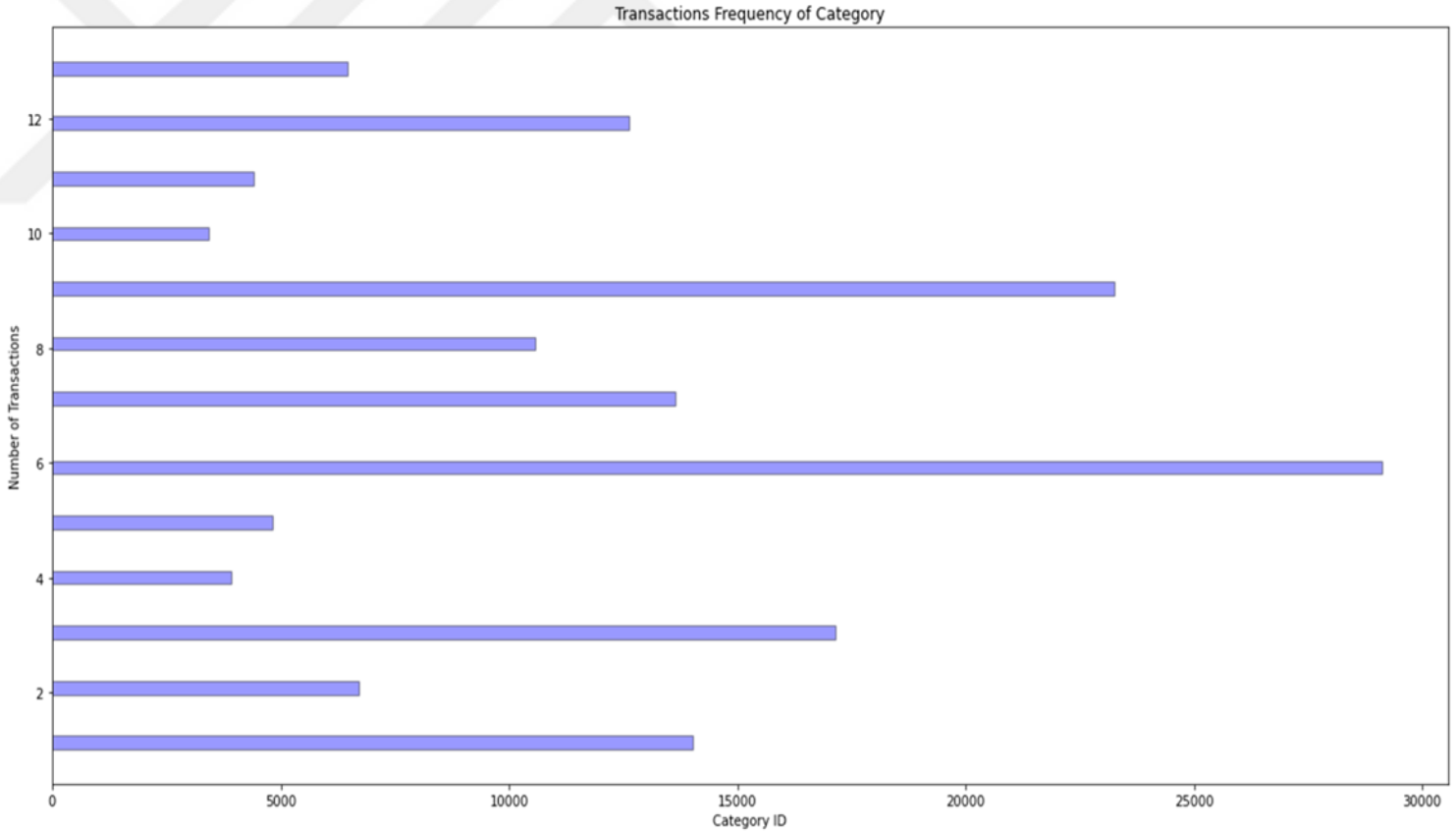

Figure B.6. Product Category Frequency after Processing

APPENDIX C

Table C.1. Single-way ANOVA with 5 models (Output of Python-Statsmodel)

	sum_sq	df	F	PR(>F)
C(model)	0.007848	4.0	60.550205	**6.825615e-11**
Residual	0.000648	20.0		

Table C.2.Tukey's HSD with 5 models (Output of Python-Bioinfokit)

group1	group2	Diff	Lower	Upper	q-value	p-value
User-user	Model 1	0.006251	-0.004523	0.017025	2.455508	0.437272
User-user	Model 2	0.021312	0.010538	0.032086	8.371524	**0.001000**
User-user	Model 3	0.035246	0.024472	0.046020	13.845087	**0.001000**
User-user	Model 4	0.036733	0.025959	0.047507	14.428937	**0.001000**
Model 1	Model 2	0.027563	0.016789	0.038337	10.827032	**0.001000**
Model 1	Model 3	0.041497	0.030723	0.052272	16.300595	**0.001000**
Model 1	Model 4	0.042984	0.032210	0.053758	16.884445	**0.001000**
Model 2	Model 3	0.013934	0.003160	0.024708	5.473562	**0.007536**
Model 2	Model 4	0.015421	0.004647	0.026195	6.057413	**0.002980**
Model 3	Model 4	0.001486	-0.009288	0.012260	0.583850	0.900000

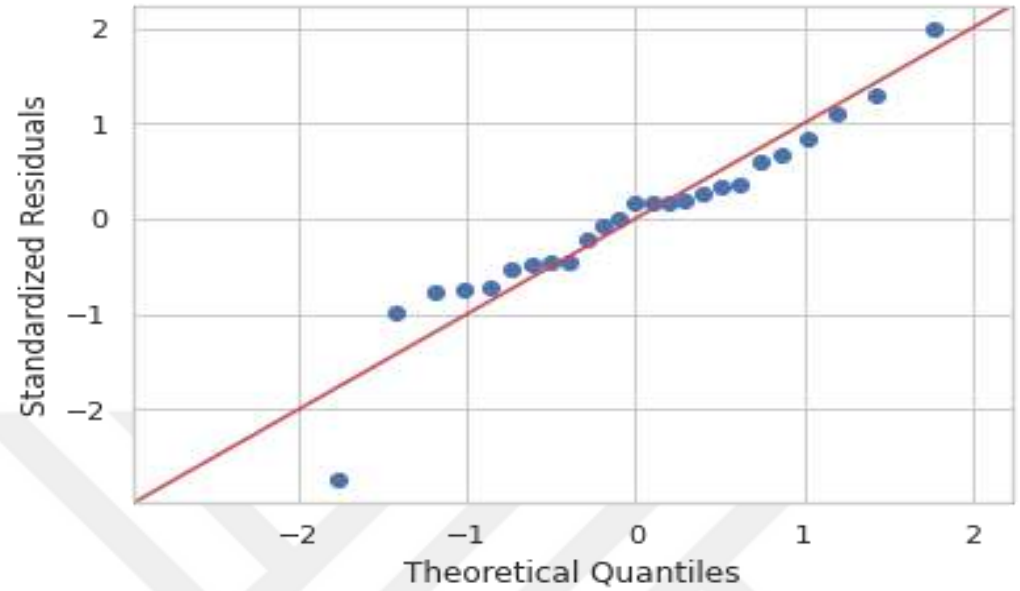

Figure C.1. QQ-Plot for Residuals of ANOVA with 5 Models

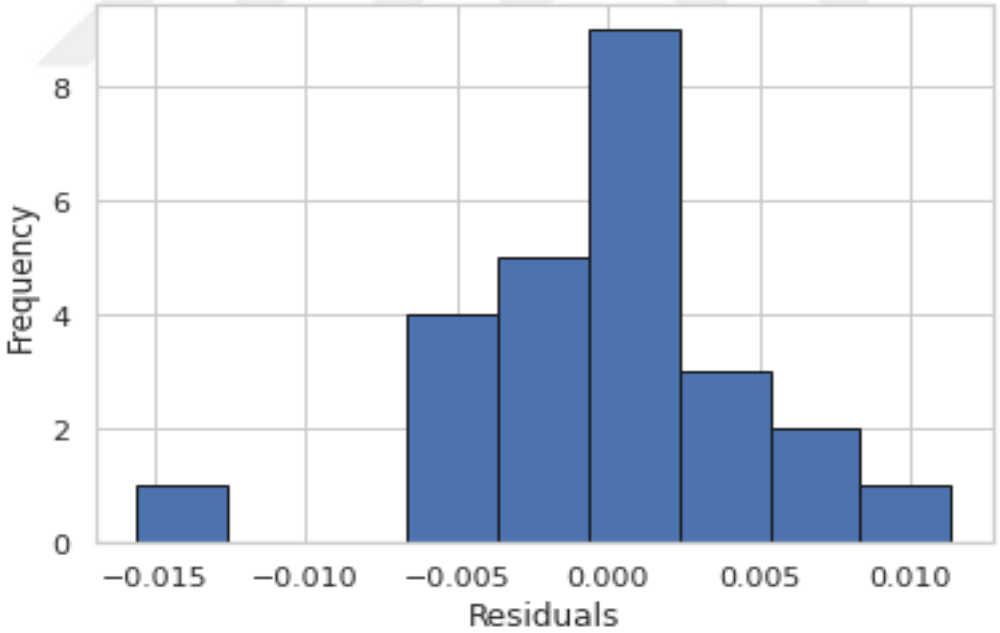

Figure C.2. Residuals Histogram of ANOVA with 5 Models

Table C.3.Single-way ANOVA with 4 models (Output of Python-Statsmodel)

	sum_sq	df	F	PR(>F)
C(model)	0.005954	3.0	51.085931	**2.037294e-08**
Residual	0.000622	16.0		

Table C.4.Tukey's HSD with 4 models (Output of Python-Bioinfokit)

group1	group2	Diff	Lower	Upper	q-value	p-value
Model 1	Model 2	0.027563	0.016283	0.038843	9.887890	**0.001000**
Model 1	Model 3	0.041497	0.030218	0.052777	14.886673	**0.001000**
Model 1	Model 4	0.042984	0.031704	0.054263	15.419880	**0.001000**
Model 2	Model 3	0.013934	0.002655	0.025214	4.998783	**0.013194**
Model 2	Model 4	0.015421	0.004141	0.026700	5.531990	**0.006136**
Model 3	Model 4	0.001486	-0.009793	0.012766	0.533207	0.900000

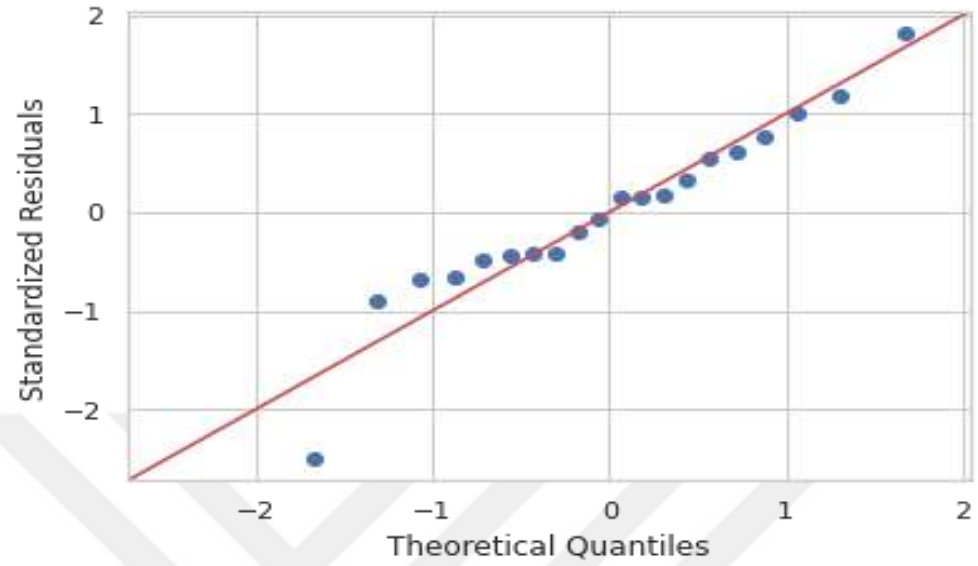

Figure C.3. QQ-Plot for Residuals of ANOVA with 4 Models

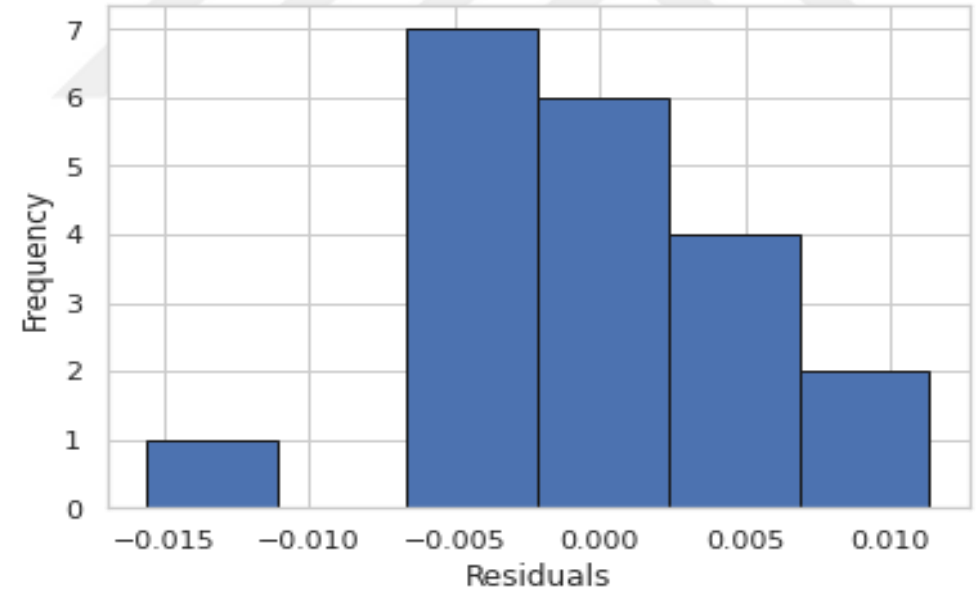

Figure C.4. Residuals Histogram of ANOVA with 4 Models

APPENDIX D

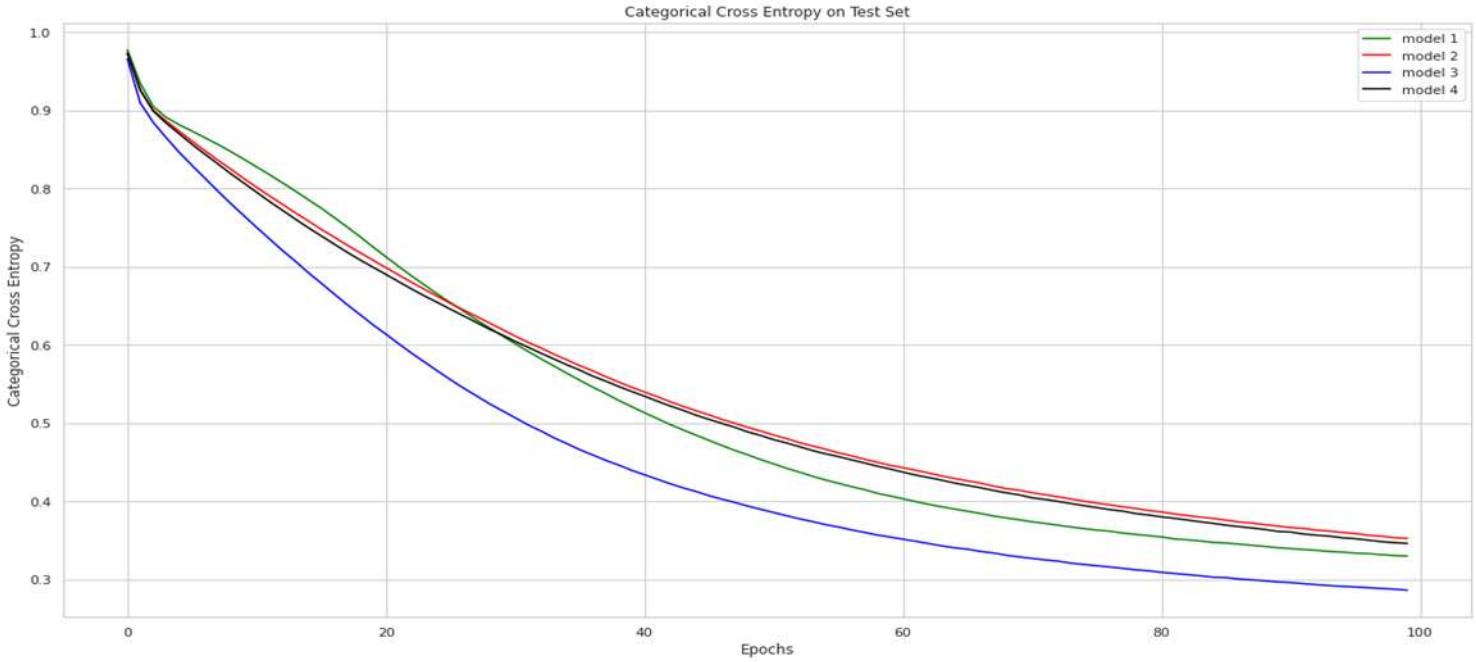

Figure D.1. Loss Tracking of Each Epoch on the Test Set

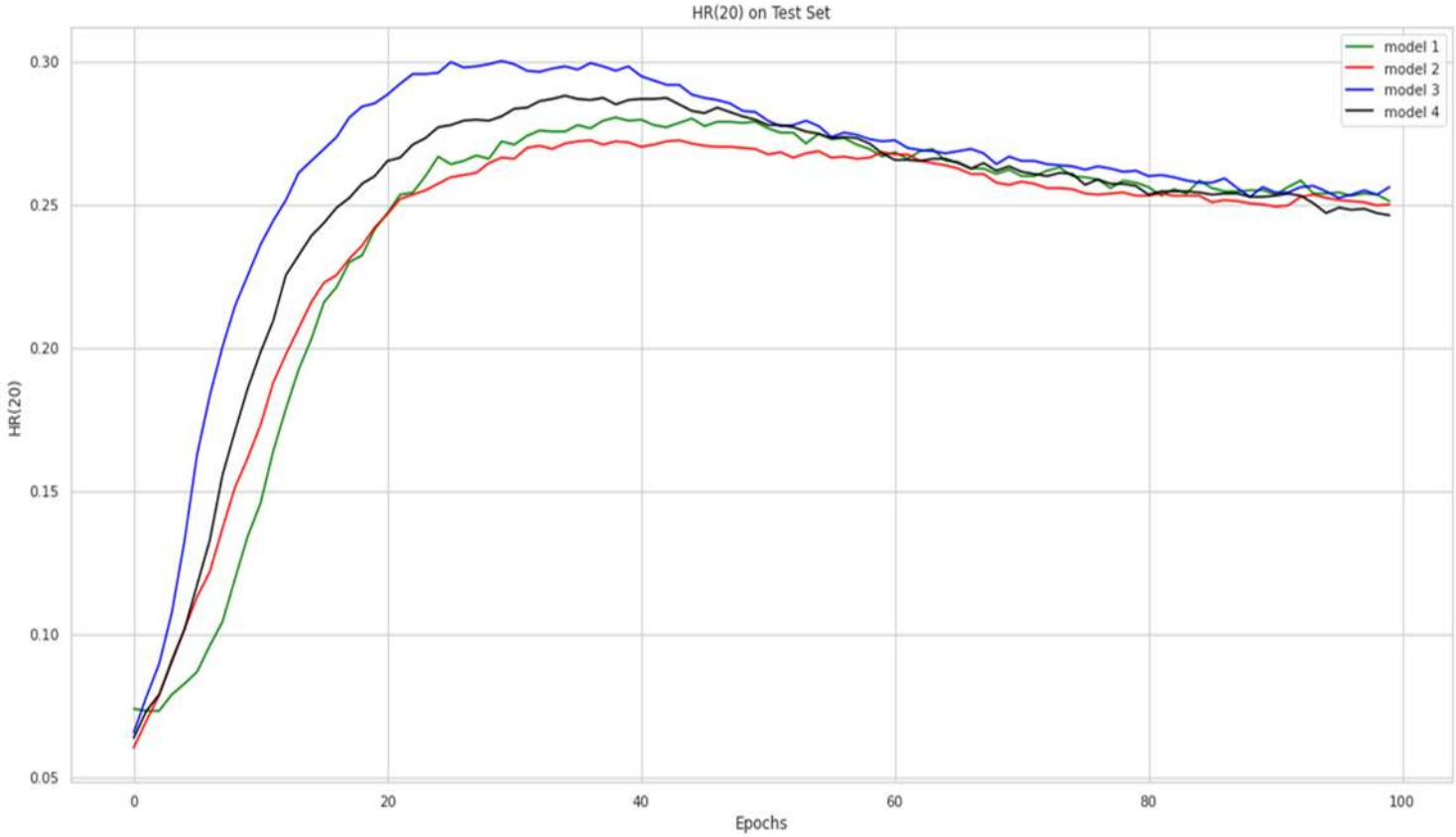

Figure D.2. HR(20) Tracking of Each Epoch on the Test Set

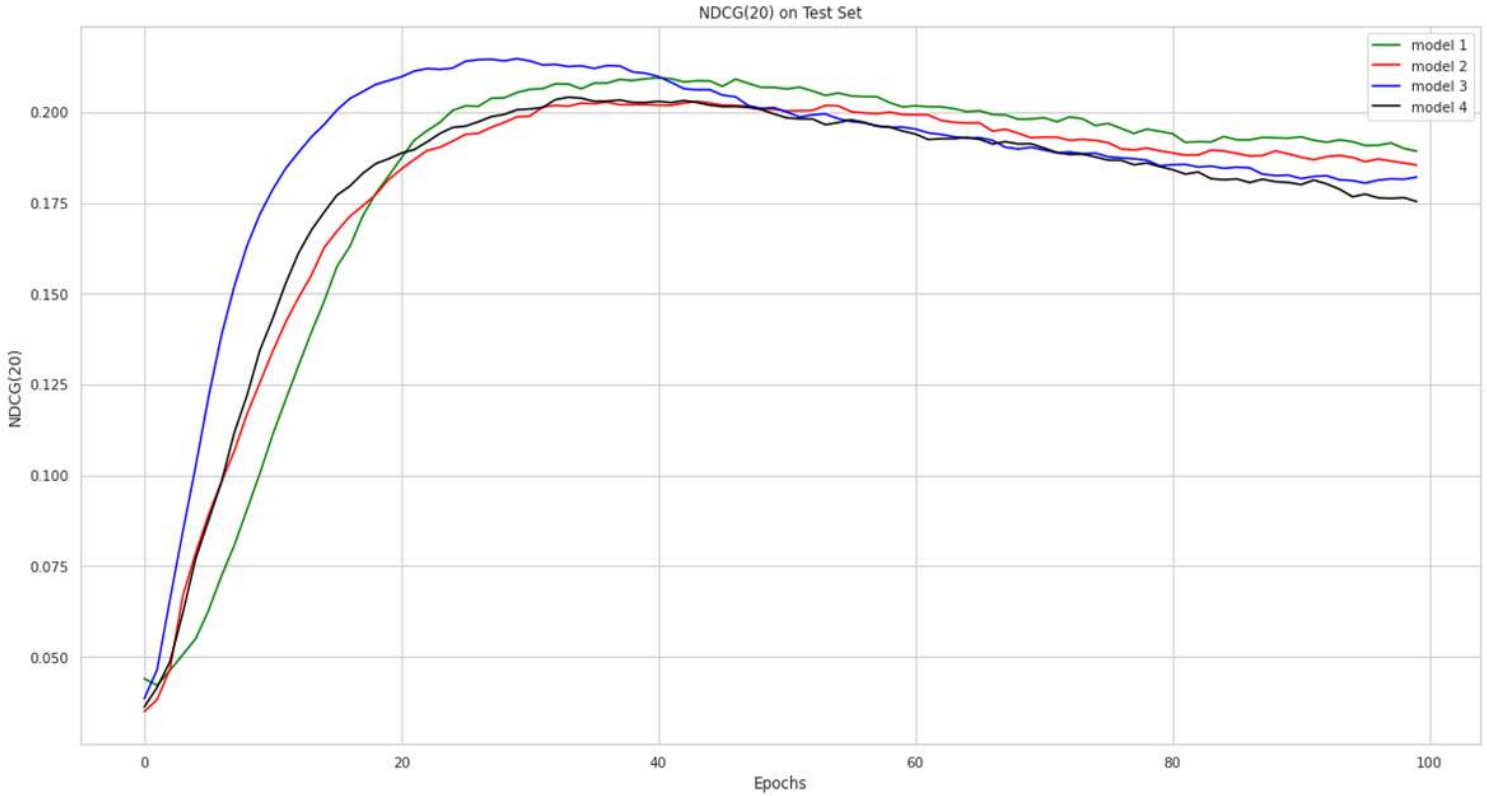

Figure D.3. NDCG(20) Tracking of Each Epoch on the Test Set