Graph-based knowledge modeling and analytics for capturing and prediction of customer behaviour

by

Heba Zahran

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfilment of the requirements for the degree of

Master of Information Technology

in

Digital Media with Specialisation in Data Science

Carleton University

Ottawa, Ontario

Abstract

Understanding customer behaviour is a challenging problem. While the customer produces a large amount of data with each touch point, most of the proposed models focus on one data source in their predictive analysis approaches. This research proposes a customer profile model based on 360 customer view. To this end, we first model a simplified data model and the basic entities based on the existing models. Then, we perform extensive feature engineering techniques, including extracting new features and transforming features to enhance their behaviour in the predictive model. Through the experimentations, we show that the models based on graphs achieve good performance. To this end, we propose a graph-based neural network capable of multitasking without sacrificing the task's performance. We examine three tasks to predict customer intentions. The final results reveal that the set of features with customer information from different data sources positively influences the predictive algorithms' performance.

Acknowledgement

Foremost, I would like to give my warmest thanks to my supervisor, Dr. M. Omair Shafiq for his continuous support of my research, for his patient, motivation, enthusiasm, and immense knowledge. Through out this research, his guidance helped me to explore new ideas, techniques and helped me to meet milestones and accomplish tasks. Dr. Omair has always been a teacher and supervisor who provides guidance and constructive feedback. It was a privilege and an honour for me to have friendly discussions during every research meeting.

Many thanks also extend to Dr. David Sprague, Dr. Gerry Chan and Dr. Elio Velazquez for their motivating pieces of advice that helped me working on exciting projects in the first year of the Master of Information Technology.

I am extremely grateful to my parents for their love, prayers, and sacrifices for preparing me for my future. I would like to thank my husband and my kids for their love, understanding, prayers and continuing support to complete this research.

Table of Contents

List of Figures

List of Tables

Table 0.1 Abbreviations

Abbreviations	Definition
360CV	The 360-customer view
CPDM	Customer Profile Data Model
CBPred	Customer Behaviour Predictive Model
RFM	Recency, Frequency, and Monetary
CLV	Customer Lifetime Value
ERM	Entity Relationship Model
PI	Purchase Intention
B2C	Business to Customer
ML	Machine Learning
LSTM	Long- Short Time Memory
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
FNN	Feed Forward Neural Network
GNN	Graph Neural Networks
SML	Standard Machine Learning Algorithms
LR	Logistic Regression
DT	Decision Tree Classifier
RF	Random Forest Classifier
SVC	Support Vector Classification
NB	Gaussian Naive Bayes Classifier

CHAPTER 1 - INTRODUCTION

Predicting customer behaviour is one of the most important businesses applications with the presence of significant competitions in the e-commerce world. The customers create a massive amount of data when they interact with several touch points with the business which is a valuable recourse to understand the customer. With the rise of big data management systems, mining and analysing this amount of data can reveal new information about the customer and their intentions, such as what is likely to happen (Teixeira et al. 2012). Predictive big data analysis helps to predict the upcoming trends and possibilities based on customer's behaviour with the implementation of several tools and methods, such as forecasting models, regression models and classification models.

Modern businesses need modern data models and analytics tools to take advantage of the available data and obtain new information they need to improve the marketing plan, develop the services, and put a strategy of engagement based on customer behaviour. Several research proposed prediction models to analyze customer behaviour data, and predict their upcoming behaviour based on their previous activities. An investigation of customers reviews Airbnb services in (Hang et al. 2019) reveals that there is a gap between customer's experience and their expectations. A model proposed by (Wong and Wei 2018) predicts the next purchase of the high-value customers of an online air travel corporation based on the online purchasing history of their customers. The changing of customer's behaviour was analyzed by (Sundararaj and Rejeesh 2021) conducting customer reviews. They applied deep analysis to extract the purchasing actions, brand behaviour and the influence of the price on purchasing decisions, and they found that the main factors on buying behaviour are the customer's profile and the social media. The works done by (Guo et al. 2019) explored real-time purchasing intent prediction with traditional browse-interactive behaviour and touch interactive behaviour, while

(Peng et al. 2020) collected data from edge devices and static user information. (Zheng et al. 2020) proposed a model to predict the user's purchasing intention based on the item price.

There are current methods to analyze some elements in the customer experience data, however, there is still a need for methods to analyze the whole context of the customer journey (Holmlund et al. 2020). It is possible to capture the customer information at the points where they interact with business. For example, the products they purchased, how frequently they visit the business website, and their satisfaction with the business. The view from different points is referred to as 360-degree customer view (Wyner, 2001).

1.1 Background

1.1.1 Modeling Customer Behaviour:

Customer modeling is a technical approach based on adopting systems to the targeted customer needs. Then, the relevant marketing content is delivered to the targeted customer to become a customer. Customer behaviour is to learn behaviours of groups of customers and their actions, such as purchasing or leaving the service or the product. Investigating customer demographics, personality, lifestyles, loyalty, purchasing patterns -when, and what - and their satisfactions reveals what they want and their patterns (Cole 2007). The discovered patterns are the key to predict how similar customers will behave under identical circumstances, such as predicting customers' churn (Ahmad, Jafar, and Aljoumaa 2019), and purchasing intention (Martínez et al. 2020).

1.1.2 Data Sources for Customer Modeling

Defining the source of information needed for the customer model is based on the customer experience with direct and indirect communication with the company. The direct contact is usually initiated with the customer, such as purchasing actions. While the indirect contact is a result of unplanned communication between the company's product and the

customer, such as advertising, and reviews (Meyer and Schwager 2007). Based on these contacts, a multidimensional customer experience is constructed with the "customer's cognitive, affective, social and physical responses" during their journey at any touch point (Stein and Ramaseshan 2019). For example, customer's feedback through different channels, or the results of tracking customers' behaviours on the website or while using their digital devices (e.g., mobile phone) (HUSPI 2020). In (Holmlund et al. 2020), they classified customer's experience insights based on customer's attitude, personality traits and physical behaviour. The behavioural insights help organisations to understand their customer's preferences, and the change of their behaviour. With the use of the predictive methods, the companies can provide useful recommendations based on the customer's behaviour history and understand the key driver of the changes in their behaviour.

1.1.3 The 360 Customer View

The 360-customer view (360CV) represents a holistic view of customers and their interaction with the business in a comprehensive manner which can be used as a base for customer behaviour models. The 360-degree data model is a collection of data of all the customer touchpoints from their daily interactions (Indhu 2021).

Creating a customer 360 profile is a challenging task because it needs the data to be collected from multiple touchpoints and then to be joined together and analyzed to get the required insights. With an increase in the number of customer touchpoints since the customer connects with the brand using multiple channels, the generated data contains massive numbers of records and relations between the objects and it is hard to be analyzed (Pines 2018). On the other hand, if a company collects and analyzes a complete context for their customer, they will be able to get valuable customer insights, personalised customer experiences, predict user intention and increase customer loyalty and retention (Indhu 2021). The are many digital-native companies that have employed the 360CV platforms such as Microsoft to create seamless

customer experiences. Microsoft Dynamic 365 Customer Insights[1] unified customer's profile based on their transactional, behavioural, interests, interactions, feedback, and demographic data to understand customer preferences and intent.

1.2 Motivation

A huge amount of data is generated when customers engage with a company's product or service in different touch points, and this data contains valuable information and customer's patterns. With the use of data mining techniques to analyze and classify the customer's data, several benefits can be gained. For example, predictive analysis helps businesses to focus on what they need to improve to gain customer's satisfaction and loyalty. Currently, studying customer behaviour gained the attention of many researchers who proposed different approaches to predict customer behaviour based on the collected data. Most of the previous studies focused on using a source of data to analyze customer behaviour and predict their intention. For example, using clickstream data as in (Koehn, Lessmann, and Schaal 2020) or user-item based data as in (Sheil, Rana, and Reilly 2018) to predict user purchasing intention. These studies proposed different models in their tasks, and complexity. The proposed models could be classification, regression, time series analysis, or recommendation tasks. The complexity of the predictive models varies based on the used machine learning algorithms. For example, Decision Tree algorithm applied by (Manjupriya and Poornima 2018) who proposed a model to predict customer's churn. A real-time purchasing intent prediction model proposed by several researchers with different algorithms, such as a model based Recurrent Neural Network (RNN) by (Guo et al. 2019) , (Peng et al. 2020) applied Reinforcement learning, and (Zheng et al. 2020) proposed a graph convolutional network model.

[1] https://dynamics.microsoft.com/en-us/ai/customer-insights/

While there are several approaches to analyzing customer behaviour in different aspects, we could not find an approach using a combination of different data sources to construct the customer profile. The disadvantage of models relying on one source of data is the limited usage of these models. For instance, If the model is based on clickstream data, the implementation should be only on the exact data source. Another drawback is that the model's performance relies on limited features, and it may need more complicated models to extract patterns in the data.

We assume that modelling the customer profile based on their interactions with different touch points and feeding them to predictive algorithms will provide valuable patterns in customer actions and increase the useability of the model.

1.3 Summary of Research Questions

In this research, we focus our research questions on four points: “360 Customer view concept”, “data structure”, “feature engineering”, and “data-driven multi-task predictive model”.

Modelling Customer profile entities based on 360 Customer view. For the first question, we build an entity relational data model for customers to demonstrate their profile based on their information in several data sources that most businesses may have. Based on the B2C business model demo in Microsoft Dynamic 365 Customer Insights platform[2], the data sources for achieving 360 customer views are demographic, subscription, sales, web pages of the company website, customer service channels, and social media feedback. Also, the implicit features that can be extracted from the transaction records add new information for each customer, such as the customer’s lifetime value (CLV) and purchasing gabs. Accordingly, our first research question is:

[2] https://dynamics.microsoft.com/en-us/ai/customer-insights/

To achieve the model, we illustrate a conceptual model, construct the data entities' sets and constructed a synthetic database based on the modeled entities.

Feature engineering and embedding techniques. The data features must be manipulated before feeding it to the machine learning algorithms to gain valuable and accurate results. Therefore, the second research question is: *what feature engineering and embedding techniques can we apply to the data model features before feeding them to the predictive algorithms?*

Data structure. We need to extract two data structures (relational database and graph database) from the customer profile data model (CPDM) and compare their influences on the performance of the baseline predictive algorithm. The third research question is: *how to extract relational database and graph database from the customer profile data model*?

Multi-task data-driven Predictive model learns the customer profile data. To achieve that, we need to find through exterminations the:

1. the best set of features in the customer profile model for each predictive task individually,
2. the best data structure for the features in CPDM, and how to feed the features to the ML algorithms
3. the best machine learning algorithm based on the performance for each task.

To this end, we have two questions. The fourth research question: *Is feeding all the features in the customer profile model achieve better prediction than one source of data for each task?*

The fifth research question*: How to construct a multi-task predictive model learn the customer features in* the customer profile data model *(CPDM)?*

Based on the answers to the research questions, we propose a data-driven multi-task predictive model using base-line machine learning algorithms which capable of receiving multiple inputs from different sources and predict customer's intentions in the future.

1.4 Objectives

In this thesis, we propose a customer profile data model (CPDM) based on the 360CV concept, and the Customer Behaviour Predictive model (CBPred). The model is a multi-task model, and the customer's intentions to be predicted are churn probability, the customer's next purchase time and next product preferences, as shown in Figure 1.1. We will extract two data structure from CPDM and compare several models. The goal is to find a combination of features from different sources of data and capable of training the ML algorithms to predict different intentions of the customer in one model.

It is not the scope of the research to compare our results with the existing work. From the literature review, we did not find a multi-task model analysing such customer data to predict the focused tasks in our research. In addition, we are applying the baseline Standard Machine Learning algorithms, Feed Forward Neural Network and Graph Neural Networks. There for, it not the research goal to compare the performance of our models with the state-of-art models for the same tasks in the literature.

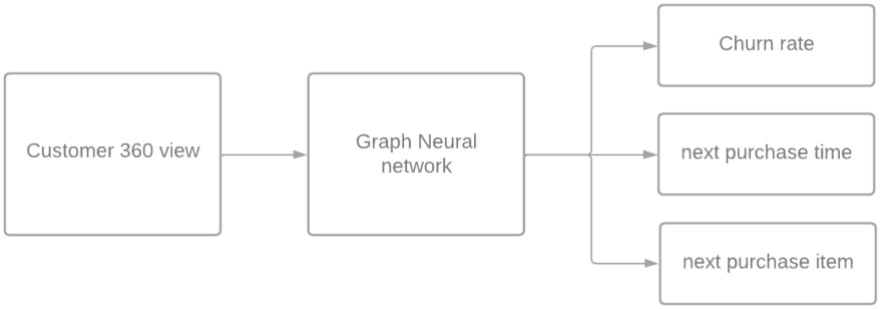

Figure 1.1. CBPred Conceptual Model

To achieve the objectives, our plan is in the following order:

1. Search the literature to learn the previous work which focused on predicting user/customer behaviour.
2. Construct a customer data model – based on 360 customer view - and model the entities with their attributes.
3. Examine several feature engineering and embedding techniques to transform and prepare the customer features before feeding them to the predictive algorithms.
4. Construct the features of the customer profile model in relational and graph database.
5. Apply several experiments to compare the influence of different sets of customers' features - in the CPDM model - on several baseline ML algorithms to predict different customer intentions.
6. Develop a multi-task predictive model based on the best performance of the algorithms in our experimentations and the learned sets of features.

1.5 Contributions

In this research, we focus on modeling customer profile data model and analyze its features to predict the customer's intentions in the future based on their behaviour patterns and similar customers groups. To summarise, the following are the contributions of this paper:

1. Simplify the existing data models based on the 360-customer view to model customer profile data (CPDM) and construct their attributes.

2. Find suitable feature engineering and embedding techniques and applying them on the CPDM's features. The techniques help to reduce the dimensionality and enhance the ML algorithm's performance.

3. Construct the entities sets in the customer view model as relational and heterogeneous graph databases.

4. We examine several sets of features in CPDM to train different algorithms to predict the three focused tasks of customer intentions (Churn probability, next purchase time, and next purchase recommendation). The CPDM's features can train algorithms to solve three customers' intentions with an adequate performance with an F1 score between 70% to 90%.

5. Based on the results of our experimentations, we designed a multi-task Customer Behaviour Predictive Model (CBPred) based on heterogeneous graph and graph neural network. The CBPred predicts the three focused tasks of customer intentions.

1.6 Summary of Evaluation

After extensive experimentation we find that using graph structure is the best fit for our customer profile model. The neural graph networks for predicting customer intentions are the best algorithm with high performance and the least computational cost.

1.7 Structure of the thesis

The rest of this document is organised as follows. Chapter 2 introduces the related work around modelling customers' intentions to observe common behaviours among groups of customers to predict the behaviour of similar customers. Section 3 focuses on the initial experimentation that explores training a purchasing intention model on an open-source dataset. Section 4 presents modelling customer profile data and the constructed datasets based on the modelled entities. In Section 5, we examine three predictive tasks individually, which are trained on different sets of features, and we comprehensively evaluate the performance of different algorithms with different feature sets. Section 6 proposal and discussion on the data-driven predictive model CBPred. Finally, Section 7 concludes the paper and future work.

CHAPTER 2 - LITERATURE REVIEW AND RELATED WORK

In this section we present the relative literature to customer behaviour analysis research and the different proposed approaches to define the gap in this area of research.

2.1 Modeling Customer lifetime

Customer lifetime value (CLV) is "the present value of the future profit stream expected over a given time horizon of transacting with the customer", as stated by (Kotler and Keller 2016). Predictive CLV gives insights about the customer which helps in developing business strategies to retain current customers and obtain new ones ("What Is Customer Lifetime Value (CLV) and How to Measure It?" 2021) (De Marco et al. 2021). In the literature, there are many predictive models proposed to predict the CLV in general and for a specific industry. One of the best ways to predict the CLV is to extract the variables as (RFM) recency, frequency, and monetary variables from the customer's transaction data. Recency is a calculation of when was the last time the customer had a purchase. Frequency is a calculation of how often a customer makes a purchase. Monetary is a calculation of the sum of revenue through the customer lifetime (Segal 2011). RFMS is a model proposed by (Ho, Park, and Zhou 2006) who added the satisfaction variable to the RFM model. Some scholars (Mosaddegh et al. 2021) who targeted the financial industry believe that using the current dynamics of the customers is more suitable for predicting the CLV with the rapid development of the products and technology than relying on the historical customer's behaviour. They selected dynamic (e.g number of transactions, turn over of customer's account) and less dynamic variables (e.g level of education, type of customer, age bracket), and proposed a framework to predict CLV based on the dynamic of customers segmentation over intervals of time. They used the dynamic variables to segment the customers.

2.2 Modeling Purchasing Intention

The task of purchasing intent prediction has been studied widely based on customer's present and historical browsing data, with a variety of models using classic machine learning and deep learning techniques. The previous work comes from (Guo et al. 2019) , proposed for real-time purchasing intent prediction a Deep Intent Prediction Network (DIPN) contains embedding layer, RNN layer, hierarchical attention layer and multi-task layer. They used two types of user interaction behaviour (traditional browse-interactive behaviour and touch interactive behaviour) and fused them using hierarchical attention mechanisms. To predict different purchasing intent, they divided the user purchasing intention into three phases and used multi-task learning to predict the user's unique behaviour. Also, (Li et al. 2020) used the user's real time behaviour sequences (real-time features) which were obtained from edge devices and static historical features to detect the user's purchasing intention by applying a deep neural network called Instantaneous Intent Detection Network (IIDN) which uses (LSTM) and a special attention mechanism to describe temporal dependencies. The user intent was modelled and formulated by (Peng et al. 2020) using a Partially Observable Markov Decision Process (POMDP) as a hidden variable inferred based on the history of the consumer's request sessions. POMDP estimates the probability of "a user visits each hidden state and to which state it may transit". (Wang et al. 2020) proposed a masked-field pre-training framework and Field-Independent Transformer network (FI-Trans). The former to tackle the limited number of labelled data by masking a small amount of the input features randomly, and then using a multitask model to learn to predict the masked features. The later framework predicts user's sparse intents with multilabel classification. FI-Trans contains an embedding layer, field-independent Transformer layers which learns the feature interactions and produces different representations for each feature, and then output layer where with each intent uses an attention mechanism to aggregate relevant features representation to generate the final prediction. The

FI-Trans can handle sequential behaviour if the user behaviours have been represented in different time slots. In (Sheil, Rana, and Reilly 2018), they proposed input representation for automatically engineering the clickstream features to be used to train a model for predicting user purchase intent.

2.3 Modeling Churn Probability

This model for predicting the customer tendency to abandon the business/brand and stop being a customer, and it has been studied in the literature for different sectors. In (Sayed, Abdel-Fattah, and Kholief 2018), they used the customer demographic data and their bank account status. In another study, the customer's choice-specific factors (service quality and price), and individual-specific (demographic and customer's usage characteristics) factors were considered to influence the customer's decision to churn or not to churn (Kim and Yoon 2004). In (Karnstedt et al. 2011), they studied social media user churn by analyzing the communication interaction between users over online forums, such as, the number of posts and replies and the importance of the user in the social graph. The user churn in the video game industry was studied in (Bonometti et al. 2019), and they defined the leaving user when they were not completing the game and being inactive for a period of time. For each user record, they measured the total minutes of game sessions, the active play time, the gap between user's sessions, number of initiated games, and information of the user's preferred game. The churn study in (Tamaddoni, Stakhovych, and Ewing 2016) was conducted on the customer's transactions, where the recency and frequency variables were chosen as predictors.

2.4 Modeling Next Purchase Time

It is a predictive analysis task to know if a customer is likely to make a purchase in the next few time periods based on the frequencies in their purchasing history. In (Huang et al. 2019), they analyzed the temporal patterns in the customer transactions history and the categories of the purchased items to predict the next purchase time using graph neural network.

In a different study, customer's demographic data, transactions history and product characteristics were learned to predict if the customer will make a purchase next week (Zhai et al. 2020). Predicting next day online purchase was studied by (Liu et al. 2021) based on the customers online shopping records, naming, past purchases, click, collect, add-to-cart and payment. Sequences of days between orders and gaps between prior orders for each product were analyzed to predict the next gap between the last purchase and the next purchase (Droomer and Bekker 2020).

2.5 Modeling Next Item Recommendation

The task is to recommend a number of products that the customer is likely to purchase. One of the most popular recommendation systems in literature is the collaborative filtering system. The system recommends products to the customers based on their preferences for items, as in (Yang et al. 2017), who studied improving the user-item rating system by extracting information about the user through their connections on social media. The AutoRec system enhances the user rating features through autoencoder implementation (Sedhain et al. 2015). A study conducted by (Zhu et al. 2021) to debiasing the effect of the item popularity on the recommending items. The sequential recommendation systems are based on analyzing a sequence of items of each customer ordered based on timestamps to predict the list of items for recommendation as in (Jiaxi Tang and Wang 2018) who proposed an approach to deal with sequence of items and their embedding as image data using horizontal and vertical filters. While (Tan et al. 2021) developed a new system based on representing each user by multiple embedding to encode different aspects of their intentions.

2.6 Graph Representation Learning

Graphs are non-linear data structures, and each graph is defined by a set of nodes and a set of edges between these nodes. The nodes and the edges can have their features and labels. The graph can be a homogeneous graph with one type of nodes and edges, or a heterogeneous

graph with multiple node types and edges (W. L. Hamilton 2020). Graph representation is the graph features embeddings which are the encoded features for the models to use in different tasks. The task of intent prediction has been achieved through graph representation learning as in (Pengyang Wang et al. 2019) who studied mobile users profiling with point of interest (POI) check-in data to predict the next POI. Their framework consisted of a deep autoencoder, an approximated sub-structure detector, a discriminator, and an adversarial trainer. They modelled user activity graphs to extract the characteristics, patterns, and preferences of mobile users. Traditional subgraph detection algorithms were applied to generate substructures to feature the unique behavioural patterns of a user's activities which approximated using pre-trained CNN. The discriminator was to classify the substructures, while the adversarial trainer role was to preserve substructures in the reconstructed graph and confuse the discriminator. From the learned user representation, the model predicts mobile users' next activity. (X. Wang et al. 2019) proposed Knowledge Graph Attention Network (KGAT) to model the high-order connectivity in collaborative knowledge graphs. KGAT framework consists of embedding layer to vectorize each node and preserve the collaborative graph structure, attentive embedding propagation layer which propagates the embeddings from a node's neighbours (users, time, or attributes) to generate the node's representation with an attention mechanism to capture the importance of the neighbours through their weights, and the prediction layer which use user-item representations to predict matching score. (Pengyang Wang et al. 2020) proposed a platform based on reinforcement learning and a spatial knowledge graph given temporal context for learning the user's patterns, preferences, and spatial entities to predict the user's next visit. In the spatial KG, three types of entities: the user POI, the POI's category, and the location distance from a functional zone. The platform solves incremental Mobile User Profiling as when the user takes an action to visit the next POI, a change in the environment will occur causing a new state of the user and the spatial entities which helps the agent (visit

planner) to accurately predict the next visit. In (Zheng et al. 2020), they proposed an approach to predict the user's purchasing intention based on the item price and taking into consideration the influence of the product category on the user's willingness to pay. They have modelled the user interaction after discretizing the price - to 10 price level categories - as a unified heterogeneous graph with four types of entities: the user, the item, the price, and the category. To learn semantic representations and to estimate interaction probability, a graph convolutional network and a pairwise interaction-based were employed as an encoder and a decoder, respectively. The framework has two branches: global branch and category branch. The former for predicting the overall user's purchasing power with the nodes of user, item, and price. While the latter predicts the user's price awareness related to the item category with the nodes of user, category, and price. In (Fan et al. 2020), they modelled a heterogeneous information network (AHIN) to describe rich semantics and complex relations between different types of entities in Hack Forums (buyer, vendor, product, comment, topic, and a meta graph scheme). In order to learn node representation, they proposed a meta graph aggregated heterogeneous graph neural network (mHGNN) which consists of meta graph-guided neighbour search, attentive propagation and aggregation, and multi-view fusion. After fusing and then concatenating the buyer and product embeddings, they have been fed to MLP layers.

2.7 Existing Datasets

Most of the proposed models to detect customer intentions were evaluated in the literature on benchmark and public datasets. YOOCHOOSE[3] dataset and DIGINETICA[4] dataset for evaluating session-based recommendation models (Li et al., 2017), and it consists of the customers' click-stream on an e-commerce site. In YOOCHOOSE, each record consists of session id, date, clicked items, category, and price. DIGINETICA dataset contains browsing

3 https://recsys.acm.org/recsys15/challenge/

4 https://competitions.codalab.org/competitions/11161

logs, product data, transactions, and product images. In (Peng et al., 2020), proposed a model to understand users' purchasing intention based on the sequential advertising strategy, and they used Taobao sequential advertising data[5] to evaluate their model. The dataset includes request session ids, session time, user ids, ad groups and ad items. A Recurrent Neural Network model was proposed by (Sheil, Rana, & Reilly, 2018) to predict purchasing intent, and they evaluated the model using Retail Rocket Kaggle[6] which consists of events (click, cart, and transactions), items properties and category tree. A public data on Kaggle[7] for a telecom company containing various customer behaviour (calls, messages, plans, and charges) was used to evaluate the Customer Churn Prediction model (Kumar Thakkar, Desai, Ghosh, Singh, & Sharma, 2022). Public data from JD.com consists of the user and product characteristics and interactions data, and (Zhai, Shi, Xu, Wang, & Chen, 2020) used it to evaluate their model for predicting the next purchase category. Alibaba Mobile Recommendation dataset[8] was used to evaluate the model in (Liu et al., 2021) for predicting who will purchase the next day. The data includes customers' actions on the e-commerce website (clicks, add-to-cart, and payment). Prediction, the next purchase date model, was proposed by (Droomer & Bekker, 2020), and they used it to evaluate an online grocery shopping dataset[9] consisting of tables for orders, departments, products, aisles and products. For the recommendation system (Yang, Lei, Liu, & Li, 2017), the model evaluation was based on three datasets epinions.com, douban.com and flixster.com. All the datasets are extracted from social networking services and contain ratings, reviews and social relations. For graph datasets, user-item review graphs such as Movies Review and CDs reviews datasets were used to evaluate the heterogeneous graph-based model to predict customer preferences proposed by (X. Wang et al., 2019).

5 https://github.com/465935564/sequential_advertising_data

6 https://www.kaggle.com/datasets/retailrocket/ecommerce-dataset

7 https://www.kaggle.com/datasets/anish9167473766/churndata

8 https://tianchi.aliyun.com/dataset/dataDetail?dataId=46&lang=en-us

9 https://data.world/datasets/instacart

2.8 Comparative Analysis and Gap Analysis

We conclude from the previous work that we reviewed that each approach focuses on a set of features to analyze the customer behaviour to learn their patterns, and develop a predictive model based on machine learning algorithms for different tasks. As in Table (2.1), the purchasing intention approaches were focused on the browsing behaviour and clickstream datasets. While the proposed models for churn probability task focused on the customer's demographic information, and their satisfaction in different aspects. The purchasing actions by the customers are the key features in predicting next purchase tasks. In addition to the classic machine learning approaches and neural networks models, the graph models opened several opportunities to improve predicting user's behaviour tasks.

After exploring the proposed approaches, we find a lack of examination to learn from different feature sources in one model to analyze customer behaviour and predict their following actions. In addition, while there are several approaches to using heterogeneous graphs for customer behaviour analysis, the proposed models are based on the user and the item entities, which belong to transaction information.

By reviewing the existing datasets which used to analyze customer behaviour in the previous work, we found that all the datasets contain features from one or two sources of data, such as sales, clickstream, or social media. We could not find an open-source dataset represents a 360-customer view.

Table 2.1. Comparative Analysis

Research	Data type	Approach	Task
(Guo et al. 2019)	Browse and touch interactive behaviour	RNN and hierarchical attention	purchasing intent prediction
(Li et al. 2020)	user-coupon features	Long Short Term Memory (LSTM) and a special attention	purchasing intent prediction

		mechanism	
(Peng et al. 2020)	Advertisements sets	Partially Observable Markov Decision Process	purchasing intent prediction
(Wang et al. 2020)	click through logs.	Masked-field pre-training framework, and a Field-Independent Transformer	purchasing intent prediction
(Sheil et al. 2018)	Ecommerce clickstream data	Long-short time memory	purchasing intent prediction
(Sayed et al. 2018)	Customer demographic data and their bank account info.	Decision tree algorithm	churn probability
(Kim and Yoon 2004)	Satisfaction variables based on the service quality and customer demographics data.	logistic regression	churn probability
(Karnstedt et al. 2011)	centrality of users in social graph	Information Gain Ratio	churn probability
(Bonometti et al. 2019)	user ‘s game sessions features	Long-short time memory	churn probability
(Tamaddoni et al. 2016)	recency and frequency variables	SVMs	churn probability
(Huang et al. 2019)	customer transactions history and the categories of the purchased items	Graph convolutional network	Next Purchase Time
(Zhai et al. 2020)	customer’s demographic data, transactions history and product characteristics	Light GBM & XGBoost	Next Purchase Time
(Yang et al. 2017)	Customers, their purchased items, and ratings	TrustPMF model based on matrix factorization	Next Item Recommendation
(Sedhain et al. 2015)	Customers, their purchased items, and ratings	Collaborative filtering	Next Item Recommendation
(Zhu et al. 2021)	Customers, their purchased items	sequential recommendation	Next Item Recommendation
(Pengyang Wang et al. 2019	point of interest	spatial knowledge graph	Next point of interest

Zheng et al. 2020	Item price and category	graph convolutional network	user's purchasing intention

CHAPTER 3 - PROBLEM STATEMENT & PROPOSED DATA MODEL

3.1 Problem Statement

Most of the previous studies use one source of data for analyzing customer behaviour, such as clickstream data and customer's transaction data. We assume that we can model customer profile data model based on 360-customer view to analysis customer behaviour and predict their future intents. In order to achieve that, we need to aggregate the customer's demographic information, the customer's transaction data, customer's browsing data, customer's feedback through different channels (social media and customer service) and the product's characteristics. With relational data, each of these entities is stored in a different table and we need to create another mapping table to reconstitute the data back end. To analyze customer behaviour using different entities, it is important to understand the relation between them, and we believe that linking these entities in one graph is a solution.

Graph data is a representation of discrete entities and describes the relation between them which can be used to represent the relation between different entities in the CPDM and answer many questions about their intentions. In order to understand and summarise the customer actions, we need to ask questions to understand their behaviour, such as what are their preferences, what patterns they have, what is the turning point in their behaviour, what is the user's spending margins, how many purchasing the customer make in a certain period of time, what influences the customer to purchase, and could the customer be influenced by their location?

In this research, we will examine if the graph-based model can be considered a solution to analyze customer profile entities and their complicated relations. Therefore, we will model the entities in CPDM and construct them into two types of datasets: relational and a graph data; then, compare the results after applying multiple queries to investigate the customer's

behaviour on these datasets to predict the customer's intentions. The proposed approach will be constructed based on the comparison results to predict the main investigated actions in this research: 1- churn probabilities, 2- next purchase time, and 3- the next products to be purchased. Churn rate is an important metric to estimate a company's growth. Companies try to reduce the churn rate because acquiring new customers is more expensive than retaining existing ones. Usually, the revenue from the latter is higher than the former (Ahmad et al. 2019). Predicting the customer's next purchase (when and what) helps to understand their purchasing patterns and target them with relative campaigns and promotions (Liu et al. 2021).

We assume that customer churn rate prediction can be influenced by different factors, such as the customer's feedback, customer loyalty, purchasing history, including CLV segmentation, and the customer's demographic. The next purchase prediction can be influenced by the Customer's online activities, customer's demographic, and purchasing history.

To achieve the proposed model, we firstly explore several models through experimentations:

1. Initial experimentation for exploring a purchasing intention model based on different ML algorithms and training it on a public dataset consists of customer behaviour from one data source. This will give us a clear view of how this type of models behave and how the data was modelled and fed to the algorithms.
2. Develop a series of experiments to achieve each predictive task using different machine learning models and learn different combinations of features in CPDM to understand what influences customer intentions. These experimentations aim to explore the best algorithms and sets of features for analyzing customer profiles.

3.2 Customer Profile Model

In this section, we illustrate conceptual data model based on several touch points. Then, we model the relational entities in CPDM model and extract tabular and graph datasets.

3.2.1 Conceptual Model

Based on B2C business model demo in Microsoft Dynamic 365 Customer Insights platform, we constructed the conceptual model as in Figure 3.1 from several touch points that the customer interacts with before, or during, or after the purchase. The first object "Demographic" demonstrates any personal information about the customer, such as age. The second object is the customer's subscription information which is created once the customer registers themselves or buys a membership. The third object demonstrates any transactions between the customer and business. Customer's satisfaction is represented by the fourth object: Feedback. The fifth object is the customer's online interactions which may include clicked advertisements, page views, received emails and any other online activities.

Figure 3.1. Conceptual Data Model for 360 Customer View

3.2.2 Customer's Data Entities Relationships Model

Based on the five objects in the conceptual model, we construct the Entity Relationship Model (ERM) (Figure 3.2) for CPDM which depicts relationships among the customer, and the

business channels, naming; membership, products, webpages, business twitter account, promotional emails, customer service, and surveys.

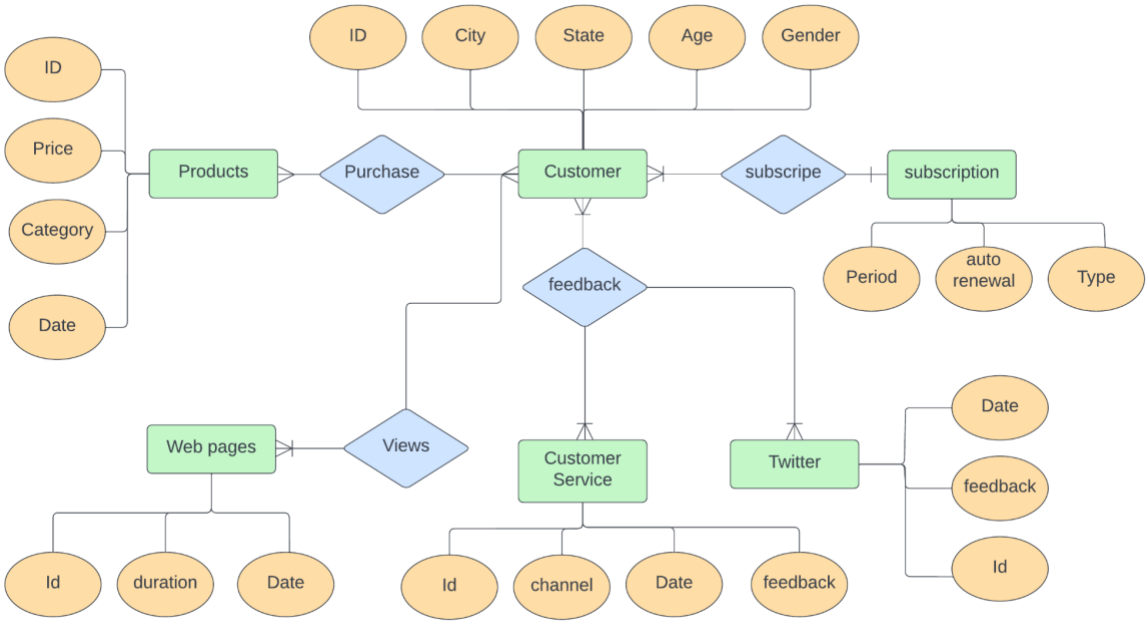

Figure 3.2. The Entity Relationship Model of Customer profile

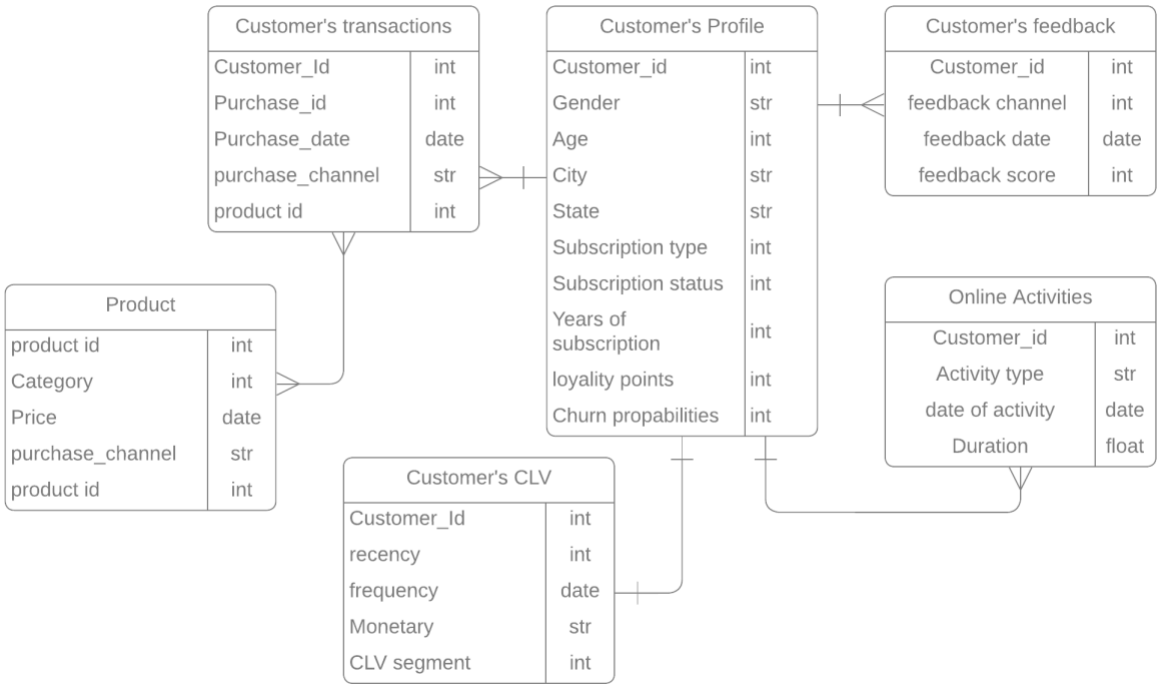

Figure 3.3. Entity Relation Diagram

In Figure 3.3, the entity relationship diagram is an implementation of our ERM with the main entities and their attributes. We constructed the entitles attributes with influence of Microsoft Dynamic 365 Customer Insights B2C demo. We eliminated the attributes that add noise to the data and distract the ML algorithms after several experimentations.

The customer profile entity combines information from two objects, the demographics, and subscription information. Customer's purchases entity is based on any purchase order in the customer lifetime with the business. The online activities entity combines web pages views. The feedback is the reviews the business receives from the customer to measure their satisfaction through different channels, such as customer's cases with customer service department, and customer's posts on social media. In addition to the explicit information in the customer entity, Customer Lifetime value (CLV) has been assigned for each customer by aggregating customer's recency, frequency, and monetary values (RFM) based on the customer transactions records.

The following are the key characteristics of each entity:

Customer's transactions: incorporates customer's purchases. The variables in this entity are customer id, purchase date, purchase id, and product id.

Customer's feedback: incorporates customer's tweets sentiments, and customer service feedback. Each customer may have several records. The variables in this entity are customer id, feedback id, feedback channel (tweet, survey, email, call, or chat), feedback score and date of the feedback.

Customer's online activities incorporates customer's page views on the business website and each customer may have several records. The variables of page views entity are customer id, page id, page view date, and view duration.

The customer profile incorporates customer's demographic data, and customer's membership information, and each customer has one record. The variables in this entity are customer Id, age, gender, city, state, subscription type (gold, silver, and bronze), subscription state (automatic renewal or not), number of years of subscription, loyalty points, and churn probabilities (initiated randomly).

Customer life value incorporates customer's RFM based on their transaction history, and each customer has one record. The variables in this entity are recency, frequency, monetary, and CLV segment.

Products consists of information about each product business sells. The variables in this entity are product id, category, and price.

3.2.3 CPDM Physical Database

In this section, description of the constructed tables to create CPDM database. Based on the relational diagram in Figure 3.3, the existing benchmark and public datasets do not conclude the records that covers all the entities. The 360 customer view data is available in e-commerce platforms, and there are many restrictions for using it, such as customer privacy and business copyrights. Thus, to examine the modelled entities, we randomly synthesised entities' values based on the ERD of CPDM.

Table 3.1. Notations

Notation	Description	Notation	Description
$prof$	Customer profile data record	U, u	Customers set, customer
$purch$	Customer's purchase record	P, p	Products set, product
tw	Customer's tweet record	$categ$	Product category
$case$	Customer service case	$pric$	Product price

In the model, there are six tables containing customer information and their history behaviour: 1- customer's profile, 2- customer's purchases, 3- customer's CLV, 4- customer's page views, 5- customer service cases, and 6- customer's tweets sentiments. The tables' instance notations are in Table 3.1.

- **Customer Profile dataset:** Based on Customer's profile entity, we constructed a dataset with (10,000 records), which consists of 10,000 unique customers. Each record is an individual information of a customer which consists of demographic, and subscription attributes, specifically, $prof = (id_i, a_i, g_i, ci_i, st_i, mt_i, my_i, mr_i, ml_i, sp_i, chrn_i)$ containing the following:
    a. Customer ID id_i *(int),*
    b. Customer age a_i *(continuous)*,
    c. Gender g_i (*male or female, nominal*),
    d. City ci_r (*298 cities, nominal*),
    e. State st_r (*47 states, nominal*),
    f. Subscription types mt_r (*3 types, ordinal*),
    g. Subscription in years my_i *(continuous)*,
    h. Auto renewal subscription mr_i (*binary*),
    i. Loyalty points ml_i (*int*),
    j. Customer's churn probability $chrn_i$ (*binary*).

- **Customer Transactions dataset:** based on the Customer's Transactions entity, and each customer may have several records. The transaction dataset consists of information about the customer purchases for the last nine month, and it was designed to contain customers with more than five transaction records as a minimum purchasing history. Each record is a transaction $purch$ represents a customer purchasing a product,

and a rating feature contains (1) for all the records. $purch = (id, prod, cat, pric, tsmp, rate)$ which contains the following:

a. Customer ID id_i *(int)*
b. Product ID $prod_i$ *(int)*
c. Times tamps $tsmp_i$ *(continuous)*
d. Ratings $rate_i$ *(binary)*

- **Customer Lifetime Value (CLV) dataset:** it is constructed based on the records of the first six month in the transactions datasets by calculating the RFM values and clustering the customers based on these values. Each record $clv = (id_i, f_i, r_i, m_i, sp_i, clv_i)$, containing:

  a. Customer ID id_{clv} *(int)*
  b. Frequency f_i the gap in days since the last purchase $(a - b)$ as a is the last date of the first six month and b is the last purchase date (*continuous)*
  c. Recency r_i the last purchase date b,
  d. Monterey $sp_{i\ i}$ the total value of the transactions for each customer as in Eq. 3.1,
  e. The customer's total number pf transactions m_i as Eq. 3.2,
  f. CLV segment clv_i (*three segments*).

$$sp_i = \sum_{i=1}^{n} price_p \ for\ each\ u\ where\ u \in U\ and\ p \in P \quad \textbf{Eq. 3.1}$$

$$\sum_{i=1}^{n} purch_u \ where\ u \in U \quad \textbf{Eq. 3.2}$$

- **Customer Page Views dataset:** it is based on online activities entities, and each customer may have several records. Each record pv consists of the customer's page views of the business website pages, specifically, $pv = (id_{pv}, pg_{pv}, ts_{pv}, dur_{pv})$ containing:

  a. Customer ID id_{pv} *(int)*
  b. Page view ID pg_{pv} *(int)*

c. Date of page view ts_{pv}

d. Duration of the page view dur_{pv} *(continuous)*

- **The customers cases dataset:** it is based on the Feedback entities, and each customer may have several records. The dataset is constructed to contain customer feedback records through different channels (call, email, and chat), and the feedback in each channel was scaled to be between 1 to 10. Each record is a customer case, $case = (id_i, cid_i, ts_i, ch_i, fb_i)$ containing:
  a. Customer ID id *(int)*,
  b. Case id cid_i *(int)*,
  c. Timestamp ts_i,
  d. Case channel ch_i *(call, email, or chat)*
  e. Customer's feedback fb_i *(continuous)*
- **The customer's tweets dataset:** it is developed based on the Feedback entities, and each customer may have several records. The dataset is constructed to contain customers' feedback on Twitter based on sentiment analysis of the customer's tweets which we scaled to be between 1 to 10. Each record is a customer tweet $twt = (id_i, tid_i, ts_i, fb_i)$ containing:
  a. Customer ID id_i *(int)*
  b. Tweet id tid_i *(int)*
  c. Timestamp ts_i,
  d. Customer's sentiment (continuous)
- **The products dataset:** Contains all the features for each product in the business. Each record is a set of product features $p = (id_i, categ_i, pric_i)$ containing:
  a. Product ID id_i *(int)*
  b. product category $categ_i$ *(int)*
  c. Product price $pric$ *(continuous)*

After constructing all the datasets, we updated the customer churn probability in the customer profile table - which was developed randomly - by applying K-means clustering technique from Sklearn library ("Sklearn.Cluster.KMeans" n.d.) to divide the customers into two groups based on their last purchase gap f_i and the total spending sp_i in the CLV dataset. The K-means algorithm grouped the customers with a long gap and with below average of total spent in one group (0), and the customers with a short gap and above average of total spent in the second group (1) as in Eq. 3.3.

$$chrn_i\ (0|1)\ = (f_i\ \&\ sp_i > average\ f_i\ and\ average\ m_i\ ) \qquad \textbf{Eq. 3.3}$$

3.3 Discussion

In this chapter, we answered the first research question:

Q1: What are the main entities for our customer profile model based on the 360 Customer View concept? Furthermore, how to model each entity?

> We described how to construct a customer profile model based on the 360-customer view model of the Microsoft 360 Customer Insights platform. We started with the conceptual model, which includes the customer's information from different data sources with a business. This includes the demographics data, feedback channels, subscription information, transaction data, and online activity, which focused on the web page views in our model CPDM. We simplified the features for each entity with a focus on the features that describe customer behaviour, preferences, and satisfaction.

The demographic data influences the customer decisions, such as, their age, gender and location. The feedback from the customer measures their satisfaction and their tendency to leave or remain in the business. Analysing customers' transactions informs the business patterns in customers' purchasing behaviour, including the time and the type of product they consume.

The loyalty of the customer towards the brand or a business reveals in the number of transactions, the total number of transactions and their subscription status. Finally, the duration of page views represents the customer's time spent on the business website which explains their tendency to purchase and their loyalty to the business.

CHAPTER 4 - INITIAL EXPERIMENTATION

The experiment's goal is to explore the performance of machine learning algorithms when learning customer behaviour from one data source modelled as tabular and graph datasets.

The input to Purchasing Intention (PI) system is a sequence of items the customer clicked during a session, and the output is a label for each sequence of inputs. The label is binary: 1 for positive intention or 0 for negative intention.

From a machine learning perspective, this problem is a sequence classification that predicts a class label for a given input sequence. A labelled dataset of positive and negative purchasing actions for each session is necessary to train a classifier. After training the model, it can be deployed to predict PI on unseen data from the system.

Using base-line neural network models, we will compare influence rational and graph datasets on the algorithms that learn customer behaviour from the session-based data model to predict the customer's purchasing intention (PI). We will train different neural network algorithms on the experimented dataset and compare the evaluation results. Each algorithm will learn the data with different label distributions: balanced, unbalanced and oversampling.

4.1 Dataset

This section contains a summary of the used datasets, and the Key characteristics of the dataset are summarised in Table 4.1.

The experiment was done on the RecSys Challenge 2015 dataset (YOOCHOOSE), containing two related tables: "click" and "buy" tables. The click table contains four features recording information about the clicked items during the sessions, and the buy table contains five features about the purchased item during each session.

Table 4.1. Summary of RecSys Challenge 2015 dataset characteristics

Dataset	No of instances	N. of features	Features
RecSys Challenge 2015 click table	33003945	3 categorical, and 1 DateTime.	Session_id, timestamps, item_id, category
RecSys Challenge 2015 buy table	1150753	5 (2 categorical, 2 continuous and 1 DateTime)	Session_id, timestamps, item_id, price and quantity.

4.2 Evaluation metrics

Several metrics and tools measure the performance of machine learning models based on binary classification tasks. In the experiment, we will rely on the confusion matrix, which calculates the following measures (Sokolova, Japkowicz, and Szpakowicz 2006):

True Positive Rate (TPR) or Recall or Sensitivity: it calculates the ratio of true positives (TP) to the sum of true positives (TP) and false negatives (FN) - TP / (TP + FN) - which measures the percentage of the positive class got correctly classified. Since it does not contain false positives (FP) and true negatives (TN), it leads to biased results in an unbalanced dataset.

Accuracy: it calculates the ratio of true predicted labels to the sum of all the predicted labels - (TP + TN) / (TP + TN + FP + FN). While it contains information about TP and TN, It is not an accurate metric with the unbalanced data.

Precision: calculates the ratio of TP to the sum TP and FP = TP / (TP + FP)

F-measure It is the harmonic mean of precision and recall (2 * (precision * recall) / (precision + recall)) ("Sklearn.Metrics. F1_score" n.d.).

AUC (Area Under the Curve): represents the connection between sensitivity and specificity, which measures the algorithm's ability to distinguish between positive and negative classes. If AUC is greater than 0.5, then there is a chance for the model to differentiate between positive and negative classes. When the AUC is 0.5, the model cannot distinguish between the positive

and negative classes and predicts random or constant class for all the observations in the data (Bhandari 2020).

Precision-Recall Curve (PRC): is a helpful metric that measures the prediction performance when the labels are imbalanced. PRC "shows the trade-off between precision and recall for different thresholds". The average Precision (AP) metric can measure the area under the curve.

4.3 Predictive Models

We showcase the effectiveness of different base-line Neural Network models to predict purchasing intention based on session-based dataset. The following models were selected for the empirical comparison:

Feed forward neural network (FNN): A neural network consists of one or more layers of neurons. There are three basic steps in the model: input, hidden layers, and output (Abirami and Chitra 2020). The output dimension of the last layer is equal to the number of classes to predict.

Convolutional Neural Network (CNN 1D): it is capable of learning how to extract features from a sequence of steps, and how to map the internal features to different tasks (Hiriyannaiah et al. 2020).

Simple Recurrent Neural Network (Simple RNN): is different from traditional feedforward neural networks, RNN has feedback connections that allows the RNN to model the impact of the early information of the sequence on the next part of the sequence (Hiriyannaiah et al. 2020).

Long-Short Term Memory (LSTM): is an RNN model with cell states and hidden states.

Graph Neural network (GNN): It is a neural networks learn the nodes and graph representations by aggregating the neighbourhood information and update the neighbours

representations (Morris et al., 2018). In our experimentation, the model is a graph-based classifier which learns each graph embedding to predict the embeddings for unseen graphs. (W. Hamilton, Ying, and Leskovec 2017). We will apply the vanilla model GraphConv and the GraphSAGE mode. GraphSAGE is an inductive framework capable of predicting the representations of a new node through leveraging node attribute information (Hamilton, Ying, & Leskovec, n.d.).

ANN, RNN, CNN and LSTM are trained on sequence datasets, while GNN models are trained on a graph dataset.

4.4 Data Preparation

The dataset does not have null values and it did not need data cleaning. In this section, we describe the pre-processing steps we performed on the datasets.

Add session label to *click* data: We compared the two datasets (click and buy) to add a label feature to the click table. If the session id is in both datasets, then the session has ended with a purchase and takes positive label (1), if not, takes negative label (0). The final *click* dataset features are session-id, timestamps, clicked item, category, and purchasing label.

Sequence dataset: The *click* data has several sessions, and a session may appear in several records with the clicked items during this session. To construct a sequence dataset, each session should represent a record with a sequence of items. Firstly, the data were filtered to retain the sessions with at least three (records) clicked items. Then, the maximum number of items in the sessions is used as the fixed number of steps in the sequence dataset as in code 4.1. The sessions with a smaller number of items were padded with zero values, and each session has its purchasing label (0 or 1) as in Table 4.2.

Code 4.1: How model sequence data

```
T=198 #maximum number of items in one session
X=[]
Y=[]

grouped = df.groupby('session_id')

for _, group in tqdm(grouped):
 t = len(group.item_id)
 x= group.item_id[0:t]
 z= np.zeros(T-t)
 x=np.hstack((x,z))
 X.append(x)
 y= group.label[0:t]
 Y.append(y.max())

X=np.array(X)
Y=np.array(Y)
```

Table 4.2. An example of the records in the data fed to neural network models

Session id	step1	step 2	step 3	step 4	step 5	...	step198	Purchasing label
3	21302	25022	29039	0	0	...	0	0
19	5110	5110	8889	8889	0	...	0	0
24	8976	8976	1946	1946	0	...	0	0
54	22760	19135	19135	19135	19135	...	19117	0
63	7354	25643	27920	0	0	...	0	0

Preparing datasets with different label distributions: the labelled sequence dataset was imbalanced. To tackle the problem of the unbalanced classes, we apply two techniques: 1- extract balanced samples, 2- Oversampling the minority class. Then, we apply the predictive models on the unbalanced dataset, the balanced samples and on the oversampled set, and compare the results.

The distribution of the three datasets is as the following:

- The unbalanced dataset: contains 500,000 sessions with 457494 negative labels and 42506 sessions with positive labels.

- The balanced sample: contains 114660 session-id with an equal number of sessions with negative and positive labels.
- The unbalanced data with oversampling technique: The SMOTE class ("SMOTE — Version 0.8.0" n.d.) was applied on the unbalanced dataset to oversample the minority class by creating synthetic examples to increase the ratio of the minority class to the majority class to 40%, and we followed that by an under sampling to the majority class as recommended in SMOTE paper for better classification results (Chawla et al. 2002) to increase the ratio of the minority class to 0.85%. The result of this technique is data containing 430657 negative labels and 366059 with positive labels.

Preparing dataset for the Graph-based model: we construct a homogeneous graph for each session consisting of nodes of the clicked items. The features of the nodes are the item id, and the item's category (Kung-Hsiang, (Steeve), and Kung-Hsiang 2019). Each graph is labelled with the session label. After constructing the dataset, we shuffled and split the data into three sections for training, validation, and training.

4.5 Hyperparameter Tuning

To find the best set of hyper-parameters configurations for each model, we applied a random search of the parameter space using KerasTuner - the scalable hyperparameter optimization framework (Keras-Tuner: Hyperparameter Tuning for Humans n.d.) from the Keras library. Table 4.3 shows the best hyper-parameters with the highest AUC score for each model on the datasets with three different distributions.

Table 4.3. The best hyper-parameters

Models	Unbalanced dataset	Balanced dataset	Oversampled dataset
ANN			
no. of layers	4	4	5
no. of units in layers	192, 96, 480, 192	320, 288, 32, 32	320, 512, 32,32,32
layer activations	ReLU	ReLU	ReLU
output layer activation	Sigmoid	Sigmoid	Sigmoid
loss function	Binary Cross Entropy	Binary Cross Entropy	Binary Cross Entropy
kernel_initializer	Glorot_uniform	Glorot_uniform	Glorot_uniform
layer dropout rates	0.2	0.1	0.0
batch size	32	32	32
optimizer	Adam	Adam	Adam
Learning rate	0.0001	0.0001	0.0001
epochs			
CNN			
no. of blocks	3	4	5
filters	224, 64, 64	224,128,160,32	128, 192, 64, 32
layer activations	ReLU	ReLU	ReLU
Pooling layers	2/Max, avg, avg, avg	2/ Max, Max, Avg, Avg	Avg, avg, max, avg, avg
output layer activation	Sigmoid	Sigmoid	Sigmoid
loss function	Binary Cross Entropy	Binary Cross Entropy	Binary Cross Entropy
kernel_initializer	Glorot_uniform	Glorot_uniform	Glorot_uniform
layer dropout rates	0.4	0.1	0.0
batch size=	32	32	32
optimizer	Adam	Adam	Adam
Learning rate	0.00075	0.0005	0.0006
RNN			
no. of layer	3	3	2
no. of units in layers	300, 300, 200	150, 150, 1502	200, 250
layer activations	Tanh	Tanh	Tanh
output layer activation	Sigmoid	Sigmoid	Sigmoid
loss function	Binary Cross Entropy	Binary Cross Entropy	Binary Cross Entropy
kernel_initializer	Glorot_uniform	Glorot_uniform	Glorot_uniform
layer dropout rates	0.2, 0.1, 0.4	0.1, 0.2, 0.3	0.3, 0.2
batch size	32	32	32
optimizer	Adam	Adam	Adam
Learning rate	0.00015	0.0002	0.0002
LSTM			
no. of layers	3	4	2
no. of nodes in layers	1150, 150, 100	100, 100, 50, 50	100, 100
layer activations	Tanh	Tanh	Tanh
output layer activation	Sigmoid	Sigmoid	Sigmoid
loss function	Binary Cross Entropy	Binary Cross Entropy	Binary Cross Entropy
kernel_initializer	Glorot_uniform	Glorot_uniform	Glorot_uniform
layer dropout rates	0.3, 0.4, 0.0	0, 0.1,0.3, 0.4	0.1, 0.2
batch size	32	32	32
optimizer	Adam	Adam	Adam
Learning rate	0.0003	0.0006	0.0006

GraphSage			
no. of layers	3	3	3
layer activations	ReLU	ReLU	ReLU
output layer activation	Sigmoid	Sigmoid	Sigmoid
loss function	Binary Cross Entropy	Binary Cross Entropy	Binary Cross Entropy
layer dropout rates	0.2	0.2	0.2
batch size	512	512	512
Optimizer	Adam	Adam	Adam
Learning rate	0.005	0.005	0.005
GraphConv			
no. of layers	3	3	3
layer activations	ReLU	ReLU	ReLU
output layer activation	Sigmoid	Sigmoid	Sigmoid
loss function	Binary Cross Entropy	Binary Cross Entropy	Binary Cross Entropy
layer dropout rates	0.5	0.5	0.5
batch size	200	200	200
optimizer	Adam	Adam	Adam
Learning rate	0.001	0.001	0.001

4.6 Results and analysis

In this section, we compare the effectiveness of different deep learning models after they were trained on the session-based dataset.

4.6.1 Models' Performance on Different Label Distribution:

The effectiveness of the models based on the metrics: Accuracy, Precision, Recall, F1-score, PRC and AUC are illustrated in Table 4.4.

Unbalanced data: From the results in Table 4.4, the AUC metric for all the models achieved a low value of around 0.65. The highest value of 0.68 was for the ANN model.

The low performance for most of the models is related to the extreme imbalanced labels in the data. Since the positive label is the minority class of the examined data in this section, these models did not predict positive labels, and all the predicted labels were negative which are relevant to the high accuracy. Therefore, these models are not capable of distinguishing between the positive and negative observations, and they labelled all the sessions with a negative label.

Balanced sample data model: From the results in Table 4, we conclude that all the models achieved better on the balanced distribution of the dataset. The best AUC results were 0.82 for the ANN and CNN models and followed by GraphSAGE with 0.81. The GraphConv model achieved the lowest performance with AUC of 0.70.

We also observed on most of the models that the recall metric is between 0.55 and 0.65 with ANN, CNN, and RNNs, which is a result of the low number of predicted true positive labels and a high number of false negatives predictions. On the other hand, the recall metric for GraphSAGE model recorded high percentage of 80%.

Unbalanced data with oversampling techniques: We observed lower performance for all the models compared with the balanced distribution. The highest performance was for GraphSAGE, ANN and CNN with an AUC of 0.71, while RNN and LSTM achieved 0.68 and 0.69 respectively. GraphConv achieved a low AUC of 0.5.

From the recall and precision metrics, we conclude that the number of the predicted false positives is not as much as the false negative labels were predicted on the examined data in this section.

4.6.2 Comparing models performance

The results indicate that all the trained models on the balanced data can predict the session purchasing intention. The lowest performance on the balanced data was for the GraphConv model with an AUC of 0.70, and the best performance was for the ANN model with an AUC of 0.82 flowed by GraphSAGE with AUC of 0.80. On the other hand, ANN, CNN, Simple RNN, and LSTM models achieved weak performance when trained on the unbalanced dataset.

ANN models: We observed the least performance of ANN models when it is trained on the

unbalanced data with an AUC of 0.68, which means its capability to distinguish between the positive and negative labels is low. The performance of the ANN model when it is trained on the oversampled data has increased by 5 points, and its capability to differentiate between the binary classes got better. In contrast, the trained model on the balanced data showed effectiveness as a binary classifier and achieved an AUC of 0.82.

CNN models: The models trained on the unbalanced and the oversampled data achieved an AUC of 0.67 and 0.70, respectively, while the trained model on the balanced data achieved an AUC of 0.82. CNN models show less performance than ANN models with the experimental data.

Simple RNN: the models achieved an AUC of 0.65,0.68 and 0.81 when trained on unbalanced, oversampled data and balanced distribution. These results are less than what ANN achieved with the same data.

LSTM models: The model trained on the balanced data achieved as the Simple RNN model did with an AUC of 0.81. The other two models achieved less than ANN models with the unbalanced and oversampled data with an AUC of 0.67 and 0.69, respectively.

GraphSAGE models: Achieved the high performance when trained on the balanced data with an AUC of 0.80 and the highest recall score 80% between the other models**.** The model achieved AUC of 0.52 when trained on the unbalanced data and achieved on the oversampled data AUC of 0.71.

GraphConv models: achieved the lowest performance among the rest of the models.

The general comparison of all the models indicates that ANN achieved better than CNN, Simple RNN and LSTM models. In addition, GraphSAGE outperforms GraphConv and achieved the lowest number of the false negative labels.

Table 4.4. performance comparison

Models	Unbalanced dataset	Balanced dataset	Oversampled dataset
ANN	TP: 0.0 FP: 16959 TN: 183041 FN: 0.0 Accuracy: 0.92 Precision: 0.0 Recall: 0.0 AUC: 0.68 PRC: 0.18 F1 Score: NA	TP: 10915 FP: 2709 TN: 14121 FN: 6155 Accuracy: 0.74 Precision: 0.80 Recall: 0.64 AUC: 0.82 PRC: 0.84 F1 Score: 0.71	TP: 40976 FP: 18087 TN: 67966 FN: 32315 Accuracy: 0.68 Precision: 0.70 Recall: 0.56 AUC: 0.74 PRC: 0.71 F1 score: 0.66
CNN	TP: 0.0 FP: 8560 TN: 91440 FN: 0.0 Accuracy: 0.92 Precision: 0.0 Recall: 0.0 AUC: 0.67 PRC: 0.18 F1 Score: NA	TP: 5459 FP: 1289 TN: 7162 FN: 3093 Accuracy: 0.74 Precision: 0.80 Recall: 0.64 AUC: 0.82 PRC: 0.85 F1 Score: 0.71	TP: 17856 FP: 8125 TN: 35021 FN: 18656 Accuracy: 0.66 Precision: 0.69 Recall: 0.49 AUC: 0.71 PRC: 0.68 F1 Score: 0.56
Simple RNN	TP: 28 FP: 61 TN: 91379 FN: 8532 Accuracy: 0.91 Precision: 0.31 Recall: 0.003 AUC: 0.65 PRC: 0.17 F1 Score = 0.006	TP: 4842 FP: 1141 TN: 7404 FN: 3616 Accuracy: 0.72 Precision: 0.81 Recall: 0.57 AUC: 0.79 PRC: 0.81 F1 Score = 0.78	TP: 20885 FP: 13465 TN: 29565 FN: 15684 Accuracy: 0.63 Precision: 0.61 Recall: 0.57 AUC: 0.68 PRC: 0.64 F1 Score =0.59
LSTM	TP: 0.0 FP: 8622 TN: 91378 FN: 0 Accuracy: 0.91 Precision: 0.0 Recall: 0.0 AUC: 0.67 PRC: 0.18 F1 Score: 0	TP: 7351.0 FP: 1112 TN: 7384 FN: 3399 Accuracy: 0.73 Precision: 0.83 Recall: 0.61 AUC: 0.81 PRC: 0.85 F1 Score: 0.71	TP: 17454.0 FP: 9196 TN: 33834 FN: 19115 Accuracy: 0.64 Precision: 0.65 Recall: 0.48 AUC: 0.69 PRC: 0.66 F1 Score: 0.55
GraphConv	TP: 0 FP: 34073 TN: 365927 FN: 0 Accuracy: 0.91 Precision: 0.0 Recall: 0.0	TP: 4686 FP: 3868 TN: 11418 FN: 1930 Accuracy: 0.40 Precision: 0.73 Recall: 0.74	TP: 0 FP: 251848 TN: 332172 FN: 0 Accuracy: 0.29 Precision: 0.57 Recall: 0.57

	AUC: 0.50 PRC: 0.18 F1 Score: 0	AUC: 0.70 PRC: 0.57 F1 Score: 0.73	AUC: 0.50 PRC: 0.43 F1 Score: 0.57
GraphSAGE	TP 119 FP: 1444 TN 22466 FN 971 Accuracy: 0.90 Precision: 0.1 Recall: 0.07 AUC: 0.52 PRC: 0.06 F1 Score: 0.09	TP 4609 FP: 1126 TN 4659 FN 1121 Accuracy: 0.80 Precision: 0.80 Recall: 0.80 AUC: 0.80 PRC: 0.74 F1 Score: 0.80	TP 1687 FP: 1853 TN 13067 FN 777 Accuracy: 0.85 Precision: 068 Recall: 0.48 AUC: 0.71 PRC: 0.43 F1 Score: 0.56

4.7 Discussion

Based on the experimentation to train different deep learning models on the session-based dataset with different distributions and different data structure, the following outcomes are obtained:

Model performance: from the results, all the deep learning models on our experiment are not capable of performing accurate predictions when trained on the unbalanced dataset and achieved better performance after applying SMOTE technique. This could be the result of the small number of the positive class in contrast with the negative class, as the models learned a small number of observations with positive labels. When the models were deployed on the unseen observation, it predicted the labels whether randomly or labelled all the data as negative labels. Graph models were learned fast as 5 epochs only were enough to achieve high performance on the balanced data, while with sequence models learned after 30 to 100 epochs with early stopping feature.

Experimental data: One of the disadvantages of the dataset is consisting of one type of observation. This may be one of the reasons affecting the performance of the models. In PI systems, it may be more convenient to combine several features of the user in the system to generate solid patterns.

CHAPTER 5 - PREDICTIVE MODELS LEARN CPDM FEATURES

In this section, we describe our experimentations to develop predictive models based on machine learning algorithms trained on the features of CPDM - see chapter 3- with different structures (table-base and graph-base). We examine three predictive tasks: churn probability, next purchase time and next item recommendation. We developed several models with different ML algorithms and different sets of features for each task to discover the best model for each task. We first present the applied algorithm, then, we discuss the techniques used for data preparation and feature engineering. Finally, we identify the predictive tasks and the models' architecture for each task and its performance.

5.1 Classification Algorithms

In the following section we identify each algorithm we applied in the experimentation process. Firstly, we applied the classic machine learning algorithms on tabular data, naming, Logistic Regression (LR), Decision Tree Classifier (DT), Random Forest Classifier (RF), Support Vector Classification (SVC), Gaussian Naive Bayes Classifier (NB), and a Stacking Ensemble Classifier which combine all the previous algorithms in one model. Then, Forward Neural networks (FNN) algorithms on tabular data, and Graph Neural Networks (GNN) algorithms on graph data.

Logistic Regression (LR): It is a statistical model, which analyzes the relationship between a set of dependent variables and the independent variables and estimates the probabilities (Q. Q. Wang et al., 2019).

Decision Tree Classifier (DT): a class discriminator which divides the training data recursively until each section consists of samples from one class(Swain & Hauska, 1977).

Random Forest Classifier (RF): a classifier contains several subsets of decision trees and takes the average between them (Chaudhary, Kolhe, & Kamal, 2016)

Support Vector Classification (SVC): classification algorithm that finds the optimal boundaries between different outputs after transforming the data using kernel function (Shawe-Taylor & Sun, 2011).

Gaussian Naive Bayes Classifier (NB): classification algorithm based on Bayes' theorem (Tripathi, Yadav, & Rajan, 2019).

Stacking Ensemble Classifier: it is an ensemble learning model that uses the previous algorithm's prediction as a new feature to train a meta-classifier (Ceballos 2019).

Graph Neural network (node-level): it is a type of neural network that can be applied on graphs (Menzli 2020). Node classification model is based on one graph containing several nodes and edges. The classifier model understands the node feature through message passing and propagation, then returns the node's updated feature to predict its label.

Graph Neural Network GNN (edge-level): The link prediction model updates the node's features, then embed the edge between nodes by combining their updated features to learn the edge label (0 or 1). The result is a new set of edges.

5.2 Data Preparation

In the experimentation, we will apply different algorithms to learn features in the relational structure and graph-based structure. This approach aims to compare the performance of the baseline machine learning algorithm to measure the CPDM features' ability to describe the customer's behaviour and to find the best structure of these features. In the following we will describe the features engineering techniques to be used to simplify the features in CPDM database, and how to extract table-base and graph-base features.

5.2.1 Feature Engineering Techniques

In this study, we used manual techniques to manipulate the features in CPDM.

5.2.1.1 Features Extraction

We extracted new features from the transactions table to find new features describe customer's purchasing behaviour. We stored the new features in the purchasing gaps table $purchGap$, and it contains the purchasing gaps values between the customer's purchases and a value to cluster the customer based on their behaviour.

We applied the method in (Droomer and Bekker 2020) to construct purchases gaps features. Firstly, we calculated the gap between the last purchase in the first six months records in *purch* table (see Eq. 5.1) and the first transaction in the last three months (see Eq. 5.2) for each customer and stored the results in a temporary value gap to classify the customers into three classes as Eq. 5.3 to be the next purchase time label. Then, we aggregated the transactions in the first six-month records to calculate the gaps between the last four purchases for each customer. Each record in the $purchGap$ contains seven variables as shown in Table 5.1: Customer ID id_i , the difference in days between the last four transactions of each customer ($diffPur1-2_i, diffPur2-3_i, diffPur3$), the mean and standard deviation of the differences, $diffMean_i, diffStd_i,$ and the segment label k_i (Karaman 2019).

- $purch_i\ prev_{6m}$:

$purch_i\ when\ (tsmp_i\ 1^{th}\ \text{in}\ 1st\ month\ <=\ tsmp_i\ <\ tsmp_i\ 1^{th}\ in\ 6th\ month)$ **Eq. 5.1**

- $purch_i\ nxt\ _{3m}$:

$purch_i\ \ when\ (tsmp_i\ 1^{th}\ in\ 6th\ <=\ tsmp_i\ <=\ tsmp_i\ 1^{th}\ in\ 9th\ \ month)$ **Eq. 5.2**

$if\ (gap\ >\ 60\ days, y=\ 0),$ **Eq. 5.3.A**

$If\ (gap\ >\ 30\ days, y=\ 1),$ **Eq. 5.3.B**

$if\ (gap\ <=\ 30\ days, y=\ 2)$ **Eq. 5.3.C**

Table 5.1. Example of Purch Gap dataset

Attribute	Value
Customer ID	456
Difference in days between purchases 1 $diffPur1_{nextPur}$	12
Difference in days between purchases 2 $diffPur2_{nextPur}$	3
Difference in days between purchases 3 $diffPur3_{nextPur}$	45
Mean of differences $diffMean_{nextPur}$	20
Standard deviation of differences $diffStd_{nextPur}$	18
Class of the next time to purchase	1

5.2.1.2 Features Transformation

To build an optimal model, we well reduce the dimensionality of the data in CPDM and transform existing features.

- In the feedback tables (cases and tweets), we reduce the number of features and data points by aggregating the data to contain one record for each customer. The feedback scores for each customer's records were averaged. Consequently, each customer has a feedback score based on their feedback score on different channels.
- In the page views table, there are several instances for each customer. To group each customer's records in one instance, the sum of the page views durations for each customer were stored as views duration variable for each customer.
- The binning technique helps to regulate the data and prevent overfitting (Rençberoğlu 2019). We applied the binning technique for some continuous:

- 5 bins for each of the feature's *age* a_i and the number of transactions m_i, in customer profile $prof$ and CLV tables, respectively.
- 4 bins for total spend sp_i in CLV table.
- 3 bins for the product price $pric$ in transactions $purch$ table.

- A log transformation was applied on the continuous features to handle skewed values:
  - Number of days since last purchase in CLV table.
  - loyalty points in customer profile table.
  - Total page views duration variable.

5.3 Extract Relational and Graph Databases

In this section, we present the extracted features for customer view from CPDM in two different structures and a discussion to compare between them.

5.3.1 Relational Data

Based in the tables in the database of CPDM and the features engineering techniques, we construct a dataset contains one record for each customer (*full-Prof)*, and a dataset contains the transactions records (*transactions)* for all the customers.

The values in ***full-Prof*** for each customer are as below:

- All the values in customer profile table.
- All the values in CLV table.
- All the values in the purchase gap table.
- The log transformation for the sum of page views duration for each customer.
- The feedback average score for each customer.
- All the values in *purchGap* table
- There are two labels: Churn probability and next purchase time.

The values in ***transactions*** dataset are as below:

- All the records in the *purch* table
- Map all the values in the product table.

We constructed for the relational data two datasets, *full-prof* and *transactions*. The *full-prof* contains all the explicit and implicit features for each customer, and the *transactions* include the purchasing records with product information.

5.3.2 Graph-based Data

Based on the graph data structure we constructed an undirected heterogeneous graph (Figure 5.1) containing several types of nodes and edges. The nodes in the graph are customers u, products p, age categories a, subscription types s, gender category g, CLV categories clv, states st and cities cty.

Each node in the graph has at least one relation with the customer node. The customer node consists of a set of features and has two labels: churn probability and next purchase time. The features in the customer node represents the customer behaviour features as described below:

- Customers subscription information: subscription period by years, the renewal status and the loyalty points.
- Customer's RFM variables: number of transactions, total spend, and number of days since the last purchase.
- The average of the Customer's feedback values through different channels.
- Customer's page views duration value.
- Customer's purchasing gabs: the gaps between the last three transactions, mean and standard deviation of the gaps between the last three transactions.

The demographic information (age, gender, state, city), the subscription types, the products and the CLV segments are represented as separate nodes, and each customer has relations with them. The strategy behind the structure of the proposed graph is to connect the customers with

similar characteristics. For example, if we have two customers in the same age group, their nodes will have a link to the same age node. Consequently, their features will be sharable during the graph representation process. The following are the key characteristics of each node connected to the customer node:

- The age nodes: there are five age nodes, and each one represents an age group. Each node has the age category id as a feature.
- The gender nodes: there are two nodes for male and female. Each node has the gender category id as a feature.
- CLV nodes: there are three nodes, and each node represents the CLV segment. Each node has the mean value of its CLV category.
- Subscription nodes: there are three nodes for three types of subscription. Each node has the id of the subscription type as a feature.
- City nodes: there are *298* nodes, and each node represents a city. The city id is the feature for each node.
- State nodes: there are 47 nodes, one for each state. The id is the feature for each node.
- Product nodes: there are 250 nodes, and each node represents a product, and it contains the product features (category and price).

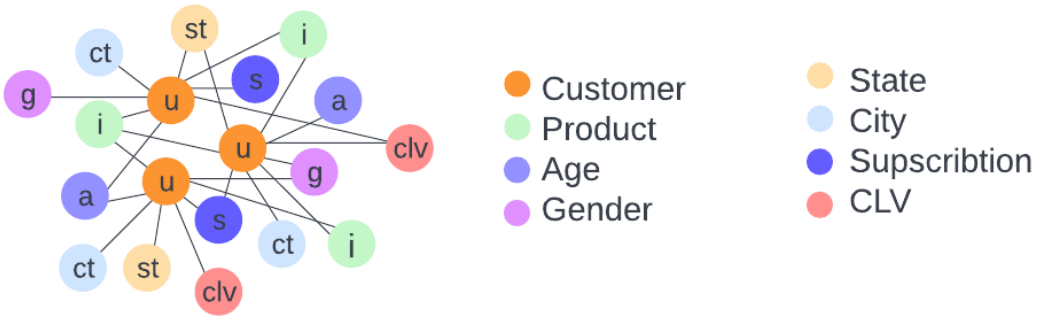

Figure 5.1. Heterogeneous graph structure based on the CPDM

5.3.3 Comparing The Proposed Relational and Graph Data

There are differences in constructing the entities and the features for each entity. In the relational database, there are two entities: Customers and Transactions. We created two sets of data to present the features in the customer and the transactions entities, naming, *full-prof* and *transactions*. We could not merge all the information in one set because the formal dataset contains one record for each customer, while the latter contains several records for each customer. Based on CPDM, the customer features contain information from the customer profile, customer activity, customer feedback, and CLV datasets. On the other hand, in the graph data, we constructed eight types of nodes representing the customer entity, product entity, and categorical entities that the customer belongs to. These nodes are customer, age, gender, city, state, CLV cluster, subscription type and product. The records of the customer transactions are represented by the edges between the customer node and the product nodes. If the customer has purchased a product several times, the relation is translated as edges between the customer and that product's node.

The relations between the data points are presented clearly in the graph data. Each customer's record on the tables does not relate to the other customers. While in the graph, the customer connects with other customers as second neighbours. For example, there is a relationship between the customers who have purchased the same product. Finding any other shared relations between these customers is easy, which facilitates analysis and clustering.

The relationship between the customer and their transactions are presented differently. The customer ID represents the relation between the customers in *full-prof* and their records in *transactions* tables. In the graph data, the edges represent the relation between the customer and the products they purchased.

Finally, the graph structure algorithms identify the customer as the most connected entity to other nodes, which is challenging to be identified by the table.

5.4 Predictive Tasks

In this section, descriptions of the predictive tasks in our proposed models.

5.4.1 Modeling Customer's Churn Probability

Predicting the churn probability is a binary classification task which needs a number of features as predictors and labels with values (0 or 1) as a dependent feature. The initial set of features are as the following:

Independent Features: to train a model for predicting the customer's churn, we relied on the customer's demographics, and subscription information.

Dependent Feature: Customer churn variable. Each customer has a binary record (1 or 0) describing whether the customer is tending to leave the business or not based on their last transaction gap and total spending.

Based on the related literature, the customer's individual information and RFM variables were considered to influence the churn action. Thus, we will experiment with these features and other information to find the best set of features that improve the predictive analysis.

5.4.2 Modeling Customer's next purchase time

In our experimentations, predicting the next purchase time is a multi classification task that needs a number of independent features and three labels. From the previous studies, we conclude that the customer's sequence of transactions influences the next purchase in the future. Initially, the main features to be used are:

Independent Features: the features that represent the purchasing gaps.

Dependent Features: The next purchase time variable. Each customer has a class (0, 1, or 2).

We conducted several experiments to enhance the model performance by feeding the model with several features.

5.4.3 Modeling Customer's next Item Recommendation

We constructed a Collaborative Filtering Recommender System that analyzes the features of the transaction and predicts a binary label to find if the customers will likely interact with the product. In the previous studies for predicting the next item, the main variables were the customer and the items they interacted with. Some studies relied on customer and product features to improve recommender performance. The features in that model selected as the following:

The independent features: The leading independent features are the customer id and the product id. To improve the performance of the recommendation, we inspected how to add features related to the items or/and the user to encourage the model to find valuable patterns. We applied several trials to feed the model with each item's category, price, and customer features.

The dependent feature: It is the explicit rating (0 or 1), and all the purchased products have labels with value (1). Negative sampling techniques are applied to add products to the customers that they have not purchased and label these products with (0).

5.5 Features Embedding Techniques

Based on the type of features in the relational and graph data, we prepared the values of these features before feeding them to the predictive algorithms. We have in the models three types of features, naming, continuous, nominal, and ordinal features.

For the graph representation, the graph nodes' attributes need to be normalized to sum-up to one. To achieve that, we use the Normalize function API from Pytorch Geometric library.

The normalization enhances the algorithms performance specially when the data has different ranges. Therefore, we will not make any further manipulation on the nodes' ordinal and the continuous features.

5.5.1 Embedding Nominal Features

The nominal features in the models are all the categorical features without order, such as gender, customer id, product id, city, and state.

We apply the One hot encoding technique in the SML models on the nominal categories. Then, they are scaled using a standard scaler (Duchesnay, 2011) to transform them into a set of features with 0 as the mean and one as the standard deviation.

In the FNN and GNN models, we add an embedding layer in the models. The embedding layer considers each category as an index to find a relative weight in a lookup table (Venkatesh, Moffat, Reck, & ‡1, 2022).

5.5.2 Embedding Ordinal Features

The ordinal features in the models are age bins, CLV clusters, subscription type, total spend pins, and product price pins.

In SML and FNN models, we use a Label encoder from the Sklearn library (Duchesnay, 2011) to encode the ordinal labels; then, we scale it using the standard scaler.

5.5.3 Embedding Continuous Features

The continuous features in our models are the loyalty score, feedback score, page views duration, and all the purchasing gap independent features.

In SML and FNN models, we use the standard scaler to transform the features into values with zero as the mean and one as the standard deviation.

5.6 Models Architecture

This section describes the models' architecture based on baseline models. We used standard ML algorithms, Feedforward Neural Networks, and Graph Neural Networks in the predictive models for the tasks: customer churn, Next time purchase and Recommendations for the next item to purchase.

5.6.1 Classification Models

Classifier relies on a set of features to predict the customer label. The dependent features are a structured feature and mix of categorical and continuous values in CPDM.

We applied three different types of classifiers. We train Standard ML algorithms (SML) and feed-forward Neural Networks (FNN) on relational data, and Graph Neural Networks (GNN) are trained on graph data. We used the cross-validation technique with SML to randomly sample the dataset to use different portions to train the model in several iterations. Data Loaders and several iterations of training were applied in training FNN and GNN. In the following, we describe the architecture for each classifier for the customer's churn task and next purchase time.

5.6.1.1 Basic ML algorithms Classifiers

The basic ML models we apply are Application Programming Interfaces (API) from Sklearn Library (Pedregosa et al. 2011). We split the data into train and test sets; then, we feed the train data to an API after manipulating it with the embedding techniques to learn the data patterns. We feed the test set to the model and measure the classification results to evaluate the model performance. Each model consists of:

- One hot encoding algorithm.
- Standard scaler to transform the all the features.
- API function for training.

- Labels prediction.

5.6.1.2 Feedforward Neural Network Classifiers

Each customer in the relational data has one instance with a set of features. Before training the set of features, we split it into train and test sets. The model learns patterns from a set of customers in the training set and predicts the labels of different customers in the test set based on the learned patterns. As in Figure (5.1), the model consists of:

- Standard scaler to transform the ordinal and the continuous features.
- Embedding layers to transform the nominal features.
- Input layer to concatenate the embedded nominal features with the other features (ordinal categories and continuous features)
- Linear hidden layers followed by an activation function learn the training features.
- Linear output layer, which produces the predicted label. Two labels in the churn probability and three labels in the next purchase time classification tasks.

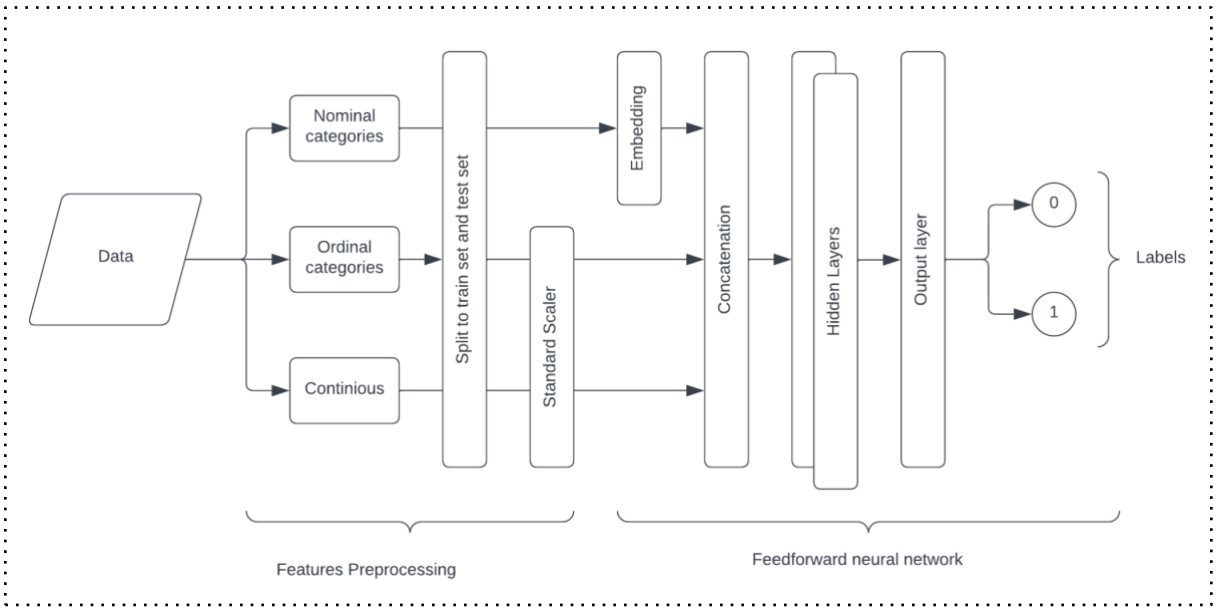

Figure 5.2. Feed Forward Neural Network

5.6.1.3 Graph Neural Network (GNN) classifier (node level)

It is a GNN for node classification to predict the labels of the customer nodes. To train the GNN model on the graph data, we first randomly split the nodes to train and test nodes, then transform the features.

For graph features aggregation, we use GraphSAGE, a graph embedding framework that leverages node features to generate unseen node embedding efficiently (W. Hamilton, Ying, and Leskovec 2017). In the model, there are two layers of GraphSAGE to aggregate, and update node features twice. In the first layer, the nodes are updated with the first neighbours' features, and in the second layer, the second neighbours with their updated features are aggregated and update the nodes with their features. At the first layer, the customer node will not obtain information from the users with similar relations. There should be another hop, so each customer node obtains information from similar customers. As in Figure (5.2), the model consists of:

- Standard scaler to transform the ordinal and the continuous features.
- Embedding layers to transform the nominal features.
- Input layer to concatenate the embedded nominal features with the other features (ordinal categories and continuous features)
- Feature normalizer layer.
- GNN layers: GraphSAGE.
- Linear hidden layers followed by an activation function learn customer node features.
- Linear output layer, which produces the predicted label.

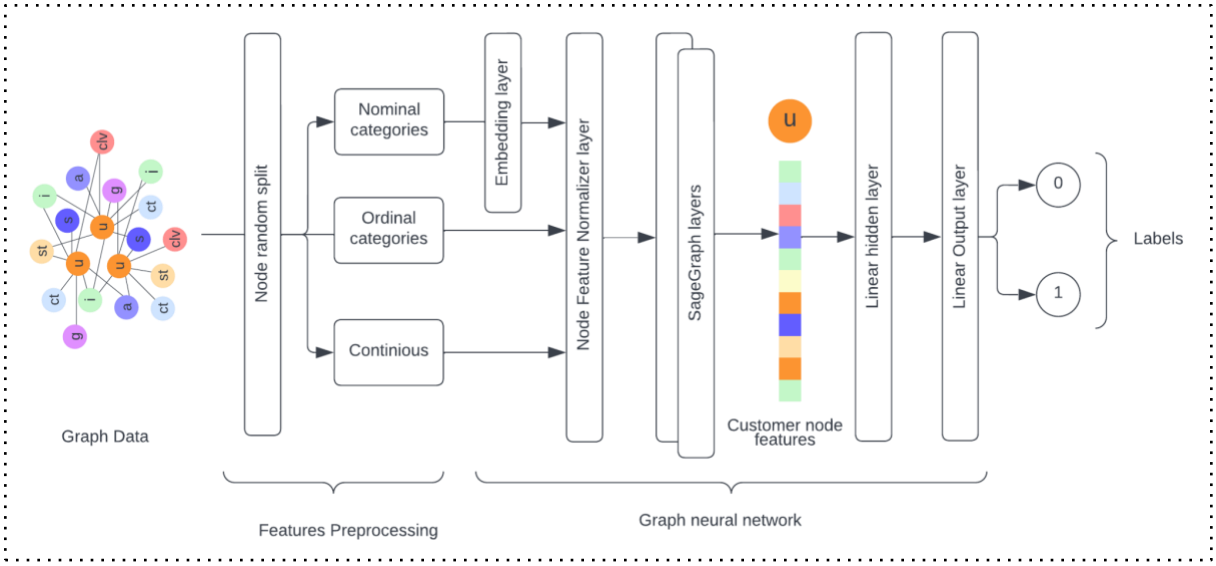

Figure 5.3. Graph neural network architecture for classification

5.6.2 Recommender Systems:

The recommendation system architecture is designed to analyze customer transaction records and learn their preferences to find patterns between customer orders and similar customers.

5.6.2.1 SML algorithms Recommender:

The models are like the classifiers; however, the data pre-processing and feeding are different. Since each customer in the *transactions* dataset has several instances, it is required to include the customer Id as a feature. In addition, the product Id and the category Id of the product are added as product features. All the ids are nominal features which we usually transform into One hot encoding. However, with many customers and products, the One hot encoding was not a suitable solution because applying it needs enormous memory. We feed the nominal features without transforming the models, instead of excluding these algorithms from this part of the experiment.

5.6.2.2 Feedforward Neural Network Recommender:

It is a model for predicting the products that will likely be purchased next by the customer based on their purchasing history. It is a feature vector model based on customers' preferences or ratings, which in our model are based on whether the customer has bought a product (rating =1) or not (rating =0).

All the products in the *transactions* dataset has 1 as the rating value, and it is required to add a number of negative samples of products the customer has not purchased in the training set to add negative ratings with value 0 as labels. We develop a popularity-based negative sampling technique inspired from (Liang et al. 2016), and (J. Tang and Wang 2018) which relies on the number of times each product has been purchased by the customers. To reduce the globality of the samples, we first applied the cluster technique 'KMeans' on the customers features to group the customers into five groups. Then, inside each group we find the number of times each product has been purchased and calculate the mean. We consider the products that were purchased less than the mean in each group as negative samples for the customers in these groups. To achieve this technique in the model, we chose randomly for each customer a small number of products (5) from their group's negative samples within each training iteration to decrease the overhead of the model training, and then update the dataset with new instances of negative products for each customer.

In the training, the model learns - as a binary classifier - each customer's preferences and their labels (0 or 1). The model learns and predicts a label for each customer with each associated product. When the predicted label is 1, the associated product is considered as a recommended product to the customer to purchase in the future.

We feed the model with features for both the customer and the product to discover if these features increase the model's performance. The architecture of the model is as shown in Figure (5.3) and it consists of:

- Standard scaler to transform the ordinal and the continuous features.
- Embedding layers to transform the nominal features.
- Input layer to concatenate the embedded nominal features with the other features (ordinal categories and continuous features)
- Negative sampling function to add negative products for each customer and label them with 0 rates.
- Linear hidden layers followed by an activation function learn the training features.
- Linear output layer, which produces the predicted rates.

The differences between the classifier model we applied in this research and the recommender model are concluded in the following three points:

1. Each customer in the recommender has several instances, thus the customer Id is used as a feature.
2. When we split the dataset instances, each customer has a percentage of their records in the train set and in the test set.
3. The model learns and predicts the label based on the patterns learned for each customer and the patterns learned from similar customers.

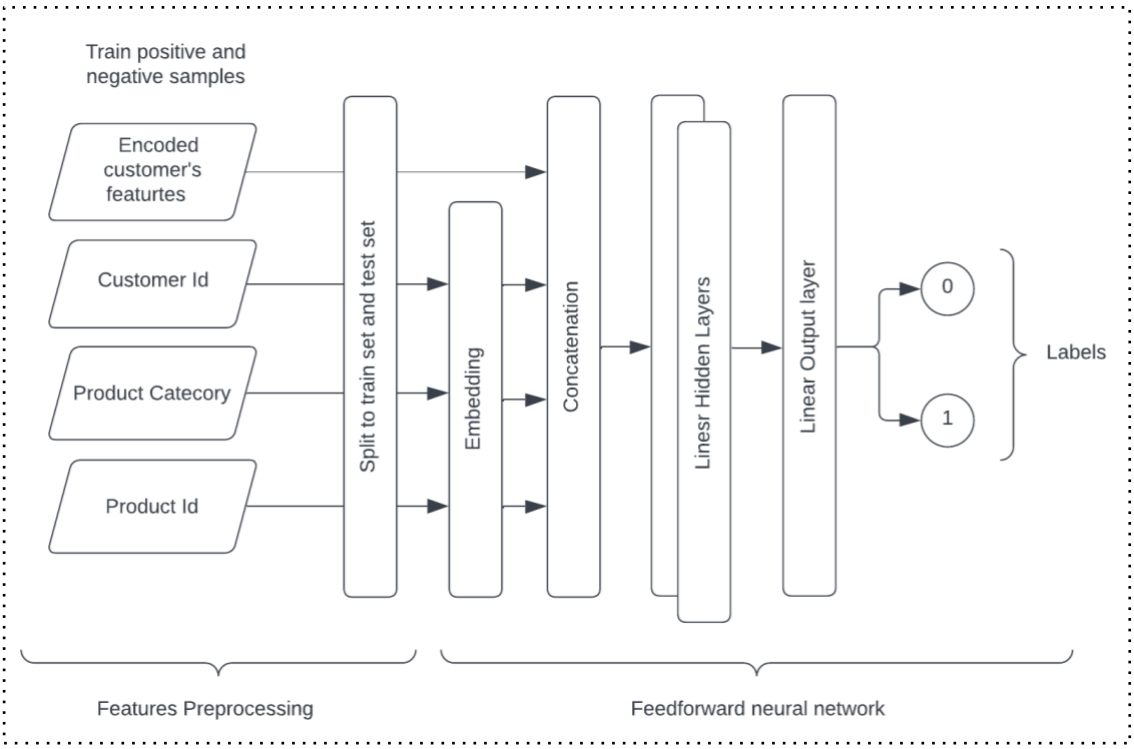

Figure 5.4. Feedforward neural network architecture for Recommendation

5.6.2.3 Graph Neural Network (GNN) recommender (edge level):

It is a GNN for edge prediction to learn and predict the edge labels between the customer node and the product nodes. The link prediction model is based on encoder and decoder systems. It learns the relation between the nodes based on analyzing the combined features of the linked nodes to predict a new set of retaliation between them.

The model's architecture in Figure (5.2) is semi-similar to the model in Figure (5.4). The GNN for recommendation model consists of:

- Randomly split for the edges to train and test sets
- Standard scaler to transform the ordinal and the continuous features.
- Embedding layers to transform the nominal features.
- Input layer to concatenate the embedded nominal features with the other features (ordinal categories and continuous features)
- Feature normalizer layer.
- GNN layers: GraphSAGE.

- Concatenation layer to combine the presentation of the product and customer features
- Linear hidden layers followed by an activation function learn the concatenated presentation of the product and customer nodes.
- Linear output layer produces the predicted label for the edges between the customers and the product nodes.

The differences between the models are concluded in three points:

1. We randomly split the edges between the customers and the products to train the GNN model to learn the edges on the graph data.
2. We developed an encoder of two layers using GraphSAGE class to encode all the updated nodes features.
3. The decoder is a short feedforward neural network to process the concatenated representations of the customer's node features and the product's node's features in relation, then produces a new set of links between them with labels of 0 or 1 values.

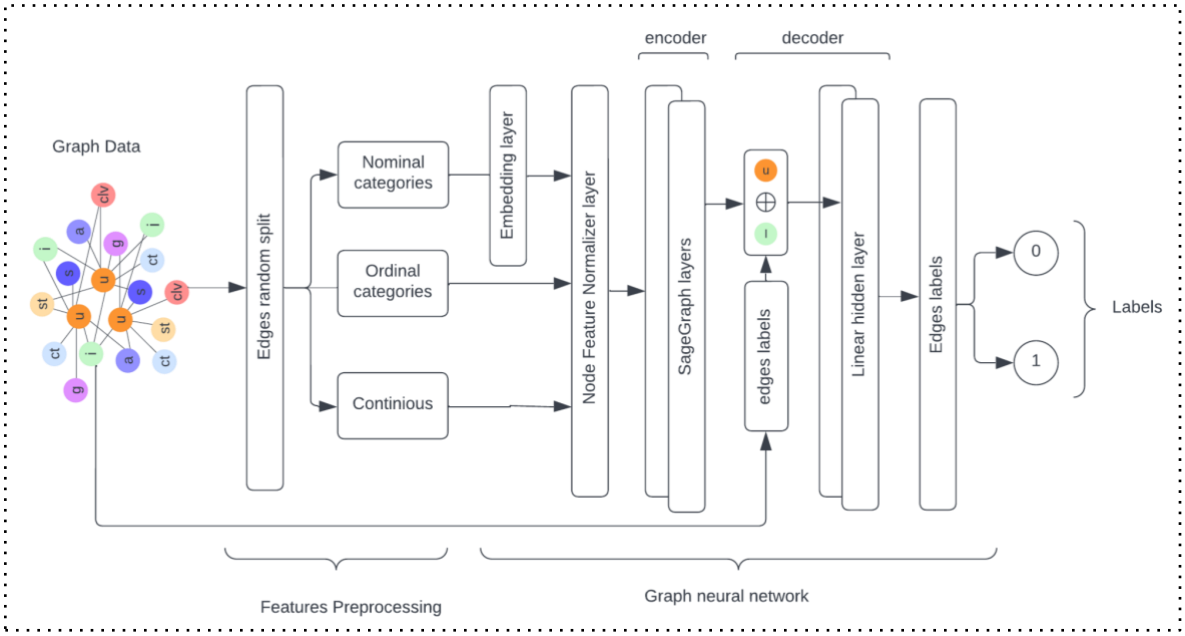

Figure 5.5. Graph neural network architecture for recommender system

5.7 Comparing The Predictive Models

There are different set up for each data structure. In this section, we compare the models' architecture in section 5.6 which influenced by the trained data.

5.7.1 Comparison of Classifiers Comparison

The classifiers predict customer labels based on customer features. The SML and FNN models analyze and learn patterns in the linear features and predict the labels of unseen data. In other words, if we use the customer features in *full-profile* for training, we can not feed the model with the features in the *transactions* dataset. The reason is that each customer has several records in the latter and it needs a more complicated model to align the two datasets in one model. In contrast, the GNN models analyze the nonlinear data and can easily align the customer with multiple transactions in the same model.

The SML and FNN analyze the fed features and predict a label for these feature representations. The GNN works differently. In the GNN, an encoder extracts a new representation and updates each node based on the neighbour's features. Then, small FNN analyzes and learns the updated representations of the customer node to predict the labels.

5.7.2 Comparison of Recommender Systems

The recommender systems in section 5.6 are binary classifiers learned from the customer and product features. The SML and the FNN models should take features from *full-profile* and *transaction* datasets. The issue is that the customer in the former has one record and, in the latter, has several records. To align the two datasets and feed them to the recommender model, we choose to duplicate each customer data point equal to the number of their records in the transactions table. After the duplications, the model can take both datasets to learn customers' features and product preferences to predict customers' next items. This approach increases the volume of the data; however, it may influence positively on the model performance. The GNN recommender systems are based on the exited edges between the

product and the customer nodes. The negative sampling in the GNNs helps create a virtual link between the negative products and the customer node and takes zero as a label. GNN prediction is a new set of edges between the customer and the product nodes. The edges with label (1) are considered recommended products.

5.8 Experiments:

This section elaborates on our implementations of three predictive tasks on different features. Libraries used in the implementations are Pandas, NumPy, Pytorch, Pytorch Geometric (PyG) (Fey and Lenssen 2019) and Sklearn. To make the computation of neural networks smoother, we used the GPU to handle the training for FNNs and GNNs.

For each task, several techniques and learning methods are applied. The results were compared to estimate the best model performance based on the features and the predictive algorithms. The performance comparison of different models in each task will be based on precision, recall, and F1 score results, as the latter is a harmonic mean of precision and recall ("Sklearn.Metrics.F1_score" n.d.).

For each task, the purpose of comparing the models' performances is to find the best features capable of training the predictive algorithms to achieve the task. Then, we find the best results and select the used features and predictive algorithm as the optimal solution for the task. Having this information for each task is for constructing the multi-task CBPred model.

We specifically need to answer the following questions:

- What are the most suitable features for each task?
- What is the best algorithm to predict each task?

With the FNN and SML, we can easily feed the model's different feature sets in each experiment. While with the graph data, we may need to delete nodes or node features to

examine the graph-based models. We change the graph data to combine only the examined customer's features in all the experimentation. For example, if the experimented features are the demographic and the subscription information. We delete the CLV nodes and keep only the customer node's features that describe the subscription information.

For experimentation with SML models, we applied the default hyperparameters for the model's APIs, as shown in Table 5.1. With FNN and GNN models, we applied an automatic hyperparameter optimization called Optuna (*Optuna: A Hyperparameter Optimization Framework* n.d.). Optuna finds the optimal settings of hyperparameters using the test set. The hyperparameters are latent dimensions, regularisation parameters, learning rate, batch size, number of layers, and the optimizer.

Table 5.2. SML Hyperparameters for classification

Model	**Hyperparameter**
LR	penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept scaling=1, class weight=None, random_state=1, solver='lbfgs', max_iter=100, multiclass='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None
DT	criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random state=1, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0
RF	n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=1, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None
NB	priors=None, var_smoothing=1e-09
SVC	C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=- 1, decision_function_shape='ovr', break_ties=False, random_state=0
KNN	n_neighbors=20, weights='distance, algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None
Stacking Classifier	estimators= level0, final_estimator= level1, cv=5, stack_method='auto', n_jobs=None, passthrough=False, verbose=0

5.8.1 Customer Churn probability experiments:

This task is a binary classification task Figures 5.1 and 5.2. The classifiers are trained separately on four sets of features to discover the best combination of features that will lead to the best performance in the model. We examine all the sets with the SML algorithms, FNN, and GNN models.

5.8.1.1 Features sets:

To answer the first question in our experiments, we split the features into four sets.

6. Set 1: the customer's demographic data and their subscription information.
7. Set 2: combining Set 1 with, CLV segment, and number of transactions.
8. Set 3: combining Set 2 with customer's feedback and page views duration.
9. Set 4: combining Set 3 with purchasing gaps between the last three transactions, the mean and standard deviation value

   Note: we exclude the features we used to segment the customers' churn label: 1-days since last purchase, 2- the total spent.

5.8.1.2 Hyperparameters:

The hyperparameters of SML models are shown in Table 5.1. Table 5.2 shows the recommended hyperparameter for FNN and GNN models after applying the Optuna framework. The following are our observations on the recommended parameters by Optuna:

- The FNN models recommended hyperparameters:
  - The models need only one or two feedforward layers with different sets.
  - With Set1, the recommended optimizer was 'Adam', which updates the model's weight based on the learning rate and the average of gradients. The optimizer

'Adadelta' was recommended to sets 2 to 4, which relies on the magnitude of the gradients to adopt the learning rate (Karim 2018).

- The recommended learning rate increases when the number of the fed features increases.

- The recommended hyperparameters for the GNN models are almost similar for all the sets. Therefore, we used one set of hyperparameters with all the feature sets.

Table 5.3. FNN Hyperparameters of Churn probabilities models

Model	Hyperparameter	
FNN	**Set #1:** num. layers = 1, embedding = 10, hidden features = 105, learning rate = 0.0002, optimizer = Adam, batch size = 224, dropout = 0.4	**Set #2:** num. layers = 2, embedding = 4 hidden features = 381, learning rate = 0.01, optimizer = Adadelta, batch size = 348, dropout = [0.2, 0.5]
	Set #3: num. layers = 1, embedding = 10, hidden features = 78, learning rate = 0.01, optimizer = Adadelta, batch size = 224, dropout = 0.1	**Set #4:** num. layers = 1, embedding = 10, hidden features = 322, learning rate = 0.1, optimizer = Adadelta, batch size = 256, dropout = 0.4
GNN	num. layers = 2, embedding = 10, hidden features = 64, learning rate = 0.004, optimizer = Adam dropout = 0	

5.8.1.3 Results:

In Table 5.3, the results of the different classifiers trained on four sets of features to predict customer churn.

Table 5.4. churn probabilities result with four sets of features

Features	**Churn probability results in SET #1** Demographic and subscription information						**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Average precision**	**Num. epochs x sec/epochs**
LR	0.50	0.50	0.48	0.50	0.49	0.49	
DT	0.50	0.50	0.48	0.51	0.49	0.48	
RF	0.50	0.50	0.49	0.50	0.50	0.49	
NB	0.50	0.50	0.49	0.53	0.51	0.49	
SVC	0.49	0.49	0.48	0.49	0.49	0.48	
KNN	0.51	0.51	0.49	0.53	0.51	0.49	
Stacking Classifier	0.49	0.49	0.47	0.40	0.44	0.48	
ANN	0.50	0.50	0.49	0.40	0.44	0.49	12x0.2=2.4
GNN	0.73	0.73	0.76	0.69	0.72	0.68	64 x 0.06= 3.84
Features	**Churn probability results in SET #2** Demographic, subscription, CLV segment, and number of transactions						**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Average precision**	**Num. epochs x sec/epochs**
LR	0.94	0.94	0.95	0.93	0.94	0.92	
DT	0.90	0.90	0.89	0.90	0.90	0.85	
RF	0.93	0.93	0.94	0.92	0.93	0.90	
NB	0.58	0.58	0.57	0.57	0.57	0.54	
SVC	0.91	0.91	0.91	0.90	0.91	0.87	
KNN	0.80	0.80	0.78	0.82	0.80	0.73	
Stacking Classifier	0.94	0.94	0.94	0.94	0.94	0.91	
ANN	0.94	0.94	0.94	0.94	0.94	0.91	88x0.23=20
GNN	0.94	0.94	0.94	0.94	0.94	0.91	33x0.06= 1.98

Features	**Churn probability results in SET #3** Demographic, subscription, CLV segment, number of transactions, feedback, and page views						**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Average precision**	**Num. epochs x sec/epochs**
LR	0.94	0.94	0.95	0.93	0.94	0.92	
DT	0.90	0.90	0.89	0.90	0.90	0.85	
RF	0.94	0.94	0.95	0.93	0.94	0.92	
NB	0.58	0.58	0.57	0.57	0.57	0.54	
SVC	0.91	0.91	0.91	0.91	0.91	0.87	
KNN	0.77	0.77	0.75	0.78	0.77	0.70	
Stacking Classifier	0.94	0.94	0.94	0.94	0.94	0.91	
ANN	0.94	0.94	0.94	0.94	0.94	0.91	80x0.23=18.4
GNN	0.95	0.95	0.95	0.94	0.95	0.92	29 x06=1.74
Features	**Churn probability results in SET #4** Demographic, subscription, CLV segment, number of transactions, feedback, page views, and purchasing gaps						**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Average precision**	**Num. epochs x sec/epochs**
LR	0.93	0.93	0.94	0.93	0.93	0.90	
DT	0.91	0.91	0.91	0.91	0.91	0.87	
RF	0.94	0.94	0.95	0.93	0.94	0.92	
NB	0.70	0.70	0.68	0.74	0.70	0.62	
SVC	0.92	0.92	0.92	0.92	0.92	0.88	
KNN	0.79	0.78	0.83	0.70	0.76	0.73	
Stacking Classifier	0.94	0.94	0.94	0.93	0.94	0.91	
ANN	0.94	0.95	0.96	0.93	0.94	0.93	79/0.2=15.8
GNN	0.95	0.95	0.96	0.94	0.95	0.93	35/0.06=2.1

5.8.1.4 Performance Comparison:

In this section, we compare the performance of the churn probability task between the predictive algorithms we applied in the experimentation, which are trained on different sets of features from CPDM.

We found that feeding different models with Set 1 did not achieve proper performance, as the features were not enough to find patterns between the records, and the F1 score for most models was around 50%. However, the GNN model achieved an F1 score of 72%, as the model's ability to predict positive labels is 69%, and the percentage of true positives is 76%. The performance of GNN is probably the best because of the existence of the product node in the graph and its relations with the customer node, which is not included in the *full-prof* dataset where each customer has one record.

For the other three sets of features, we found that with Set 2, the model: LR, ANN and GNN achieved a similar F1 score of 94%. In comparison, GNN achieved the best F1 score of 95% with Set 3 and Set 4. By applying the Early Stop feature while training the models, we found that the GNN needs only 35 epochs and 0.06 seconds for each epoch to achieve these results, which is less than seven times that of the FNN model.

From the results of churn probability models:

1. Set 2 was a good feature for all the models to achieve their highest performance in predicting the churn. Some models' performance is reduced with Set 4 as in the KNN model because of the noise the features may cause. Most models retain high performance with Set 2, Set 3 and Set 4.
2. The best algorithm for the churn probability task is the GNN model when trained on Set 4. The mode achieved the highest F1 score.

5.8.2 Customer Next Purchase time frame models:

This task is a multi-classification task. We examined several sets of features with the same algorithms we applied for the churn probability task (as in Figure 4.1 and Figure 4.2) by changing the number of labels in the output layer.

5.8.2.1 Datasets:

The classifiers were trained on four sets of features to discover the best dependent variables that will lead to the best performance for the models.

- Set 1: the purchasing gaps features
- Set 2: combining Set 1 with demographic and the subscription information dataset.
- Set 3: combining Set 2 with the RFM and CLV cluster.
- Set 4: combining Set 3 with customer's feedback and page views duration.

5.8.2.2 Hyperparameters:

The hyperparameters for the standard ML algorithm are shown in Table 5.1. The ANN and GNN hyperparameters are shown in Table 5.4. Our observation in the following points:

- The FNN Models recommended hyperparameters:
    - Ten dimensions for embedding the nominal categories were enough with all the sets.
    - The learning rate increases when the number of layers increases, as with Set 4.
    - The smaller the training rate, the more significant the recommended dropout values
- The GNN Models recommended hyperparameters:

- The slowest training process (0.00001 as learning rate) was when the model fed with Set 2, which reflected that the features in this set were not the best choice for the GNN model.
- The learning rate increases with Set 3, and Set 4, which means the model learns faster with these sets.

Table 5.5. Next Purchase time models hyperparameter

Model	Hyperparameter	
FNN	**Set #1:** num. layers = 2, embedding = 10, hidden features = 175, learning rate = 0.002, optimizer = Adagrad, batch size = 224, dropout = [0.2, 0.3]	**Set #2:** num. layers = 4, embedding = 10, hidden features = 280, learning rate = 0.1, optimizer = Adadelta, batch size = 348, dropout = [0.4,0.0, 0.0,0.4]
	Set #3: num. layers = 2, embedding = 10, hidden features = 308, learning rate = 0.00008, optimizer = Adam, batch size = 224, dropout = [0.4,0.5]	**Set #4:** num. layers = 2, embedding = 10, hidden features = 184, learning rate = 0.03, optimizer = Adadelta, batch size = 256, dropout = [0.4, 0.4]
GNN	**Set 1:** num. layers = 2, embedding = 50, hidden features = 96, learning rate = 0.0008, optimizer = Adagrad	**Set 2:** embedding = 60, num. layers = 2, hidden features = 112, learning rate = 0.00001, optimizer = Adam
	Set 3: num. layers = 2, embedding = 60, hidden features = 128, learning rate = 0.002, optimizer = Adagrad	**Set 4:** num. layers = 2, embedding = 50, hidden features = 96, learning rate = 0.005, optimizer = Adagrad

5.8.2.3 Results

In Table 5.5, the results of the different classifiers trained on four sets of features to predict the customer's next time to purchase.

Table 5.6. customer's next purchase time model performance

Features	**Next time purchase task results in SET #1** Purchasing gaps features					**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Num. epochs X sec/epochs**
LR	0.72	0.63	0.63	0.72	0.64	
DT	0.65	0.58	0.64	0.65	0.64	
RF	0.73	0.73	0.66	0.73	0.65	
NB	0.1	0.5	0.52	0.11	0.069	
SVC	0.72	0.63	0.65	0.72	0.61	
KNN	0.72	0.59	0.66	0.72	0.61	
Stacking Classifier	0.72	0.71	0.66	0.72	0.63	
ANN	0.73	0.73	0.65	0.73	0.64	186 x 0.2
GNN	0.73	0.65	0.62	0.73	0.65	275x0.06
Features	**Next time purchase task results in SET #2** Purchasing gaps, demographic, and subscription information					**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Num. epochs X sec/epochs**
LR	0.72	0.63	0.63	0.72	0.64	
DT	0.63	0.75	0.62	0.63	0.63	
RF	0.72	0.73	0.65	0.72	0.64	
NB	0.1	0.5	0.52	0.1	0.06	
SVC	0.72	0.63	0.65	0.72	0.61	
KNN	0.72	0.57	0.60	0.72	0.61	
Stacking	0.72	0.70	0.62	0.72	0.62	

Classifier						
ANN	0.73	0.73	0.59	0.73	0.63	186 x 0.3
GNN	0.71	0.66	0.61	0.71	0.64	235x0.06
Features	**Next time purchase task results in SET #3** Purchasing gaps, demographic, subscription information, RFM and CLV cluster					**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Num. epochs X sec/epochs**
LR	0.78	0.84	0.76	0.78	0.76	
DT	0.75	0.71	0.74	0.75	0.75	
RF	0.82	0.87	0.80	0.82	0.80	
NB	0.1	0.5	0.52	0.07	0.3	
SVC	0.74	0.83	0.67	0.74	0.65	
KNN	0.72	0.68	0.59	0.72	0.61	
Stacking Classifier	0.8	0.85	0.78	0.80	0.78	
ANN	0.80	0.78	0.78	0.80	0.78	491 ep x 0.2sec
GNN	0.79	0.79	0.76	0.79	0.77	80 x 0.06
Features	**Next time purchase task results in SET #4** Purchasing gaps, demographic, subscription information, RFM, CLV cluster, feedback and page views duration					**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**Num. epochs X sec/epochs**
LR	0.78	0.84	0.76	0.78	0.76	
DT	0.75	0.71	0.74	0.75	0.74	
RF	0.82	0.87	0.80	0.82	0.80	6 sec/ c.v.* repetition
NB	0.11	0.52	0.52	0.11	0.07	
SVC	0.74	0.83	0.67	0.74	0.65	
KNN	0.72	0.67	0.59	0.72	0.61	
Stacking Classifier	0.80	0.85	0.78	0.80	0.78	

ANN	0.80	0.85	0.78	0.80	0.79	443 x0.5 sec
GNN	0.79	0.80	0.78	0.79	0.78	44x0.07 sec

* c.v.: cross validation

5.8.2.4 Performance Comparison:

In this section, we discuss the results in Table 5.5 and compare our experimented algorithms to select the best performance with a set of features.

By feeding the different classifiers with set 1, the F1 score results for most of the algorithms were around 60%, and the best performance was with RF and GNN models which both achieved 65%. The calculated gaps out of the purchasing features are not enough to predict the next time the customer will likely make a purchase. Adding the demographic, and subscription features in Set 2 has not enhanced the results of all the classifiers, and the best performance was an F1 score of 64% with the algorithms: LR, RF and GNN. All the model performances were enhanced when they were fed with Set 3 features which had the customers' RFM variables. With Set 3, RF algorithms had the best F1 score with 80%, and the second and third best were ANN and GNN, respectively. While the RF and ANN models achieved an F1 score higher one to two points more than the GNN model, the latter learns faster. The GNN model training needs 80 epochs with 0.06 sec for each epoch to achieve an F1 score of 77%, while ANN to achieve one point higher needed around 500 epochs with 0.2 sec for each epoch. Finally, with Set 4, the performance for ANN and GNN improved by one point.

From the results, we conclude the three outcomes of our experiment:

1. The features in Set 1 and Set 2 are not enough to influence the predictive algorithms to find adequate patterns for predicting the next time span for customer purchase. The best performance was for Set 4 which has features from different sources data.

2. The algorithms with best performance were trained on Set 4. The RF model achieved the highest F1 score of 80% but the learning process is very slow in contrast with GNN model. GNN was the fastest and achieved an F1 score of 78%.

5.8.3 Customer's Next item Recommendation:

We constructed the ratings-based recommender systems based on binary classification techniques and trained them on three sets of features. We examined the three sets with the SML algorithms, the FNN model and the GNN model.

5.8.3.1 Datasets:

The experiment on this model was applied on three different sets of features. Set 1 contains the customers and the products ids only. In the other sets, each instance contains the combined features of the customer and the related product as dependent features. We used the product id, and its category as product features, and we examined the following sets for customer features:

10. Set 1: Customers Id.
11. Set 2: Set 1 combined with the customer's demographic data and their subscription information.
12. Set 3: Set 2 combined with the CLV cluster, RFM, page views duration, purchasing gaps and feedback features.

5.8.3.2 Hyperparameters:

The hyper parameters for the standard ML algorithm are shown in Table 5.1., and we excluded the SVC model from this part of the experiment because it is not suitable with the enlarged ratings dataset specially after adding the negative samples instances.

The ANN and GNN hyper parameters are shown in Table 5.6. For all the models, the optimizer Adam was recommended. In FNN models, the recommended number of layers decreases when the number of features increase.

Table 5.7. Recommender systems hyperparameters

Model	**Hyperparameter**	
FNN	**Set #1**: num. layers = 2, embedding = 40, hidden features = 320, learning rate = 0.002, optimizer = Adam, batch size = 192, dropout = [0.2, 0.0]	**Set #2:** num. layers = 3, embedding = 30, hidden features = 555, learning rate = 0.002, optimizer = Adam, batch size = 288, dropout = [0.0,0.2, 0.0]
	Set #3: num. layers = 1, embedding = 30, hidden features = 570, learning rate = 0.003, optimizer = Adam, batch size = 288, dropout = [0.0,0.0, 0.3,0.2]	
GNN	num. layers = 2, embedding = 50, hidden features = 128, learning rate = 0.002, optimizer = Adam dropout=0	

5.8.3.3 Results:

In Table 5.7, the results of the different algorithms for recommendation were trained on three sets of features to predict the customer's next product to purchase.

Table 5.8. Recommender systems performance

Features	**Recommender System results in SET #1** Customers Id, product Id and product category, demographic, and subscription information.		**Training time in sec:**

Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**AVG P**	**Num. epochs x sec/epochs**
LR	0.80	0.50	0	0	0	0.20	
DT	0.84	0.80	0.58	0.74	0.65	0.48	
RF	0.84	0.80	0.56	0.72	0.65	0.48	
NB	0.80	0.50	0	0	0	0.20	
KNN	0.90	0.83	0.75	0.72	0.73	0.60	
Stacking Classifier	0.84	0.80	0.58	0.74	0.65	0.48	
ANN	0.75	0.73	0.72	0.92	0.81	0.71	9 epochs/11sec
GNN	NA	NA	NA	NA	NA	NA	NA
Features	**Recommender System results in SET #2** Customers Id, product Id and product category, demographic, and subscription information						**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**AVG P**	**Num. epochs x sec/epochs**
LR	0.80	0.5	0	0	0	0.20	
DT	0.84	0.81	0.58	0.76	0.66	0.49	
RF	0.69	0.61	0.32	0.47	0.38	0.26	
NB	0.80	0.50	0	0	0	0.20	
KNN	0.64	0.53	0.23	0.35	0.28	0.21	
Stacking Classifier	0.84	0.81	0.58	0.76	0.65	0.49	
ANN	0.78	0.76	0.76	0.90	0.83	0.74	10 epochs/10 sec
GNN	0.70	0.70	0.68	0.74	0.71	0.64	74/1.5 sec
Features	**Recommender System results in SET #3** Customers Id, product Id and product category, demographic, and subscription information, CLV cluster, RFM, purchasing gaps, page views duration, and feedback features.						**Training time in sec:**
Models	**Accuracy**	**AUC**	**Precision**	**Recall**	**F1**	**AVG P**	**Num. epochs x sec/epochs**
LR	0.69	0.55	0.27	0.32	0.29	0.22	
DT	0.84	0.81	0.58	0.76	0.66	0.49	

RF	0.65	0.57	0.27	0.43	0.33	0.23	
NB	0.62	0.57	0.26	0.49	0.34	0.23	
KNN	0.64	0.53	0.24	0.36	0.28	0.21	
Stacking Classifier	0.84	0.81	0.58	0.76	0.66	0.49	
ANN	0.57	0.50	0.57	1.00	0.72	0.57	20 epochs/9sec
GNN	0.70	0.70	0.69	0.74	0.71	0.64	115/1.3

5.8.3.4 Performance Comparison:

By comparing the models in the experimentations based on the F1 score we find that the FNN model achieve the best performance with the features in Set 3 with F1 score 83%. Then, the GNN model in the second order with Set 2 or Set 3, and achieved F1 score of 71%.

The recall score leads to the model's ability to predict positive labels which leads to the best products to recommend for a customer. The FNN achieved the highest recall scores among the rest of the models, which makes it the best algorithm for recommendation on CPDM.

From the results of the recommendation systems, we conclude the three outcomes:

- With GNN models, it is not possible to train the model based on the features of Set 1 without deleting the other nodes in the graph.
- While Set 1 contains only the IDs of the customer and their associated product, the results were adequate and led to useful predictions.
- We consider the features in Set 3 as the best information to feed the FNN to learn customer's preferences because it increases the model's ability to predict the positive labels.

5.9 Discussion:

Our experimentations had different feature sets and different ML algorithms to train for different tasks. With the three tasks, we found that the Sets with information from different sources increase the model's performance, especially with the FNN and GNN models. The features that the models learned were collected from customers' different touch points, including their demographic data, their subscription information, purchased products, their online actions, and their feedback.

In our models, we used the basic architecture of each task and focused on how to engineer different features and feed them to the models.

The best classifier is the GNN (node-level) model, which showed the highest performance based on predicting a good percentage of true labels with less computation cost. The best recommender system is the FNN model, which achieved results better than GNN (edge-level) model. The FNN model was slow in learning the data but achieved the best results. To explain, we fed the FNN model with duplicated data points to align the number of *full-prof* records to the number of records in the *transactions* dataset, which influenced the learning ability positively and slowed down the iteration because of the increased volume of the data.

In this chapter, we answered a three research questions:

Q2: What techniques can we apply to the data model features before feeding them to the predictive algorithms*?*

> We illustrated the different feature engineering we applied to the CPCM features and the embedding methods we adopted to feed the features to the predictive algorithms.

Q3: How to extract relational database and graph database from the customer profile data model?

We explained the entities in each database and the features of each entity. Furthermore, we compared the differences between them.

Q4: Is feeding all the features in the customer profile model achieve better prediction than one source of data for each task?

Through several experiments in this chapter, we found that the churn probability and next purchase time tasks in the GNN models achieved their best performance when they learned all customer features and their transactions in the graph data. Also, the SML and FNN models achieved their best performance when they learned all customer features in the *full-prof* dataset. The FNN recommendation system achieved better when excluding the features: CLV cluster, RFM, page views duration, purchasing gaps and feedback features.

In the next section, we propose a multi-task model which will be trained on the customer's information based on the features of CPDM to predict three tasks: Churn probability, Next purchase time, and next product recommendation.

Based on the results in our experimentations, we believe that using graph data and graph neural networks algorithms is a reasonable choice for the customer information in our CBPred for the following reasons:

- GNNs achieved the best performance for the first two tasks, and the second best in the third task.
- The performance of the FNN recommender system is better than the GNN recommender. However, the former is slow and has duplicated data points influencing the high prediction performance but slowing down the training.
- In our experimentation, the commutation cost for GNNs is less than FNN's models.

- The heterogeneous graph structure makes the data modeling and analysing more flexible than the relational datasets. For example, if a node does not have direct relation with the customer, we still can obtain information from these nodes using several hops (several layers for updating node's features) in the GNN model to update the customer's features.
- The graph structure can obtain one to one, and one to many relations in one dataset. For example, in our model the customer has a relation with the age node, and multiple relations with the product node.

CHAPTER 6 - THE PROPOSED MODEL

In this section, we propose a graph-based multi-task model CBPred to learn customer behaviour. The purpose of the proposed model is to use the customer information from different sources of data and analyze it to predict the different customer's intentions. The model learns the customer's information in the constructed graph data based on CPDM to learn their behaviour and predict their future actions.

We propose a multi-learning neural network based on graphs. The model contains several layers to update the customer's features, then process these features and predict the required labels.

6.1 Data

We use the graph dataset which we extracted from CPDM. The structure of the proposed graph is represented in section (5.3.2) and Figure 5.1. We will feed all nodes and the nodes' features to the predictive algorithms. The nodes and their features are representing the features in the CPDM: demographic information, subscription information, CLV cluster, RFM variables, page views duration, and feedback features, purchasing gaps, and product category.

6.2 Features embedding

Before feeding the nodes' features to the predictive model CBPred, we apply the following procedures:

1. the nodes' features are normalized to sum-up to one, and to achieve that, we use the Normalize function API from Pytorch Geometric library.
2. We randomly split the graph nodes by adding three sets of masks for training, validation and testing.
3. We randomly split the graph edges into three sets for train, validation, and test sets.

We feed the the nominal nodes' features to an embedding layer in the model which will update the values of the features according to their index in a look up table as described in section (5.5).

6.3 CBPred Model Architecture

The model is designed to perform heterogeneous graph representation learning the graph data of CPDM to predict three different tasks: churn probability, next purchase time and next item recommendation.

The churn probability and the next purchase time tasks are similar in analysing and learning the patterns in the updated representation of the customer nodes based on their neighbours in the graph.

The next purchase recommendation task is based on the rating the customer will likely give the products. However, we do not have an explicit rating in the data. We constructed the ratings based on whether the customer has bought a product or not. The graph shows only a positive edge between the customer and the products they purchased. We applied a random negative sampling to have negative edges to the products the customer had not purchased. The model analyzes and learn the negative and the positive edges and construct a new set of edges between the customer and the product nodes. In the new set of edges, edges with positive labels between the customer and product nodes that have not been purchased by the customer before should be recommended. To achieve the purpose of the model, CBPred as shown in Figure (6.1) consists of serval layers:

- **Embedding layer:** Embedding layers: neural network layers transform the categorical values into N-dimensional space representation to obtain essential properties for each value (Bressan 2020). In the model, the features of the nodes: gender, product id, city, state and product category were fed to embedding layers - the latent dimensions are fixed for all the categories.

- **Graph features Encoder (GraphSAGE 2 layers):** these layers aggregate and update each node's features based on their neighbours. The output of the two layers is graph embedding.
- **Churn task layer:** An FNN layer takes the updated representation of the customer's nodes and feeds it to a linear layer to train a binary classifier.
- **Next purchase time layer:** An FNN layer takes the updated representation of the customer's nodes and feeds it to a linear layer to train a multi-class classifier.
- **Negative sampling:** it is a function to randomly add edges between the customer and the new product nodes and create negative labels between the two nodes**.**
- **Edge Decoder:** contains a layer to concatenate the updated representation of the customer and the updated product nodes after the negative sampling and create a new set of edges between the two nodes with negative and positive edges.
- **Multi-task training loss:** to combine the losses of the three tasks in training, we applied the method explained in (Liebel and Körner 2018). The loss function adds unrelated assisting tasks to the existing tasks to improve the performance of the focused tasks to generalize the unseen data.
- **Output layer**: For churn probability and next purchase time table prediction.

6.4 Model hyperparameters

In the proposed model, the embedding dimension is set as 200, and the size of the updated graph features is set to 96. In the training process, we update the model parameters by the optimizer 'Adam' with a learning rate of 0.002. The model needs 20 epochs to learn the fed features, and each epoch takes 0.1 seconds.

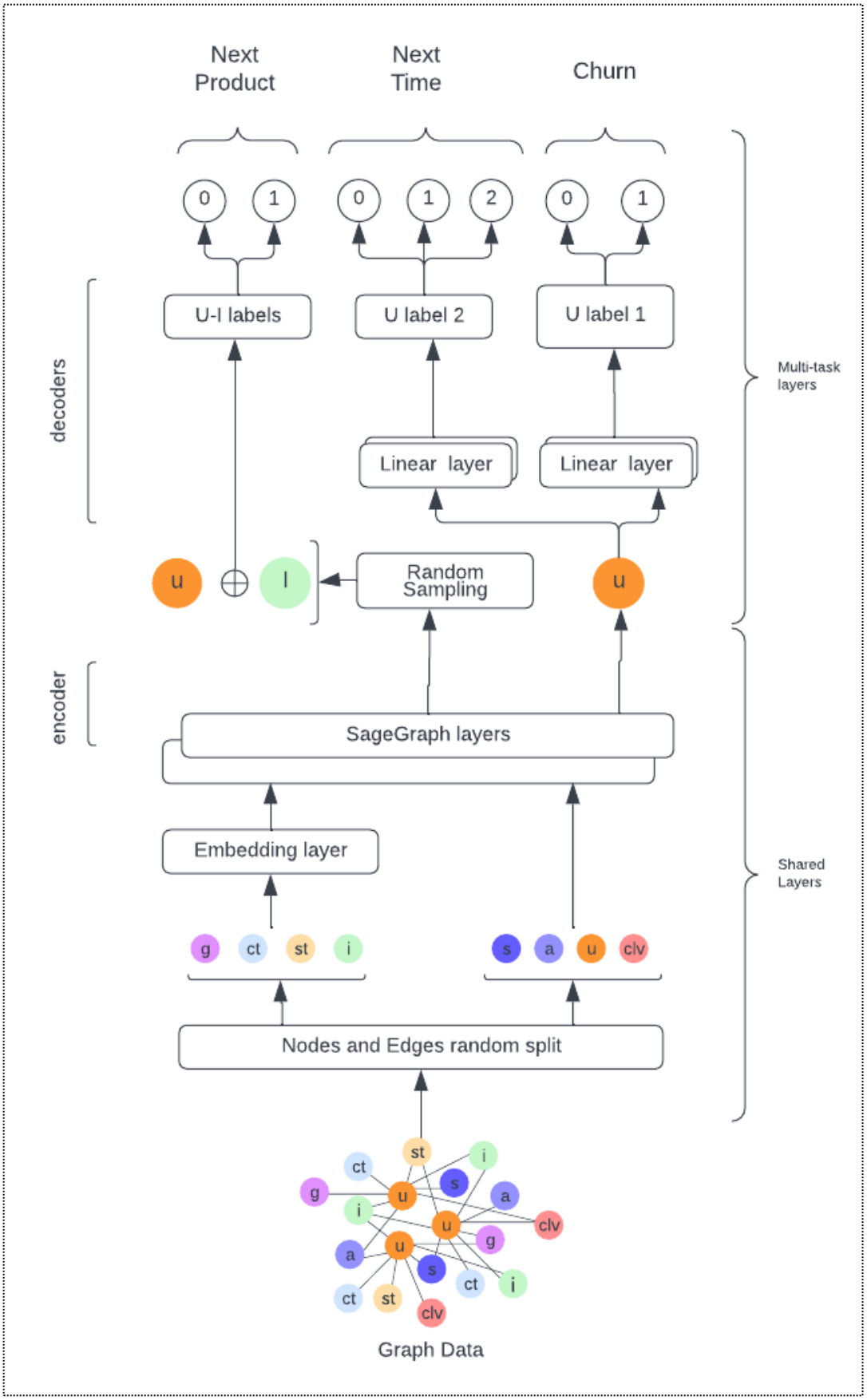

Figure 6.1. CBPred Model Architecture

Table 6.1. The performance of CBPred models for three predictive tasks

Tasks	Accuracy	AUC	Precision	Recall	F1	AVG P
Churn probability	0.93	0.92	0.93	0.91	0.91	0.90
Next purchase time	0.80	0.92	0.78	0.80	0.79	NA
Next product recommendation	0.70	0.70	0.70	0.74	0.71	0.64

6.5 Results

In this section, we represent and discuss the results of the multi-task model CBPred, which contains binary and multi-classifiers. The performances of the model are reported in Table 6.1.

6.6 Discussion

In training CBPred, the model needs 20 iterations, and each iteration consumes 0.1 seconds. We consider the computational cost of our model is low, as the time to learn the three tasks is equal to the time needed to learn each task separately. The results showed that 93% of the churn probability labels were predicted correctly. The correct predictions for the next purchase time and the next product recommendation based on rating were 80% and 70%, respectively.

The performance of CBPred model was slightly similar to the results of the same tasks when learned individually:

- **Churn rate prediction:** The task performance is decreased by around 2% in CBPred compared with the results of the churn probability model in section 5.8.1. However, the results for all the metrics are above 90%.
- **Next purchase time:** The results are similar to the model in section 5.8.2 with an increase in the recall metric by 1%, which reveals the ability to determine when the customer will purchase their next product.

- **Next product to be purchased:** The results are similar to the recommender system in section 5.8.3.

The performance of the proposed model reveals that the model capability to predict new edges for recommendation was the least compared with the other tasks. We assume that might because of two reasons:

1. The negative samples were constructed randomly, and it is possible for the random technique to construct a negative label between the customer and product that they might purchased. We consider that as a factor that might confuse the model to learn the customer preferences probably.
2. The products node does not have a descriptive information more than the categories.

The concept of the model is to process a customer's information and predict their future intentions. For example, if we feed the model with a new customer graph similar to the proposed graph, the model will predict the labels of each customer node and recommend a set of products. The model is capable of predicting three customers intentions with a performance range from 70% to 90%.

In this chapter, we answer the last research question:

Q5: How to construct a multi-task predictive model learn the customer features in CPDM?

We illustrated several entities in the graph dataset and how to manipulate the features before being fed to the GNN. The model's architecture was designed to transform the categorical features, extract graph representations, and achieve the focused tasks based on node-based and edge-based algorithms.

CHAPTER 7 - CONCLUSION AND FUTURE WORK

7.1 Conclusion

In the paper, we present a customer behaviour model based on graph embeddings to learn the intentions of the customers in the future. The research covers modelling the customer profile data, extracting different entities, forming different sets of features, and using several machine learning algorithms for three tasks. The results showed that the data the customer creates with different touch points should be considered in analyzing customers' behaviour and predicting their intentions. In other words, the customer's behaviour is influenced by their demographic information, feedback on the business through different channels, the time they spent on the business website, the history of their purchases, and their subscription types.

The graph data on CPDM guarantees to find the customer's relation in case there are no direct relations through defining nodes from the demographic information for the customers. For example, customers of the same age or living in the same city will share their features and their neighbours' features.

The proposed graph-based model can analyze and embed the updated feature in a low-dimensional space of different sources of information for the customer and their neighbours. Graph embedding is an encoded feature that can be decoded for different tasks. We decoded the graph representation to obtain the labels of the customer nodes and the edges between the customer and product nodes for classification tasks and a recommendation system, respectively.

The proposed model's performance reveals that the capability to predict new edges for the recommendation was the least compared with the other tasks. However, we believe that the model flexibility and the low computation cost compensate for that drawback.

We demonstrate that we can use customer information from different sources to enhance task performance, as in the experimentation in Chapter 5, especially with neural networks. Our research examined how to analyze customer behaviour using several types of features from different data sources instead of relying on specific features. After the experimentation, we found that the predictive model based on graph data structure and GNN are the most efficient models for analyzing our customer profile features.

7.2 Future Work:

Many different tests, and experiments have been left for the future due to lack of time, and we discuss it the following point:

- The proposed CPDM does not include the social media records for the customer rather than their feedback through Twitter. We believe that if the data contains the relation between the customers via social media platforms, that may link the customers and their inputs in the social platforms and create a more robust model.
- The proposed model does not include enough information about the products, such as product description and product brand. We believe that including more informative product features could enhance the recommendation task.
- The proposed CBPred multitask model is constructed based on the basic architecture for each task without adding any mechanisms to enhance the performance rather than tweaking the hyperparameters. This is one of the model's advantages at some point, especially since the results of the two classification tasks were around 80% and 90% of the F1 score using a simple system design. However, the recommender task achieved the lowest results of 71%. We believe that to enhance the edge prediction task for the recommendation, a more intelligent technique should be used to produce negative

samples of the edges between the customer and the product nodes rather than the basic random method we applied in the model.

- We realized that our multitask model outcomes have not changed from the individual model in the experimentation in Chapter 5. This could be for one or two reasons, firstly, the tasks are not related to each other, and there is nothing for each task to learn from the other two tasks. Secondly, weighting the loss for each task might not be suitable for the model and needs a more sophisticated loss function.

- The proposed model can be extended in both data and tasks. In other words, the graph can be upgraded with new nodes with relations with the customer nodes, new features for customer nodes, and new tasks. The customer's node can have more features, such as personal interest, personality type, and occupation. Also, product nodes should include more informative features, such as the product's brand and description. Social media nodes can be added to find more relations between the customers. The model's tasks can be extended to predict more intentions for the customers, such as classifying the next product (category, brand, price).

References

Abirami, S., and P. Chitra. 2020. “Energy-Efficient Edge Based Real-Time Healthcare Support System.” In *Advances in Computers*, 339–68. Advances in Computers. Elsevier.

Ahmad, Abdelrahim Kasem, Assef Jafar, and Kadan Aljoumaa. 2019. “Customer Churn Prediction in Telecom Using Machine Learning in Big Data Platform.” *Journal of Big Data* 6 (1): 28.

Bhandari, Aniruddha. 2020. “AUC-ROC Curve in Machine Learning Clearly Explained.” June 15, 2020. https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/.

Blokdyk, Gerardus. 2018. *IBM Docs: Complete Self-Assessment Guide*. North Charleston, SC: Createspace Independent Publishing Platform.

Bonometti, V., C. Ringer, M. Hall, A. R. Wade, and A. Drachen. 2019. “Modeling Early User-Game Interactions for Joint Estimation of Survival Time and Churn Probability.” In *2019 IEEE Conference on Games (CoG)*, 1–8.

Bressan, Rodrigo. 2020. “Enhancing Categorical Features with Entity Embeddings.” Towards Data Science. February 10, 2020. https://towardsdatascience.com/enhancing-categorical-features-with-entity-embeddings-e6850a5e34ff.

Chaudhary, A., Kolhe, S., & Kamal, R. (2016). An improved random forest classifier for multi-class classification. *Information Processing in Agriculture*, *3*(4), 215–222. https://doi.org/10.1016/J.INPA.2016.08.002

Ceballos, Frank. 2019. “Stacking Classifiers for Higher Predictive Performance.” Towards Data Science. July 20, 2019. https://towardsdatascience.com/stacking-classifiers-for-higher-predictive-performance-566f963e4840.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. “SMOTE: Synthetic Minority Over-Sampling Technique.” *The Journal of Artificial Intelligence Research* 16

(June): 321–57.

Cole, C. A. 2007. “Consumer Behavior.” *Encyclopedia of Gerontology*. https://doi.org/10.1016/b0-12-370870-2/00040-8.

De Marco, Marco, Paolo Fantozzi, Claudio Fornaro, Luigi Laura, and Antonio Miloso. 2021. “Cognitive Analytics Management of the Customer Lifetime Value: An Artificial Neural Network Approach.” *Journal of Enterprise Information Management* 34 (2): 679–96.

Droomer, M., and J. Bekker. 2020. “Using Machine Learning to Predict the next Purchase Date for an Individual Retail Customer.” *South African Journal of Industrial Engineering*. http://www.scielo.org.za/scielo.php?pid=S2224-78902020000300008&script=sci_arttext&tlng=es.

Duchesnay, F. P. and G. V. and A. G. and V. M. and B. T. and O. G. and M. B. and P. P. and R. W. and V. D. and J. V. and A. P. and D. C. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830. Retrieved from http://jmlr.org/papers/v12/pedregosa11a.html

Fan, Yujie, Yanfang Ye, Qian Peng, Jianfei Zhang, Yiming Zhang, Xusheng Xiao, Chuan Shi, Qi Xiong, Fudong Shao, and Liang Zhao. 2020. “Metagraph Aggregated Heterogeneous Graph Neural Network for Illicit Traded Product Identification in Underground Market.” In *2020 IEEE International Conference on Data Mining (ICDM)*, 132–41.

Fey, Matthias, and Jan Eric Lenssen. 2019. “Fast Graph Representation Learning with PyTorch Geometric.” *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1903.02428.

Guo, Long, Lifeng Hua, Rongfei Jia, Binqiang Zhao, Xiaobo Wang, and Bin Cui. 2019. “Buying or Browsing?: Predicting Real-Time Purchasing Intent Using Attention-Based Deep Network with Multiple Behavior.” In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1984–92. KDD ’19.

New York, NY, USA: Association for Computing Machinery.

Hamilton, William L. 2020. “Graph Representation Learning.” *Synthesis Lectures on Artificial Intelligence and Machine Learning* 14 (3): 1–159.

Hamilton, Will, Zhitao Ying, and Jure Leskovec. 2017. “Inductive Representation Learning on Large Graphs.” In *Advances in Neural Information Processing Systems*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf.

Hang, Lee Carmen Kar, Tse Ying Kei, Minhao Zhang, and Jie Ma. 2019. “Analyzing Online Reviews to Investigate Customer Behaviour in the Sharing Economy: The Case of Airbnb.” *Information Technology & People* 33 (3): 945–61.

Hiriyannaiah, Srinidhi, A. M. D. Srinivas, Gagan K. Shetty, Siddesh, and K. G. Srinivasa. 2020. “A Computationally Intelligent Agent for Detecting Fake News Using Generative Adversarial Networks.” In *Hybrid Computational Intelligence*, 69–96. Elsevier.

Holmlund, Maria, Yves Van Vaerenbergh, Robert Ciuchita, Annika Ravald, Panagiotis Sarantopoulos, Francisco Villarroel Ordenes, and Mohamed Zaki. 2020. “Customer Experience Management in the Age of Big Data Analytics: A Strategic Framework.” *Journal of Business Research* 116 (August): 356–65.

Ho, Teck-Hua, Young-Hoon Park, and Yong-Pin Zhou. 2006. “Incorporating Satisfaction into Customer Value Analysis: Optimal Investment in Lifetime Value.” *Marketing Science* 25 (3): 260–77.

Huang, Chao, Xian Wu, Xuchao Zhang, Chuxu Zhang, Jiashu Zhao, Dawei Yin, and Nitesh V. Chawla. 2019. “Online Purchase Prediction via Multi-Scale Modeling of Behavior Dynamics.” In *Proceedings of the 25th ACM SIGKDD International Conference on*

Knowledge Discovery & Data Mining, 2613–22. KDD '19. New York, NY, USA: Association for Computing Machinery.

HUSPI. 2020. "What Is User Modeling and How to Use User Modeling for Business." HUSPI. June 4, 2020. https://huspi.com/blog-open/what-is-user-modeling-and-how-to-use-it-to-sell-more-products/.

Indhu. 2021. "How 360-Degree Customer View Helps Your Business?" Data Science Central. February 7, 2021. https://www.datasciencecentral.com/how-360-degree-customer-view-helps-your-business-4/.

Karaman, Barış. 2019. "Predicting Next Purchase Day - Towards Data Science." Towards Data Science. June 2, 2019. https://towardsdatascience.com/predicting-next-purchase-day-15fae5548027.

Karim, Raimi. 2018. "10 Stochastic Gradient Descent Optimisation Algorithms + Cheatsheet." Towards Data Science. November 22, 2018. https://towardsdatascience.com/10-gradient-descent-optimisation-algorithms-86989510b5e9.

Karnstedt, Marcel, Matthew Rowe, Jeffrey Chan, Harith Alani, and Conor Hayes. 2011. "The Effect of User Features on Churn in Social Networks." In *Proceedings of the 3rd International Web Science Conference*, 1–8. WebSci '11 23. New York, NY, USA: Association for Computing Machinery.

Keras-Tuner: Hyperparameter Tuning for Humans. n.d. Github. Accessed September 2, 2021. https://github.com/keras-team/keras-tuner.

Kim, Hee-Su, and Choong-Han Yoon. 2004. "Determinants of Subscriber Churn and Customer Loyalty in the Korean Mobile Telephony Market." *Telecommunications Policy* 28 (9): 751–65.

Koehn, Dennis, Stefan Lessmann, and Markus Schaal. 2020. "Predicting Online Shopping

Behaviour from Clickstream Data Using Deep Learning." *Expert Systems with Applications* 150 (July): 113342.

Kotler, Philip, and Kevin Lane Keller. 2016. *Marketing Management*. Pearson.

Kung-Hsiang, Huang (Steeve), and Huang (steeve) Kung-Hsiang. 2019. "Hands-on Graph Neural Networks with PyTorch & PyTorch Geometric." Towards Data Science. May 30, 2019. https://towardsdatascience.com/hands-on-graph-neural-networks-with-pytorch-pytorch-geometric-359487e221a8.

Kumar Thakkar, H., Desai, A., Ghosh, S., Singh, P., & Sharma, G. (2022). *Clairvoyant: AdaBoost with Cost-Enabled Cost-Sensitive Classifier for Customer Churn Prediction*. https://doi.org/10.1155/2022/9028580

Liang, Dawen, Jaan Altosaar, Laurent Charlin, and David M. Blei. 2016. "Factorization Meets the Item Embedding: Regularizing Matrix Factorization with Item Co-Occurrence." In *Proceedings of the 10th ACM Conference on Recommender Systems*, 59–66. RecSys '16. New York, NY, USA: Association for Computing Machinery.

Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., & Ma, J. (2017). *Neural AAentive Session-based Recommendation*. https://doi.org/10.1145/3132847.3132926

Liebel, Lukas, and Marco Körner. 2018. "Auxiliary Tasks in Multi-Task Learning." *arXiv [cs.CV]*. arXiv. http://arxiv.org/abs/1805.06334.

Lim, Nicholas, Bryan Hooi, See-Kiong Ng, Xueou Wang, Yong Liang Goh, Renrong Weng, and Jagannadan Varadarajan. 2020. "STP-UDGAT: Spatial-Temporal-Preference User Dimensional Graph Attention Network for Next POI Recommendation." In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 845–54. New York, NY, USA: Association for Computing Machinery.

Liu, Bang, Hanlin Zhang, Linglong Kong, and Di Niu. 2021. "Factorizing Historical User Actions for Next-Day Purchase Prediction." *ACM Trans. Web*, 1, 16 (1): 1–26.

"Logistic Regression." n.d. Accessed April 28, 2022. https://www.ibm.com/topics/logistic-regression.

Manjupriya, R., and A. Poornima. 2018. "Customer Churn Prediction in the Mobile Telecommunication Industry Using Decision Tree Classification Algorithm." *Journal of Computational and Theoretical Nanoscience* 15 (9-10): 2789–93.

Martínez, Andrés, Claudia Schmuck, Sergiy Pereverzyev, Clemens Pirker, and Markus Haltmeier. 2020. "A Machine Learning Framework for Customer Purchase Prediction in the Non-Contractual Setting." *European Journal of Operational Research* 281 (3): 588–96.

Menzli, Amal. 2020. "Graph Neural Network and Some of GNN Applications: Everything You Need to Know." Neptune.ai. November 30, 2020. https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications.

Meyer, Christopher, and Andre Schwager. 2007. "Understanding Customer Experience." *Harvard Business Review* 85 (2): 116–26, 157.

Microsoft Corporation. n.d. "Customer Data Platform." Accessed June 7, 2022. https://dynamics.microsoft.com/en-us/ai/customer-insights/.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., & Grohe, M. (2018). Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks. *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, 4602–4609. https://doi.org/10.48550/arxiv.1810.02244

Mosaddegh, Abdolreza, Amir Albadvi, Mohammad Mehdi Sepehri, and Babak Teimourpour. 2021. "Dynamics of Customer Segments: A Predictor of Customer Lifetime Value." *Expert Systems with Applications* 172 (June): 114606.

Optuna: A Hyperparameter Optimization Framework. n.d. Github. Accessed June 2, 2022. https://github.com/optuna/optuna.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research: JMLR* 12 (85): 2825–30.

Peng, Zhaoqing, Junqi Jin, Lan Luo, Yaodong Yang, Rui Luo, Jun Wang, Weinan Zhang, et al. 2020. "Learning to Infer User Hidden States for Online Sequential Advertising." In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2677–84. New York, NY, USA: Association for Computing Machinery.

Peng, Z., Jin, J., Luo, L., Yang, Y., Luo, R., Wang, J., … Gai, K. (2020). Learning to Infer User Hidden States for Online Sequential Advertising. *International Conference on Information and Knowledge Management, Proceedings*, *11*(20), 2677–2684. https://doi.org/10.1145/3340531.3412721

Pines, Zak. 2018. "Clear These 6 Data Hurdles to Achieve a 360-Degree Customer View." CMSWire.com. January 17, 2018. https://www.cmswire.com/customer-experience/clear-these-6-data-hurdles-to-achieve-a-360-degree-customer-view/.

Popescu, Cristian-Constantin. 2018. "Improvements in Business Operations and Customer Experience through Data Science and Artificial Intelligence." *Proceedings of the International Conference on Business Excellence* 12 (1): 804–15.

Pore, Prasad. n.d. "Must-Know: How to Evaluate a Binary Classifier." Accessed September 2, 2021. https://www.kdnuggets.com/2017/04/must-know-evaluate-binary-classifier.html.

"Random Forest Algorithm." n.d. Www.javatpoint.com. Accessed April 28, 2022. https://www.javatpoint.com/machine-learning-random-forest-algorithm.

Rençberoğlu, Emre. 2019. "Fundamental Techniques of Feature Engineering for Machine Learning." Towards Data Science. April 1, 2019. https://towardsdatascience.com/feature-

engineering-for-machine-learning-3a5e293a5114.

Sayed, Hend, Manal A. Abdel-Fattah, and Sherif Kholief. 2018. "Predicting Potential Banking Customer Churn Using Apache Spark ML and MLlib Packages: A Comparative Study." *IJACSA) International Journal of Advanced Computer Science and Applications* 9 (11). https://www.researchgate.net/profile/Manal-Abdel-Fattah-2/publication/329427276_Predicting_Potential_Banking_Customer_Churn_using_Apache_Spark_ML_and_MLlib_Packages_A_Comparative_Study/links/5c08086f4585157ac1aaf58e/Predicting-Potential-Banking-Customer-Churn-using-Apache-Spark-ML-and-MLlib-Packages-A-Comparative-Study.pdf.

Sedhain, Suvash, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. "AutoRec: Autoencoders Meet Collaborative Filtering." In *Proceedings of the 24th International Conference on World Wide Web*, 111–12. WWW '15 Companion. New York, NY, USA: Association for Computing Machinery.

Segal, Troy. 2021. "Recency, Frequency, Monetary Value (RFM)." September 13, 2021. https://www.investopedia.com/terms/r/rfm-recency-frequency-monetary-value.asp.

Shawe-Taylor, J., & Sun, S. (2011). A review of optimization methodologies in support vector machines. *Neurocomputing*, *74*(17), 3609–3618. https://doi.org/10.1016/J.NEUCOM.2011.06.026

Sheil, H., Rana, O., & Reilly, R. (2018). Predicting purchasing intent: Automatic Feature Learning using Recurrent Neural Networks. *CEUR Workshop Proceedings*, *2319*. Retrieved from http://arxiv.org/abs/1807.08207

Swain, P. H., & Hauska, H. (1977). The decision tree classifier: Design and potential; The decision tree classifier: Design and potential. In *IEEE Transactions on Geoscience Electronics* (Vol. 15). https://doi.org/10.1109/TGE.1977.6498972

"Sklearn.Cluster.KMeans." n.d. Scikit-Learn. Accessed June 7, 2022. https://scikit-

learn.org/stable/modules/generated/sklearn.cluster.KMeans.html.

"Sklearn.Metrics.F1_score." n.d. Scikit-Learn. Accessed April 26, 2022a. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html?highlight=f1%20score.

———. n.d. Scikit-Learn. Accessed May 27, 2022b. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html.

"Sklearn.Metrics.Precision_score." n.d. Scikit-Learn. Accessed April 26, 2022. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html?highlight=precision.

"Sklearn.Metrics.Recall_score." n.d. Scikit-Learn. Accessed April 26, 2022. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html.

"Sklearn.Preprocessing.LabelEncoder." n.d. Scikit-Learn. Accessed April 27, 2022. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html.

"Sklearn.Preprocessing.StandardScaler." n.d. Scikit-Learn. Accessed June 7, 2022. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.

"SMOTE — Version 0.8.0." n.d. Accessed September 2, 2021. https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html.

Sokolova, Marina, Nathalie Japkowicz, and Stan Szpakowicz. 2006. "Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation." In *AI 2006: Advances in Artificial Intelligence*, 1015–21. Springer Berlin Heidelberg.

Stein, Alisha, and B. Ramaseshan. 2019. "The Customer Experience--Loyalty Link: Moderating Role of Motivation Orientation." *Journal of Service Management*. https://www.emerald.com/insight/content/doi/10.1108/JOSM-04-2019-0113/full/html?casa_token=VK0ICzOVCjoAAAAA:ADgq6KMnMxe7QizclGlWvFQhe

5puRMigxLJzXRcqEqc5hlRfR24WpPAz46gGfHo_5Xu4bVyFEcr98_G0pFFTj_WCKlfKCN5oE_EwjG-QfT070XUrtVfC.

Sundararaj, Vinu, and M. R. Rejeesh. 2021. "A Detailed Behavioral Analysis on Consumer and Customer Changing Behavior with Respect to Social Networking Sites." *Journal of Retailing and Consumer Services* 58 (January): 102190.

Tamaddoni, Ali, Stanislav Stakhovych, and Michael Ewing. 2016. "Comparing Churn Prediction Techniques and Assessing Their Performance: A Contingent Perspective." *Journal of Service Research* 19 (2): 123–41.

Tang, Jiaxi, and Ke Wang. 2018. "Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding." In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 565–73. WSDM '18. New York, NY, USA: Association for Computing Machinery.

Tang, J., and K. Wang. 2018. "Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding." *Proceedings of the Eleventh ACM International*. https://dl.acm.org/doi/abs/10.1145/3159652.3159656?casa_token=qziGZBodtZMAAAAA:EQAMsWJUNZlL3uxCDm0-ZzimecG0n3mS612SG07xK1FodYCSIh2QZdsNhWzn2_YiyKVJDL2xOvGTAA.

Tan, Qiaoyu, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. 2021. "Sparse-Interest Network for Sequential Recommendation." In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 598–606. WSDM '21. New York, NY, USA: Association for Computing Machinery.

Teixeira, Jorge, Lia Patrício, Nuno J. Nunes, Leonel Nóbrega, Raymond P. Fisk, and Larry Constantine. 2012. "Customer Experience Modeling: From Customer Experience to Service Design." *Journal of Service Management* 23 (3): 362–76.

Tripathi, A., Yadav, S., & Rajan, R. (2019). Naive Bayes Classification Model for the Student Performance Prediction. *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2019*, 1548–1553. https://doi.org/10.1109/ICICICT46008.2019.8993237

Venkatesh, S., Moffat, D., Reck, E., & ‡1, M. (2022). *Word Embeddings for Automatic Equalization in Audio Mixing*.

Wang, Peng, Jiang Xu, Chunyi Liu, Hao Feng, Zang Li, and Jieping Ye. 2020. "Masked-Field Pre-Training for User Intent Prediction." In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2789–96. New York, NY, USA: Association for Computing Machinery.

Wang, Pengyang, Yanjie Fu, Hui Xiong, and Xiaolin Li. 2019. "Adversarial Substructured Representation Learning for Mobile User Profiling." In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 130–38. KDD '19. New York, NY, USA: Association for Computing Machinery.

Wang, Pengyang, Kunpeng Liu, Lu Jiang, Xiaolin Li, and Yanjie Fu. 2020. "Incremental Mobile User Profiling: Reinforcement Learning with Spatial Knowledge Graph for Modeling Event Streams." In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 853–61. KDD '20. New York, NY, USA: Association for Computing Machinery.

Wang, Xiang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. "KGAT: Knowledge Graph Attention Network for Recommendation." In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 950–58. KDD '19. New York, NY, USA: Association for Computing Machinery.

"What Is a Support Vector Machine, and Why Would I Use It?" n.d. KDnuggets. Accessed April 28, 2022. https://www.kdnuggets.com/2017/02/yhat-support-vector-machine.html.

"What Is Customer Lifetime Value (CLV) and How to Measure It?" 2021. March 26, 2021. https://www.qualtrics.com/experience-management/customer/customer-lifetime-value/.

Wang, X., Ji, H., Shi, C., Wang, B., Cui, P., Yu, P., & Ye, Y. (2019). Heterogeneous Graph Attention Network. *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, 2022–2032. Retrieved from https://arxiv.org/abs/1903.07293v2

Wang, Q. Q., Yu, S. C., Qi, X., Hu, Y. H., Zheng, W. J., Shi, J. X., & Yao, H. Y. (2019). [Overview of logistic regression model analysis and application]. *Zhonghua Yu Fang Yi Xue Za Zhi [Chinese Journal of Preventive Medicine]*, *53*(9), 955–960. https://doi.org/10.3760/CMA.J.ISSN.0253-9624.2019.09.018

Wong, Eugene, and Yan Wei. 2018. "Customer Online Shopping Experience Data Analytics: Integrated Customer Segmentation and Customised Services Prediction Model." *International Journal of Retail & Distribution Management* 46 (4): 406–20.

Wyner, G. A. (2001). When 360 Degrees Is Not Enough. *Marketing Management*, *10*(2), 4. Retrieved from https://web-s-ebscohost-com.proxy.library.carleton.ca/ehost/pdfviewer/pdfviewer?vid=0&sid=3c00c93a-d2fd-4c15-9996-070772039d0e%40redis

Yang, Bo, Yu Lei, Jiming Liu, and Wenjie Li. 2017. "Social Collaborative Filtering by Trust." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (8): 1633–47.

Zhai, X., Shi, P., Xu, L., Wang, Y., & Chen, X. (2020). Prediction Model of User Purchase Behavior Based on Machine Learning; Prediction Model of User Purchase Behavior Based on Machine Learning. In *2020 IEEE International Conference on Mechatronics and Automation (ICMA)*. https://doi.org/10.1109/ICMA49215.2020.9233677

Zheng, Yu, Chen Gao, Xiangnan He, Yong Li, and Depeng Jin. 2020. "Price-Aware Recommendation with Graph Convolutional Networks." In *2020 IEEE 36th International*

Conference on Data Engineering (ICDE), 133–44.

Zhu, Ziwei, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. 2021. "Popularity-Opportunity Bias in Collaborative Filtering." In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 85–93. WSDM '21. New York, NY, USA: Association for Computing Machinery.