박사학위논문
Ph.D. Dissertation

매장 내 센서 데이터를 활용한 고객 재방문 예측

Customer Revisit Prediction Using In-Store Sensor Data

2019

김선동 (金先東 Kim, Sundong)

한국과학기술원

Korea Advanced Institute of Science and Technology

박사학위논문

매장 내 센서 데이터를 활용한 고객 재방문 예측

2019

김선동

한국과학기술원

산업 및 시스템 공학과 (지식서비스공학대학원)

매장 내 센서 데이터를 활용한 고객 재방문 예측

김선동

위 논문은 한국과학기술원 박사학위논문으로
학위논문 심사위원회의 심사를 통과하였음

2019년 05월 16일

심사위원장 이재길 (인)

심사위원 이문용 (인)

심사위원 김경국 (인)

심사위원 장영재 (인)

심사위원 차미영 (인)

Customer Revisit Prediction Using In-Store Sensor Data

Sundong Kim

Advisor: Jae-Gil Lee

A dissertation submitted to the faculty of
Korea Advanced Institute of Science and Technology in
partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Industrial and Systems Engineering (Knowledge Service Engineering)

Daejeon, Korea
May 16, 2019

Approved by

Jae-Gil Lee
Professor of Industrial and Systems Engineering

The study was conducted in accordance with Code of Research Ethics[1].

[1] Declaration of Ethical Conduct in Research: I, as a graduate student of Korea Advanced Institute of Science and Technology, hereby declare that I have not committed any act that may damage the credibility of my research. This includes, but is not limited to, falsification, thesis written by someone else, distortion of research findings, and plagiarism. I confirm that my thesis contains honest conclusions based on my own careful research under the guidance of my advisor.

DKSE
20155042

김선동. 매장 내 센서 데이터를 활용한 고객 재방문 예측. 산업 및 시스템 공학과 (지식서비스공학대학원) . 2019년. 96+iv 쪽. 지도교수: 이재길. (영문 논문)
Sundong Kim. Customer Revisit Prediction Using In-Store Sensor Data. Department of Industrial and Systems Engineering (Graduate School of Knowledge Service Engineering) . 2019. 96+iv pages. Advisor: Jae-Gil Lee. (Text in English)

초록

센서 기술의 발전으로 오프라인 환경에서 대량의 고객 데이터 수집이 이루어지고 있다. 수집된 데이터를 기반으로 다방면의 분석 결과를 제공하는 솔루션은 운영하는 매장의 지표들 모니터링을 가능케 하였고, 관리자들은 정량적인 분석을 통해 타깃 마케팅, 매대 배치 변경 등 만족스러운 고객 경험을 위한 조치를 취할 수 있게 되었다. 이러한 노력의 궁극적인 목표는 지속적인 수익 창출인데, 이를 위해서는 고객의 잠재적 가치를 높일 수 있는 재방문을 끌어내는 것이 매우 중요하다.

본 학위 논문에서는 매장 내부에서 수집된 센서 데이터를 활용한 고객의 재방문 예측 (Revisit prediction)의 중요성을 설명하고 두 가지 예측 모델링 기법을 제시한다. 재방문이란 지표를 잘 예측하게 되면 상점 관리자는 고객의 방문 패턴을 파악하여 예상 수익을 간접적으로 측정할 수 있다. 또한 고객의 재방문 의도를 알면 고객군별 타깃 마케팅을 활용할 수 있다. 타깃 마케팅의 예로, 단골에게는 상위 브랜드를 추천하여 다양한 경험을 제공하는 동시에, 재방문 의지가 낮은 고객에게는 현재 방문 안에 대량 구매를 유도하거나 공격적인 할인 정책을 제공함으로써 고객의 재방문을 유도하고 객단가를 높이는 효과를 얻을 수 있다.

재방문 예측을 위해 매장 내부에서 수집된 센서 데이터 (In-store sensor data)를 활용하였는데 이는 쇼핑할 때에 발생하는 고객의 이동 경로를 활용하기 위함이다. 매장 안에서 발생하는 데이터만 수집할 수 있다는 조건하에, 매장 내부에서의 이동 경로는 와이파이 핑거프린팅 기술이 적용된 센서를 매장 곳곳에 설치하는 방법으로 비교적 쉽게 얻을 수 있기 때문이다. 마찬가지로, 재방문이라는 지표 역시 기기의 고유 ID 값을 바탕으로 확인할 수 있다. 이외에도 고객의 재방문과 관련이 있는 특성들로는 신상 정보나 주로 방문하는 장소들, 기 방문에서 얻을 수 있는 구매 정보 등이 있지만, 복합적인 고객 관리 시스템이 존재하거나 애플리케이션 등을 통해 고객의 위치 정보를 확보한 경우만 한정적으로 입수할 수 있다.

본 논문의 첫 번째 파트에서는 센서들로부터 얻어진 데이터만으로 고객의 재방문을 결정짓는 다양한 특성을 디자인하였고 (Feature engineering), 이러한 특성들을 적용한 기계 학습 모델이 그렇지 않은 모델에 비해 재방문 예측에 4.7–24.3%만큼 효과적임을 입증하였다. 특히 방문 횟수가 적어서 예측이 힘들었던 고객군에서 이러한 특성들이 재방문 예측에 매우 효과적임을 밝혔다. 이외에도 설계한 특성들의 설명과 함께 각 특성 그룹별 예측력을 살펴보았으며, 고객의 데이터 대부분이 누락되는 상황에서도 재방문 예측 모델의 성능이 유지됨을 실험적으로 보였다. 또한 데이터 수집 기간의 변화에 따른 모델의 성능과 센서 데이터를 활용할 때 주의해야 할 점을 고찰하였다. 이 파트에서 소개한 특성 모델링 기법부터 다양한 실험 세팅 및 결과 분석론까지의 일련의 프로세스들은 다양한 예측 문제에도 적용될 수 있다.

본 논문의 두 번째 파트에서는 딥 러닝 (Deep learning)과 생존 분석 (Survival analysis) 방법을 결합하여 부분적으로만 관측된 고객 데이터를 놓치지 않고 활용하는 방안을 제안한다. 고객의 방문 횟수가 적은 경우, 부분 관측 데이터 (Partial observations)가 필연적으로 발생할 수밖에 없는데, 부분 관측 데이터의 경우 재방문 간격 정보가 존재하지 않아 회귀분석을 활용한 기존 기계 학습 모델에서 활용하기에 어려움이 있다. 생존 분석 기법을 활용하면 부분 관측 데이터를 활용할 수 있지만, 고객의 매장 방문 간격과 널리 알려진 분포는 확연히 다르기 때문에 생존 분석 기법을 적용하기 위한 기본적인 가정들을 무시하게 된다. 분포를 가정하지 않으면서, 고객의 방문 간격을 보다 정교하게 학습하기 위해 이산적 재방문율 (Quantized revisit rate)을 출력하는 딥 러닝 모델을 제안하였다. 제시하는 *SurvRev* 프레임워크는 딥 러닝 모델과 생존 분석 모델의 조합을 통해 각 방문에 대해 고객의 방문 이후 365 일간의 재방문율을 예측할 수 있는 모델이다.

재방문 예측의 다양한 지표들을 보다 잘 반영하기 위하여 *SurvRev* 모델은 다양한 손실 함수를 최적화한다. 또한 실험 결과를 통해 *SurvRev* 모델이 기존 방법론들에 비해 우수함을 입증하였다. 이 파트에서는 데이터 마이닝 문제에서 생기는 중요한 이슈를 모델의 개선을 통해 해결하는 방법을 서술하고자 하였다.

재방문 예측 모델의 적용을 위해 우리는 서울 도심에 위치한 7개 주요 매장에서 2.5년 간 570만 건 이상의 실내 이동 패턴 데이터를 수집하였고, 일부를 정제하여 벤치마크 데이터 세트로 공개하였다. 본 연구 및 데이터가 고객의 쇼핑 패턴을 탐구하는 다양한 후속 연구들에 활용되길 바란다.

핵심낱말 재방문 예측, 고객 행동 예측, 예측 분석, 소매 분석, 매장 내에서 수집된 발자취 분석, 유저 모델링, 센서 데이터, 움직임 데이터, 특성 추출, 데이터 마이닝, 기계 학습, 딥 러닝, 생존 분석, 종적 데이터

Abstract

With the advancement of sensor technology, offline data collection has become possible, and many retail analytics companies are beginning to offer solutions that provide data collection and analysis. Thereby, store managers can grasp the status of their stores, thus trying to satisfy the customers' experience. Many of these efforts are carried out in order to secure regular customers for continuous store management and profit generation. To get closer to customers, companies strive to understand customers' interest and profile. Furthermore, they make an effort to predict customers' potential lifetime values, purchasing patterns, revisits, and stickiness. Among these objectives, *customer revisit* is a *feasible* and *valuable* metric to study since it can be recognized by only using customer *foot-traffic* data. This is very important to note since purchase data and user profiles are considered as proprietary information and difficult to obtain outside the company, but *customer mobility* becomes relatively easy to obtain through location monitoring technology once we get the customers' permission through their mobile device.

By knowing customers' visitation pattern, store managers can indirectly gauge the *expected revenue. Targeted marketing* can also be available by knowing customers' revisit intention. By offering discount coupons, merchants can encourage customers to accidentally revisit a store nearby. Also, they can offer a sister brand with finer products to provide new shopping experiences to loyal customers. In this way, they can *increase* the revenue as well as *satisfy* their customers. My thesis focuses on these closely related questions—revisit prediction—to capture the potential regular customers of the store. To achieve the goal, we formally design predictive analytics and develop two *frameworks* using mobility data captured from in-store sensors.

In the first part, we introduce a traditional machine learning model with carefully designed *handcrafted features.* We design extensive handcrafted features using semantic areas of the stores, and we investigate the predictive powers of feature groups and semantic levels of areas. We confirm the *effectiveness* of considering customer mobility by showing the performance improvement of 4.7–24.3%. Furthermore, we provide an in-depth analysis regarding the effect of the data collection period as well as missing customers. Throughout this chapter, we look forward to sharing a series of processes to solve the predictive analytics problem by finding the right features.

In the second part, we introduce a *survival analysis model* powered by a *deep architecture.* We propose this model to challenge more realistic prediction settings having partial observations with the imbalanced distribution. Unlike the framework in the first part, our new *SurvRev* model can predict the event rate of the next 365 days for each visit. We are able to handle partial observations by survival analysis, and the underlying deep learning architecture effectively learns the hidden representation of customers and their visits. By optimizing a custom loss function, our *SurvRev* model can be tuned for various prediction purposes. Throughout this chapter, we introduce our various efforts to refine the model and verify its superiority over other revisit prediction models.

We successfully apply our models to mobility datasets collected from seven flagship stores in downtown Seoul, including more than *5.7 million* visits over *2.5 years*. For fertilizing research, we also release a benchmark dataset of customer indoor movement patterns. We hope that our research and datasets can be used for offspring studies that require understandings of customers' shopping patterns.

Keywords Revisit Prediction, Human Behavior Prediction, Predictive Analytics, Retail Analytics, In-Store Motion Pattern Analysis, User Modeling, Sensor Data, Mobility Data, Feature Engineering, Data Mining, Machine Learning, Deep Learning, Survival Analysis, Longitudinal Data,

Contents

List of Tables

List of Figures

Chapter 1. Introduction

Significant part of the total purchasing activity still happens in the off-line market. And there is an increasing demand for offline retail analytics. By targeting the potential loyal customers who are likely to revisit, merchants can considerably save promotion cost and enhance return on investment [88]. Then, how can we detect a customer who is willing to visit a store again, without performing extensive surveys? Is it really possible to predict a customer's intention to revisit the store without knowing their purchase history, store satisfaction, age, or even their residence location? Can mobility data be useful for predicting revisits? How long should we collect the data to perform revisit analysis? Answers to such questions are vital to retail industries and to give a further direction to marketers, merchants, and retail analysts.

Figure 1.1: Capturing customer mobility in off-line stores[1].

1.1 Main Achievements and Contributions

Get closer than ever to your customers.
So close that you tell them what they need well before they realize it themselves.
— Steve Jobs

My thesis focuses on the *revisit prediction task* in the domain of off-line retail analytics. Customer mobility data is the main resources to investigate customer revisit in this study since the customer mobility data becomes a relatively easy-to-obtain data source, which comes with a large scale. We choose *revisit* as our prediction objective since we are able to track it by sensors but we could not obtain customer's purchase record from our clients. Besides, the *revisit* is a good proxy for estimating customer lifetime value and a meaningful measure by itself. Unless there are no previous studies which aligned with customer revisit and their mobility, it is well known that motion patterns unconsciously reflect

[1] Image courtesy of Walkinsights.

consumer's interest in and satisfaction with the store [54]. Therefore, our key task is to find patterns that affect a customer's revisit.

Of course, people go everywhere and tracking millions of people all days will be costly. So we reduce the scope of our research to utilize the movement patterns of customers inside the target stores. In this way, tracking customer behavior in and around the store becomes feasible by installing dozens of sensors. Naturally, we focus on how customers' behaviors in the store relate to their future revisit. In order to carry out this study, we obtain fine-grained indoor mobility dataset of seven flagship stores over 7–33 months, collected by dozens of in-store sensors for each store. Figure 1.1 illustrates how can we capture customer movements in the store.

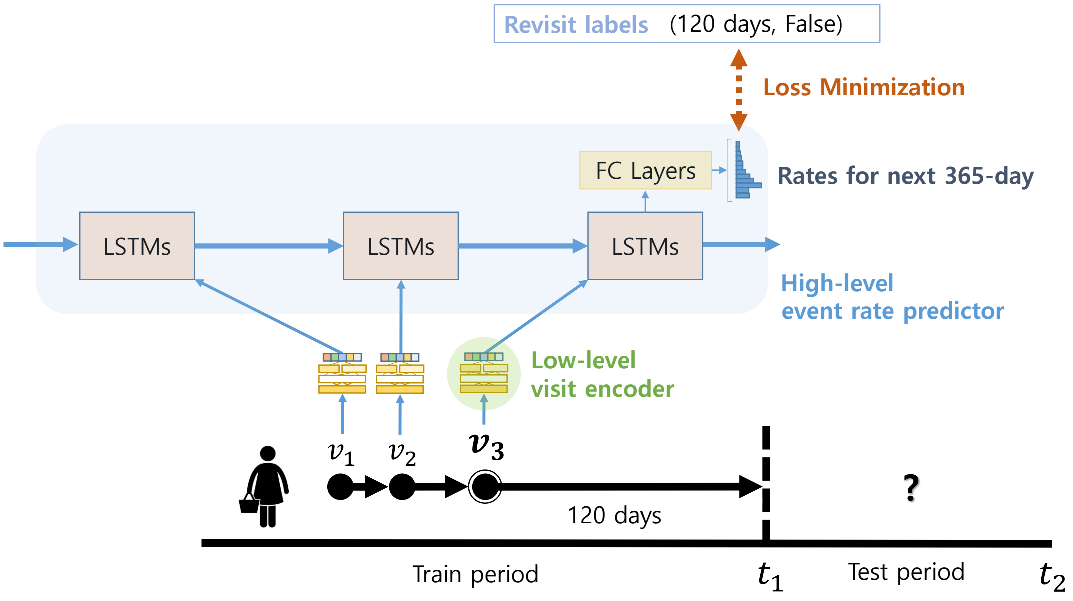
We design predictive analytics of revisit prediction: a classification task to predict customers' intention to revisit, and a regression task to predict customers' future revisit interval. In this thesis, we introduce two different approaches to tackle our problem. A brief summary of our contribution is as follows:

- **Revisit prediction by designing features (§ 3)**: To predict customer revisit, we focus on finding effective feature sets from their motion patterns. We design ten groups of features and utilize five different semantics. We confirm the effectiveness of our features by achieving 64–80% accuracy on binary classification task, with the performance improvement of 4.7–24.3% compared to baselines. We also show that our prediction framework is robust even if we miss the footage of large number of users. Moreover, we show how the performance changes as the data collection time period gets longer. Figure 1.3 shows the essence of above four illustrative findings.
- **Revisit prediction by designing models (§ 4)**: We design a deep survival model that works in a more realistic environment. Our model successfully predicts revisit rates for next time horizon by encoding each visit and managing personalized history. By applying survival analysis concepts, we smoothly handled censored visits, that caused huge data imbalance to led our previous approach to test in a downsampled prediction scheme. Our model is also free from data distribution inconsistency according to the ratio of training and testing set length.

In addition to the ones we listed above, we share diverse findings around the revisit prediction task. In Chapter 3, We investigate the predictive powers of the feature groups and semantic levels of areas. We address the problem of data inconsistency caused by Wi-Fi turn-off rate. In this respect, we develop an approach for revising the probability of being group customers in the data. And we point out the difficulties of securing prediction accuracy in spite of having a noticeable difference between the two groups of customers. We also show that the overall prediction performance can largely depend on the visit statistics of the dataset. In Chapter 4, we talk about the prediction settings that should be kept in order to be used immediately in business. In addition to explaining the various objective functions in the revisit prediction task, we explain how each loss function optimizes its objective function. We show the effectiveness of our model by comparison with survival analysis models, deep learning models as well as our previous framework.

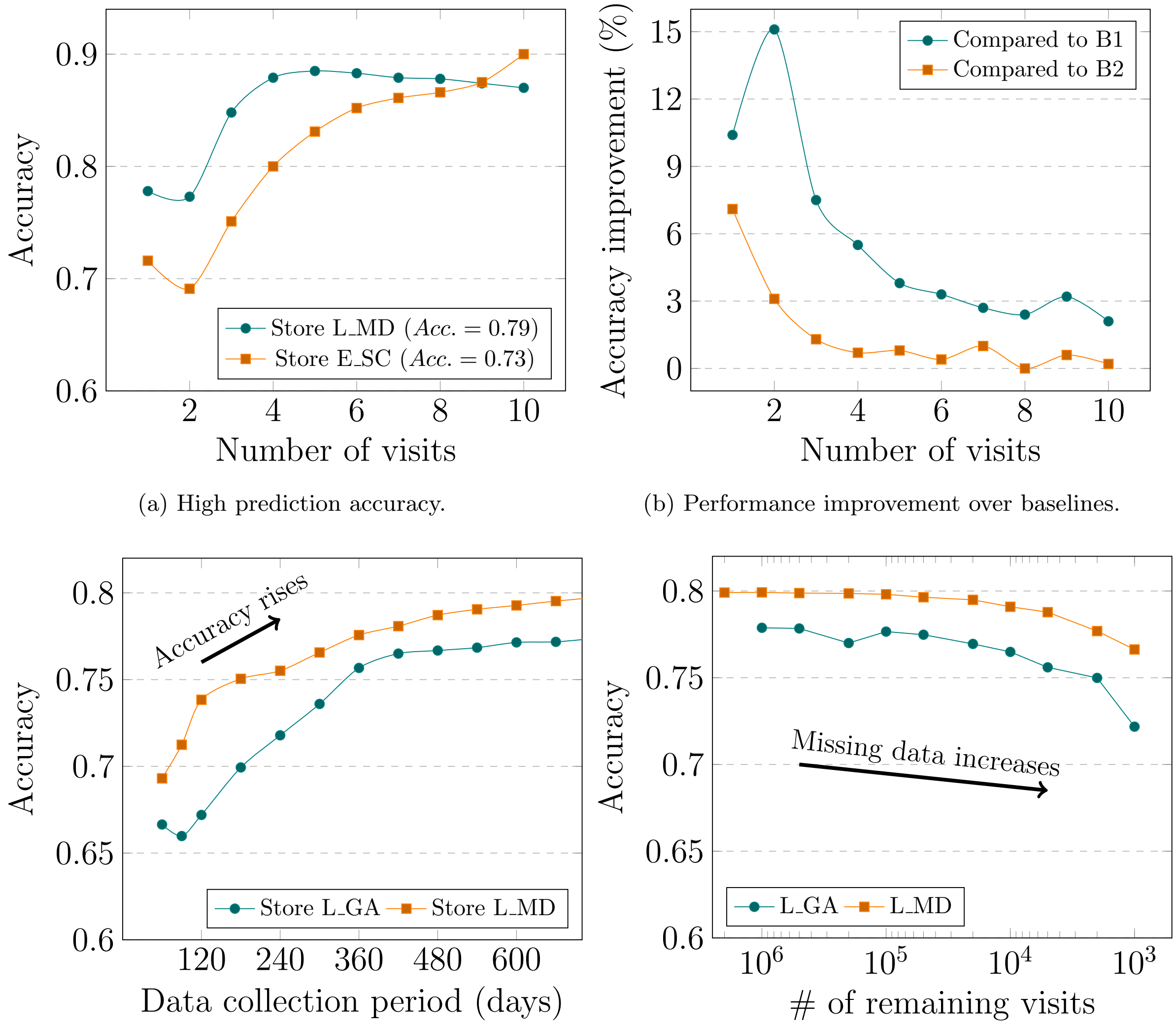
Data sources	Feature groups	Twenty representative features (Among 866 features of store E_GN)
Moving pattern of the visit	Overall statistics **[OS]** (IV-A1)	f1 = Total dwell time
		f2 = Trajectory length
		f3 = Skewness of dwell time of each area
	Travel Distance/ Speed/Acceleration **[TS]** (IV-A2)	f4 = Total distance traveled inside the store
		f5 = Speed based on transition time
		f6 = First-k HWT coefficients of acceleration
	Area preference **[AP]** (IV-A3)	f7 = Coherency of dwell time for each level
		f8 = Top-k-area dwell time
	Entrance and Exit pattern **[EE]**	f9 = Exit gate
		f10 = Number of previous re-entry on that day
	Heuristics **[HR]**	f11 = Wears clothes but does not buy
	Statistics of each area **[ST]**	f12 = Number of time sensed in the area
		f13 = Stdev of dwell time for the area
Temporal information of the visit	Time of visit **[TV]**	f14 = Day of the week
	Upcoming events **[UE]** (IV-A8)	f15 = Remaining day until the next sale
		f16 = Number of holidays for next 30 days
Occurrences before the visit	Store accessibility **[SA]** (IV-A9)	f17 = Number of days since the last access
		f18 = Average interarrival time
Simultaneous visits	Group movement **[GM]** (IV-A10)	f19 = Presence of companions
		f20 = Number of companions

(a) Extensive feature engineering (§ 3).



(b) Deep survival model (§ 4).

Figure 1.2: Key contributions of the thesis.



(a) High prediction accuracy.

(b) Performance improvement over baselines.

(c) Accuracy increases as data becomes longer.

(d) Model robustness on missing customers.

Figure 1.3: Effectiveness of mining customer mobility for revisit prediction. (a) Our model shows high prediction accuracy around 70% even when it comes to predicting the revisit intention of first-time visitors. (b) Performance improvement of our model by using the feature sets against two baselines. (c) To guarantee the revisit prediction performance, we need *sufficient* amount of data. (d) 95% of the performance of our model is maintained with a very small fraction of the dataset. (Full results in § 3.3.2 and § 3.3.3).

1.2 Thesis Organization

The rest of this dissertation is organized as follows. In Chapter 2, we introduce the importance of our revisit prediction problem with our data collection efforts. In Chapter 3, we present our feature engineering approach to tackle this problem. We begin Chapter 4 by emphasizing the importance of realistic prediction settings and introduce the effectiveness of our new deep survival analysis approach. In Chapter 5, we provide conclusions and discuss future directions. An overview of this dissertation can be seen in Table 1.1.

Table 1.1: Overview of the thesis.

Chapter	Contents
(§ 2) Customer Revisit Prediction	(§ 2.1) Motivation and overview (§ 2.2) Related work on revisit prediction (§ 2.3) Indoor data description (§ 2.4) Problem definition
(§ 3) Revisit Prediction by Feature Engineering	(§ 3.1) Motivation: Importance on feature engineering (§ 3.2) Feature engineering (§ 3.3) Experiments
(§ 4) Revisit Prediction by Deep Learning	(§ 4.1) Motivation: Towards practival application settings (§ 4.2) Background on survival analysis (§ 4.3) Deep survival model (§ 4.4) Experiments
(§ 5) Conclusions	(§ 5.1) Contributions (§ 5.2) Impact and achievements (§ 5.3) Vision and future directions
Appendices	(§ A) Benchmark data (§ B) Preliminary neural network approaches (§ C) Application to points-of-interest check-in datasets

Chapter 2. Customer Revisit Prediction

Recently, there is a growing number of off-line stores that are willing to conduct customer behavior analysis. In particular, predicting revisit intention is of prime importance, because converting first-time visitors to loyal customers is of prime importance, because converting first-time visitors to loyal customers is very profitable. However, revisit analyses for offline retail businesses have been conducted on a small scale in previous studies, mainly because their methodologies have mostly relied on manually collected data. With the help of noninvasive monitoring, analyzing a customer's behavior inside stores has become possible, and revisit statistics are available from the large portion of customers who turn on their Wi-Fi or Bluetooth devices. Using Wi-Fi fingerprinting data from ZOYI, we propose a feature engineering framework and a deep learning framework to predict the revisit intention of customers using only signals received from their mobile devices. Our frameworks showed feasibility to predict revisits from customer mobility captured by in-store sensors that have not been considered in previous marketing studies.

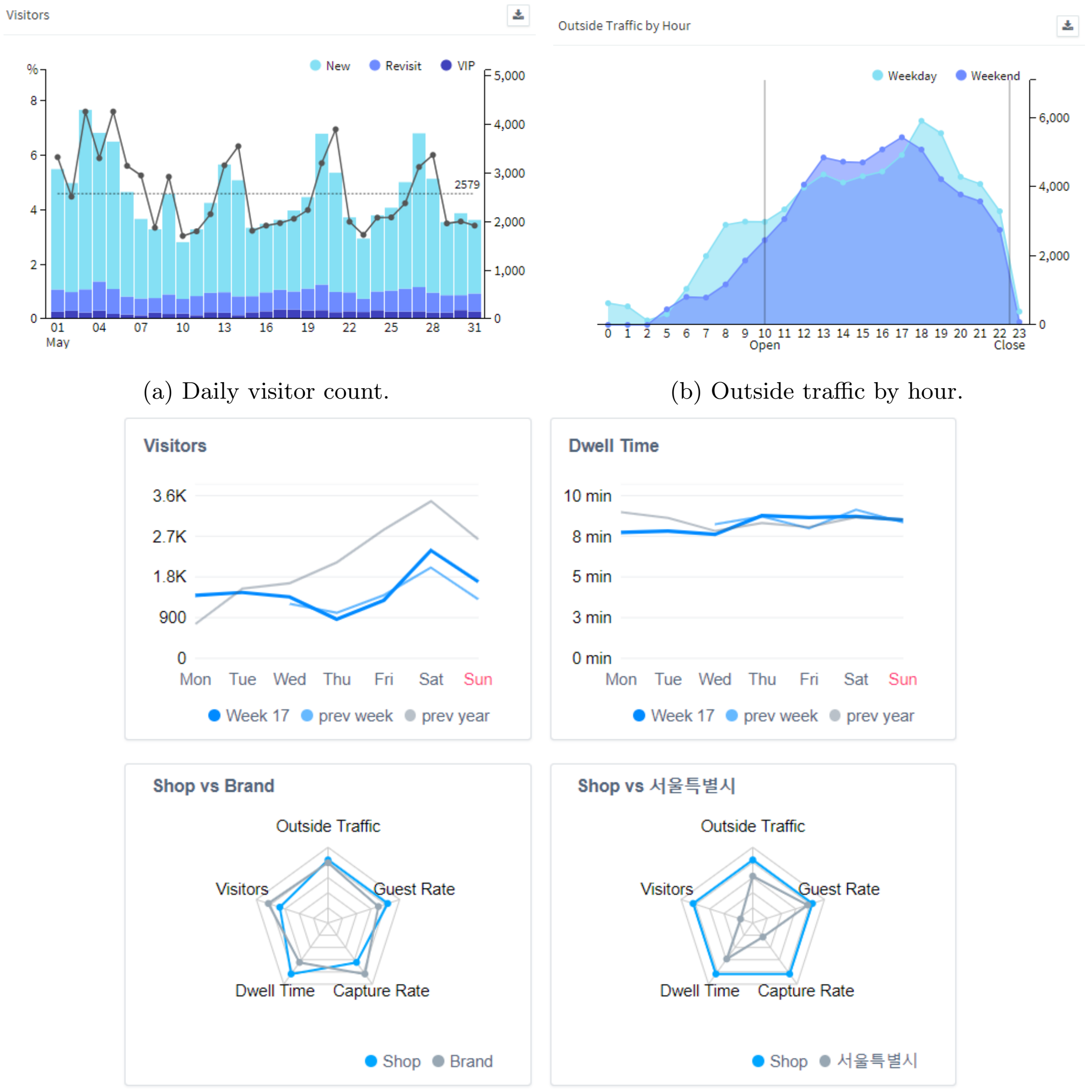
2.1 Motivation and Overview

> *We see our customers as invited guests to a party, and we are the hosts.*
> *It's our job every day to make every important aspect of the customer experience a little bit better.*
>
> — Jeff Bezos

How can we detect a customer who is willing to visit a store again? More challengingly, how can we capture intrinsic mobility patterns that represent future retention? In this study, we introduce a revisit prediction framework using only Wi-Fi signals collected by in-store sensors.

By identifying the potential loyal customers who are likely to revisit, merchants can considerably save promotion cost and enhance return on investment [88]. Many studies in recent years have focused on *online* stores and online text reviews with the help of a data provider [72, 21, 116]. In contrast, the analysis of revisit intention in the *offline* environment has not advanced significantly over the last few decades. The main reason for this lack of progress lies in the difficulties of collecting large-scale data that is closely related to key attributes of revisiting, such as customer satisfaction with products, service quality, atmosphere of the store, purchase history, and personal profiles [116, 110, 33]. The first three attributes are subjective information that is difficult to capture in the offline environment, and the last two attributes are considered as confidential corporate information that is not easily accessible. Owing to these limitations, research on customer revisits in offline stores has been conducted through surveys. These studies help us gain an understanding of underlying hypotheses that affect customer satisfaction. However, because of their inherent limitations of a small sample size, we believe that their findings cannot be easily generalized. Therefore, the large-scale customer analyses for revisit prediction are urgently needed.

With the advance of sensing technologies such as radio-frequency identification (RFID) [37, 106], Bluetooth [122], or Wi-Fi fingerprinting [109], we are capable of collecting high-frequency signal data without installing any applications on customer devices [87, 96, 98, 99]. These signals can be converted to fine-grained mobility data. Using such data, noninvasive monitoring of visitors has been carried out in different settings, such as in museums [122] and supermarkets [114], providing empirical findings of customer behavior, such as finding the corridor where the most visitors pass or the area where the visitors stay the longest. Nowadays, collecting data in a certain physical boundary is called as geofencing [102] and the market size is accounted for USD 8 billion in 2014 and is expected to reach 40 billion by 2019 [90]. Various retail analytics companies installed their own sensors to geotrack real-time mobility patterns of customers in their clients' stores. Their proprietary solutions provide visitor monitoring results, such as funnel analysis or hot-spot analysis results through a dashboard (Figure 2.1). In addition, it is expected that huge amounts of shopping behaviors will be generated in cashier-less stores introduced by the enterprises such as Alibaba and Amazon.



(a) Daily visitor count.

(b) Outside traffic by hour.

(c) An example report showing several statistics.

Figure 2.1: Dashboard examples of a retail analytics company Walkinsights.

This study moves one step forward, from visitor monitoring to customer revisit prediction. We propose a systemic framework for predicting the revisit intention of customers using Wi-Fi signals captured by in-store sensors. It is known that motion patterns unconsciously reflect consumer's interest in and satisfaction with the store [54]. However, among many attributes, customer mobility is not well known to determine their life-time value. Therefore, the key challenge is how to generate the most effective set

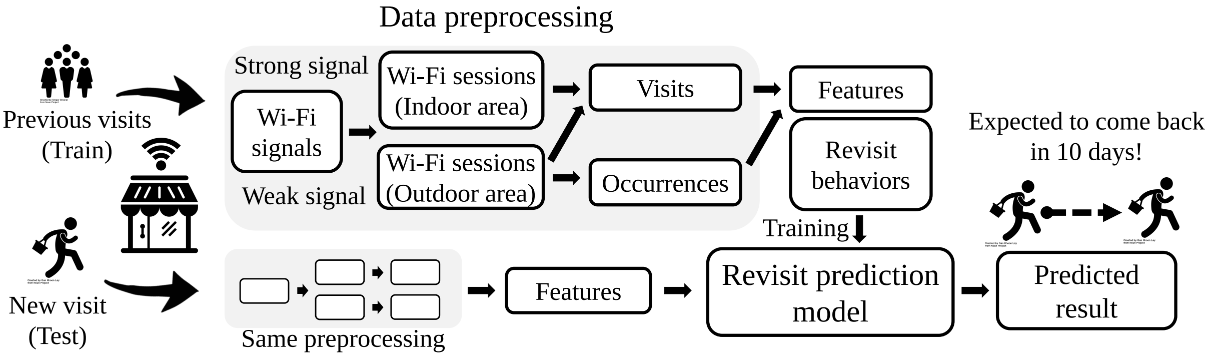


Figure 2.2: Revisit prediction framework architecture.

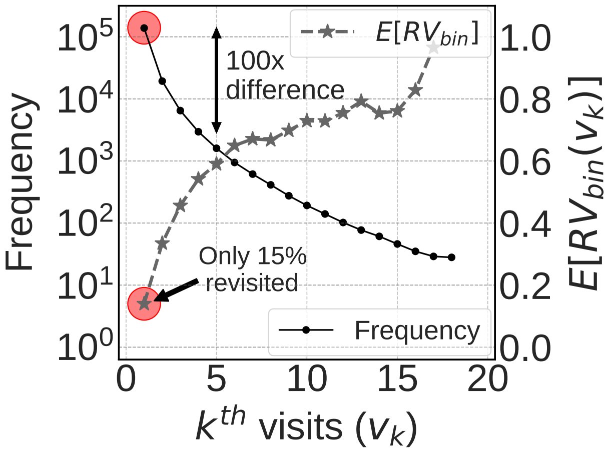


Figure 2.3: Revisit statistics of store E_GN. $E[RV_{bin}(v_k)]$ denotes the average revisit rate of the group of visitors who visit k times.

of features from the Wi-Fi signals and to find the best model to learn from those features. Figure 2.2 illustrates the overall procedure of the revisit prediction framework. If a customer comes into a store, the framework detects his/her Wi-Fi signals and transforms the signals to a visit and an occurrence through the data processing steps. From the customer's visit and previous occurrences, we designed features to describe his/her motion patterns. Finally, we can predict his/her revisit behavior, using a trained model. We benefit from large-scale customer mobility data captured by in-store sensors. Seven flagship stores in downtown Seoul were carefully selected to cover various shop categories. The number of unique customers collected in the seven stores reaches 3.75 million. The data is very attractive because we can capture approximately 20–30 % of the customer mobility data without customer interruption.[1] Furthermore, the data collection period is 1–2 years, which is long enough to study revisit behaviors.

Figure 2.3 illustrates the observed revisit statistics during the data collection period in one of our store. The black line denotes the number of observations $|v_k|$ of k^{th} visits (v_k), and the gray line denotes the average revisit rate $E[RV_{bin}(v_k)]$ of all v_k's. The fact that the $|v_5|$ is 100 times less than $|v_1|$ implies that it is very difficult to retain first-time visitors as regular customers. It also describes how valuable it is to raise the revisit rate of first-time visitors that account for 70 % of total customers,[2] since it is well

[1] Customers carrying a smartphone with their Wi-Fi on are detected by the Wi-Fi positioning system. The proportion of users in their twenties who keep their Wi-Fi on is 29.2 %, according to a conducted by Korea Telecom in July 2015 [128].

[2] In Figure 2.3, the ratio of the first-time visitors in store E_GN is over 70 %. We made a few assumptions to interpret the data as it is and will discuss them in Section 3.3.3.

known that retaining first-time visitors is extremely important. By increasing 5% in the retention rate of the customer results in an increase of 25-95% in profit[3]. Therefore, the proposed framework should be capable of handling instances without having any history, which is well known as a cold-start problem.

In the following two chapters, we introduce our two approaches to predict customer revisit. Our experiments demonstrate that our frameworks successfully predicts the revisit, especially for first-time visitors. The inputs for prediction framework are all derived from Wi-Fi signals with minimal external information (dates of public holidays, clearance sales). Thus, we expect that the prediction power will rise significantly by adding private data such as personal profiles and purchasing patterns.

Before moving to the main part, we introduce some backgrounds in the remainder of this chapter. In Section 2.2, we summarize previous studies related to this thesis. In Section 2.3, we describe the datasets used. Then we introduce the key terms and formalize our problem in Section 2.4.

2.2 Related Work

In this chapter, we discuss previous works that related to this thesis. We start this chapter by introducing works belongs to revisit studies and efforts in retail analytics. Then, we summarize works related to customer research and indoor analysis. Last, we broadly review the techniques used in trajectory mining and POI recommendation task.

2.2.1 Revisit Studies

Majority of previous studies related to revisit intention or repurchase behaviors, qualitatively studies several causes to affect customer's behaviors through surveys. The typical approach is as follows. First, researchers sample several hundred customers to fill out surveys then researchers perform correlation analysis to check which factors are related to revisit or repurchase. In outdoor brands, engagement and trust have a positive correlation with revisit intention [39]. And the salesperson's service affects customer's satisfaction of the offline store, and convenience of mobile payments affects the store's overall satisfaction level. Furthermore, customer's satisfaction level has a strong correlation with revisit intention [71]. Positive sentiments lead customers to repurchase although they have some negative sentiments on stores [62]. These factors are intuitive when considering customer satisfaction or loyalty. However, the required information is subjective so it can be only achieved through surveys, which is difficult to collect in large quantities. On the other hand, in this thesis, we confirmed the association between the customer revisit and their mobility which are not discovered in the previous studies. Most importantly, the mobility data can be collected with scale. In the next section, we introduce researches and companies that have gathered their offline data and created new horizons of retail analytics.

2.2.2 Data Collection Efforts

Recently, RFID [37, 106], Bluetooth [122], or Wi-Fi fingerprinting [109], enable us to collect high-frequency signal data without installing any applications on customer devices [87, 96, 98, 99]. These signals can be converted to fine-grained mobility data. Using such data, noninvasive monitoring of visitors has been carried out in different settings, such as in museums [122] and supermarkets [114], providing empirical findings of customer behavior, such as the corridor where the most visitors pass or the area where the visitors stay the longest. Nowadays, collecting data in a certain physical boundary

[3] https://hbswk.hbs.edu/archive/the-economics-of-e-loyalty

is called as geofencing, and the market size is accounted for USD 8 billion in 2014 and is expected to reach 40 billion by 2019 [90]. Companies such as ZOYI[4], VCA[5], RetailNext[6], Euclid[7], ShopperTrak[8], and Purple[9] have their proprietary monitoring solutions and provide a dashboard to their clients. Using their monitoring technology, they perform a funnel analysis to identify a ratio of customer over outside traffics. These statistics is being used to determine which area it would be best to open a new store. Another company named Loplat[10] collects customer foot traffic in a different way. Without installing sensors, they collect a Wi-Fi fingerprint snapshot from the store and use it as an identifier of each store. Based on the collected Point-of-Interests (POI) information, they provide their location recognition software development kit to third party applications, for other companies to add location-based features on their products. In this way, Loplat also collect the customer mobility data from third party applications. They recently launched a B2B service, which provides both customer analysis and marketing campaigns.

2.2.3 Behavior Analysis of Shoppers

Park et al. [85] examined the factors of route choice in three clothing outlets by tracking 484 customers. They considered spatial characteristics of the outlet, types of customers, and their shopping behaviors. For the perspective of interior design, they analyzed the passage types of shopping centers, the location of the main entrance, and a direction of the main escalator to find relations with customer route choices. They were interested in the relation between the length of hallway and in-depth shopping rate. In addition to that, they studied movement patterns of different customer groups. Their analysis helps us to develop diverse features in our revisit prediction framework. In the grocery store, an RFID-based tracker system with shopping carts enabled Hui et al. [37] to find evidence for interesting behaviors, such as consumers who spent more time in the store become more purposeful. They also reveal another interesting behavior through their collected data, which is, after purchasing virtue categories, presences of other shoppers attract consumers yet reduce their tendency to shop. Yada et al. [114] applied a character string analysis techniques EBONSAI originally developed in the field of molecular biology. They convert each shopping area into a character to apply their algorithm in order to discover purchasing behaviors. Hwang et al. [38] introduced process mining techniques to understand customer pathways. The Petri-net model learned by inductive learning algorithms provides a formal representation of the shopping path of customers. With the collaboration of sensor providers as well as their clients, they showed customers' behavioral patterns and sales revenue were changed according to the change of store display and the process model according to them also changed. This study is also meaningful in terms of that they utilized the Kolon store dataset collected by ZOYI corporation, a data provider[11] of seven indoor datasets for this thesis. Although these studies did not focus on customers' revisit, they were valuable resources for us to develop features that describe customers' motion patterns. Currently, Alibaba's Hema Xiansheng[12] and Amazon Go[13] are the most widely known "future retailers" by breaking

[4] https://walkinsights.com/

[5] https://www.ucountit.com/

[6] https://retailnext.net/en/aurora/

[7] https://www.ucountit.com/

[8] https://www.shoppertrak.com/

[9] https://purple.ai/

[10] http://loplat.com/

[11] We purchased the data access right of the seven stores.

[12] https://www.freshhema.com/

[13] https://en.wikipedia.org/wiki/Amazon_Go

the traditional retail experience by introducing technology. We expect that there will be tremendous opportunities to study customer behavior patterns during their shopping.

2.2.4 Behavior Analysis in Other Indoor Places

Traditionally, the analysis of customers' indoor movement and connections to space has been conducted in the area of architecture or interior design. Especially in the case of museums, various movement patterns were captured to rearrange the exhibits to enhance the satisfaction of visitors [69, 68, 115, 44, 67]. In these researches, itinerary tracking and time tracking techniques [105, 115] were used to investigate the spectator's behavior in exhibition halls. With the manual tracking technique, one can capture very accurate visitor's movement and the status of the exhibition space, which helped them to manage exhibitions. For example, the extent of visibility of the display was studied [70]. Moreover, the behavior of passive visitors was utilized to arrange the main display [44]. They concluded that visitors are influenced by the continuity in display within their view, since they select their movements in an efficient way between the exhibition halls. These studies were conducted with a limited data around several hundreds of customers. Nonetheless, concepts and analysis techniques presented in these existing study were valuable to find the insights the large-scale data that we have.

With the help of noninvasive monitoring, visitor studies in the museum have come to a new phase. Yoshimura et al. [122] collected more than 80,000 devices' footprints by installing 8 beacons in main hallways of the Louvre Museum and analyzed the most used trajectories between exhibition halls to mitigate a micro-congestion inside a museum. By tracking visitors' movements, the Guggenheim museum [31] increased customers' engagement with the museum by making smarter curatorial decisions. Both museums and clothing stores are places where customers' indoor mobility data are meaningful resources to study for customer satisfactions. We expect the framework we have presented is also applicable in the museum visitors studies.

2.2.5 Predictive Analytics Using Trajectories

Using trajectories, next location prediction is one of the most studied topics in the computer science community. The studies using GPS-based trajectories are introduced in this paragraph. To predict the next location, frequent trajectory patterns [82], nonlinear time series analysis of the arrival and residence time [100], HMM [81] were applied. The results support the prediction of the next location using partial trajectories is feasible, along with the regularity studies of human mobility [76, 26, 103]. Within the subject of predicting the next location, the prediction of the final destination of a taxi [7, 78] was also actively studied after the 2015 ECML/PKDD competition [45].

Next location prediction study is also well known in the field of Point-of-interest(POI) recommendation. The main data sources are check-in datasets in location-based social networks such as Foursquare [118] or Gowalla [74], where trajectories were actively recorded by the users. The survey [124] gives the basic understandings of the difference between traditional recommendation tasks and POI recommendation tasks, and they classify POI recommendation algorithm into four categories: pure check-in based, geographical, temporal, and social-influenced. Yiding et al. [73] extensively evaluated 12 state-of-the-art POI recommendation models to better understand and utilize models in various scenarios. They also summarized the characteristics of 41 existing papers from 2010 to 2016. From their observation, top 3 outperform models [64, 75, 66] are based on implicit feedback models, and consider geographical information. Details on each model can be found on the above-mentioned experimental paper [73].

The main difference between our study and previous studies are a prediction objective. We studied the customers' revisit intentions in the offline stores using indoor trajectories and check-in traces. Thus, our model focused on predicting revisits instead of predicting next locations. As far as we know, there are no studies of predicting revisit intention using semantic trajectories captured by in-store sensors.

2.3 Indoor Data Description

In this section, we introduce our customer mobility data captured from off-line stores. The number of customers in our data is very high, and the collection period is long enough to obtain reliable results. Throughout this section, we share some statistics of our datasets and introduce necessary preprocessing to find meaningful semantics from the raw Wi-Fi signals.

2.3.1 Data Collection

We collected data from seven flagship stores[14] located in the center of Seoul. Each of these stores is one the largest stores of each brand, consisting of several floors. These stores are known to be the busiest stores in Korea except for some complex malls or department stores. Because of their location and size, these stores have up to 10,000 daily visitors. For example, our target store E_GN is a four-story building located on the side of a Gangnam boulevard where two million people walk by each month[15]. Store E_SC is located at the ground floor of a major department store in the downtown area of Seoul, which is also connected to one of the busiest subway stations used by college students. The main target customer of our stores is the young generation, ranging from the age of high teens to mid-twenties. Store L_MD and store O_MD are located at the main street of Myeongdong area, which is the most popular district in Korea by foreigners. Table 2.1 presents the statistics of the seven datasets, and Figure 2.5 illustrates the location of sensors and categories of two stores E_GN and E_SC to give readers a sense of how sensors are installed throughout the store.

Revisit Statistics of Seven Stores

Figure 2.4 illustrates the observed revisit statistics of seven stores during the data collection period, which is a full version of Figure 2.3. The black line denotes the number of observations $|v_k|$ of k^{th} visits (v_k), and the gray line denotes the average revisit rate $E[RV_{bin}(v_k)]$ of all v_k's. Overall, the visit count distribution follows a power-law distribution, and as the number of visits increases, the revisit rate also tends to increase to certain threshold. All seven stores seem to have difficulties of retaining first-time visitors as regular customers. Even if the stores with data collection period nearby two years, such as L_GA, L_MD, and O_MD, the number of frequent visitors (more than 5 times) is less than 10 % of the number of first-time visitors.

2.3.2 Preprocessing

Raw data—Wi-Fi Signals

To collect Wi-Fi signals, we utilized ZOYI Square sensors developed by WalkInsights[16]. The installed sensors enable us to collect Wi-Fi signals from any device that turns on its Wi-Fi. A single Wi-Fi

[14] Owing to a nondisclosure agreement, brand names and exact locations cannot be disclosed, and neither the data.

[15] These statistics were measured from September 18, 2017 to October 17, 2017.

[16] `https://walkinsights.com/sensors`

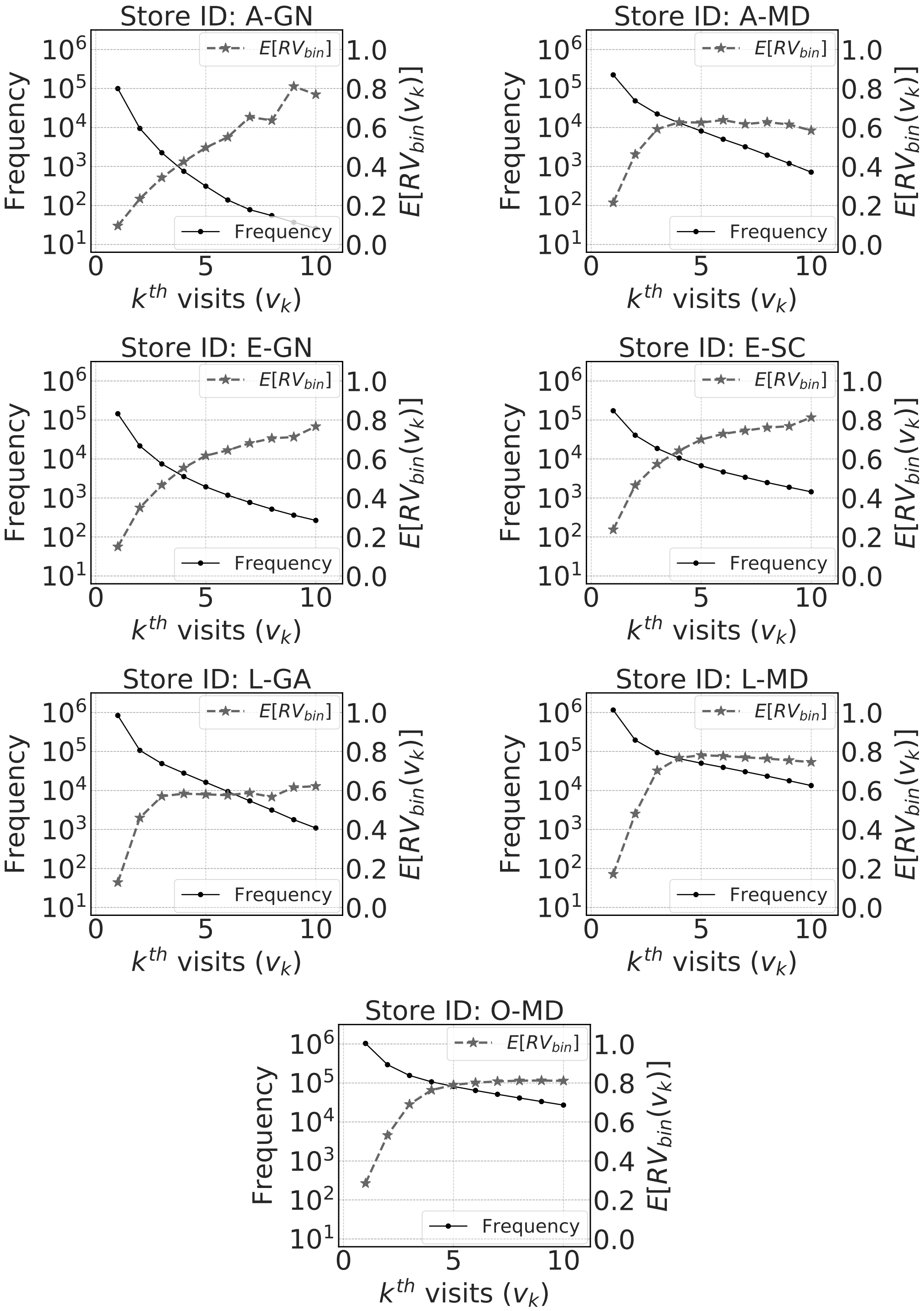
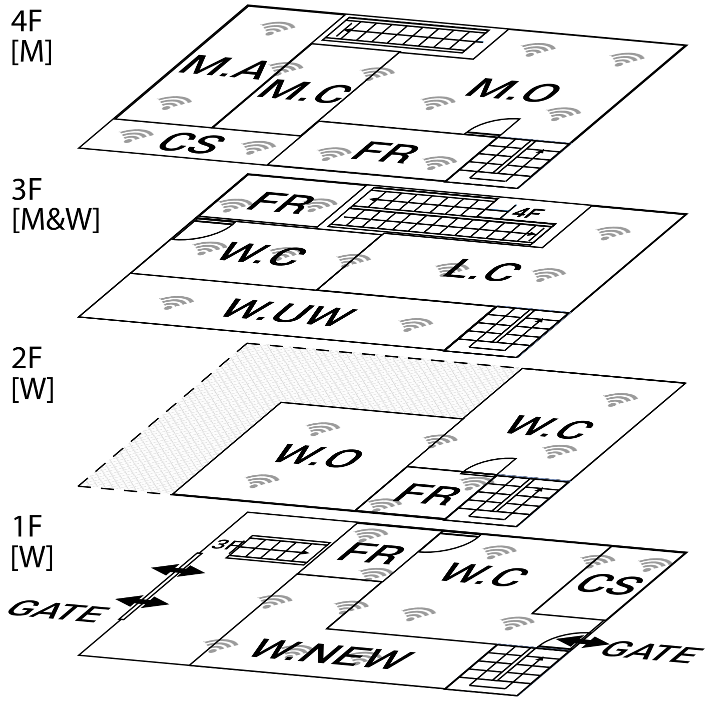


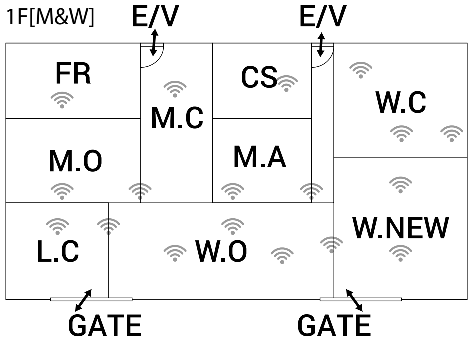
Figure 2.4: Revisit statistics of seven stores.

Table 2.1: Statistics of the datasets.

	A_GN	A_MD	E_GN	E_SC	L_GA	L_MD	O_MD
Category	Footwears	Footwears	Fast-fashion	Fast-fashion	Character shop	Character shop	Cosmetics
Location	Gangnam	Myeong-dong	Gangnam	Sinchon	Garosu-gil	Myeong-dong	Myeong-dong
Collection period	17.05.24 – 17.12.31	17.05.26 – 17.12.31	17.01.13 – 17.11.08	16.11.01 – 17.11.08	15.04.17 – 17.12.31	15.12.15 – 17.12.31	15.12.07 – 17.11.03
Collection length	222 days	220 days	300 days	373 days	990 days	747 days	698 days
# of sensors	16	27	40	22	14	11	27
# of total signals	165,443,933	890,267,554	939,815,485	632,106,890	1,935,362,316	2,818,001,166	6,499,265,088
Signal data size	15 GB	77 GB	148 GB	99 GB	164 GB	242 GB	567 GB
# of total sessions	19,937,461	33,862,373	81,449,603	34,131,034	40,282,894	74,324,676	90,713,930
# of indoor sessions $\geq$ 5 s	636,843	3,250,072	1,353,709	1,921,635	5,461,060	11,065,561	15,581,820
# of visits $\geq$ 60 s	112,672	327,940	183,246	270,366	1,062,226	1,718,359	2,008,384
# of unique visitors $\geq$ 60 s	100,741	232,051	147,096	186,617	846,487	1,171,583	1,065,803
Avg. revisit rate	11.73 %	31.99 %	21.18 %	36.55 %	21.22 %	32.98 %	48.73 %
Power-law coeff α	11.019	3.904	5.887	3.587	5.548	3.563	2.732



(a) Floor plan of store E_GN. Store E_GN has five semantic levels: sensor, category, floor, gender, and in/out.



(b) Floor plan of store E_SC.

Area	Description
M.O	Men's office wear
M.C	Men's casual wear
M.A	Men's accessories
W.O	Women's office wear
W.C	Women's casual wear
W.UW	Women's underwear
W.New	Women's new arrival
L.C	Limited collection
FR	Fitting room
CS	Checkout counter

(c) Area explanation.

Figure 2.5: Location of sensors and categories of two stores considered in the study. Wi-Fi icons indicate the location of the sensors, and the category names for each section are described in (c).

signal includes an anonymized device ID, sensor ID, timestamp, and its Received Signal Strength Indicator (RSSI) level [99]. RSSI is an indication of signal strength received by sensors, which is represented in a negative value. The closer the RSSI level is to 0, the clearer the signal is. Signals are collected continuously from each device at fairly short intervals, which are less than 1 s. Since the size of the signal data is considerably large – 1.3 TB, we carried out a conversion process to remove redundant signals and combine into Wi-Fi session logs. The leftmost part of the Figure 2.7 illustrates examples of Wi-Fi signals and the main preprocessing concept, which will be explained in the following section.

Signal to Session Conversion

With the received signals, we approximate the location of a device by indoor positioning. We call this a signal-to-session conversion. A row of Wi-Fi session data becomes an element of a semantic trajectory that includes device ID, area ID, and dwell time. Predefined RSSI thresholds are utilized for signal-to-session conversion. These thresholds are controlled during the installation and the values guarantee that the device is in the vicinity of a sensor. The logic of this conversion is simple. For instance, a new session is created when a sufficiently strong RSSI is received for the first time. The session continues if the sensor receives consecutive strong signals, and it ends if the sensor no longer receives strong signals. The session also ends if another sensor receives a strong RSSI from that device.

Location Semantics

It is also possible to detect the semantic location of a customer by taking advantage of the semantic coherency of contiguous sensors. For example, we can identify if the customer is looking at daily cosmetics or she/he is in a fitting room. Additionally, we can describe a customer's location to floor-level or gender-level semantic areas. Moreover, we generate in/out level areas by examining whether the customer is inside the store, nearby the store (up to 5 m), or far away from the store (up to 30 m). This becomes possible by controlling multiple RSSI thresholds to activate detection with weaker signals. Therefore, an entity of Wi-Fi session data encompasses a customer's dwell time not only in the area corresponding to sensors, but also in the wider semantic areas. By integrating the Wi-Fi sessions with different semantics, we construct a multilevel semantic trajectory to describe each visit as illustrated in Figure 2.7.

Additional Data Cleaning Steps

There are several additional preprocessing steps that we would like to mention. We removed the top-100 most frequent visitors as outliers. Perhaps those outliers are retail clerks or courier delivery persons. We considered the people who stayed less than 60 s as pedestrians who simply walk through the store, and therefore we removed these trajectories as well. Besides, we removed the data received from Apple devices, which follow a MAC addresses randomization policy after iOS 8.0 [80], which makes infeasible to identify the same customer.

Interval Between Sessions: Defining a Threshold for Customer Revisits

As explained in Chapter 2.3.2, if a customer has stayed within the radius of a particular sensor, a single Wi-Fi session of the customer is created. Multiple sessions are created for that customer if he moves to the other area, or he left and comes back to the store again. Here, we consider a session interval between two consecutive Wi-Fi sessions, which is defined as a time difference between the end of the previous session to the start of the current session. Figure 2.6 shows a session interval distribution of our datasets. Interestingly, the interval distribution between sessions has a multi-modal form with three peaks.

- The first mode contains intervals between 0.01–100 s and the peak appears in 1 s. This set of records indicates intervals between consecutive sessions taken on the same visit. In other words, those represent transition time between two indoor spaces while customers are walking around the store.
- The second mode contains intervals between 10 min to 10 h and a relatively small peak appears around 2 h. This represents customer *revisit* on the same day. There might be diverse reasons to do so. For instance, a customer who comes back to the store to buy items he dibs on.
- The last mode contains intervals over 10 h. Intervals in this mode refer to the case of returning to the store on another day. Through this analysis, we have taken a 10 h interval threshold to set the criteria for defining a new visit. Revisit in our study means the corresponding intervals in this case.

2.4 Problem Definition

In this section, we formally define the main concepts introduced in our paper. First, we define a multilevel semantic trajectory ($\mathbb{T}$) that expresses a customer's moving pattern, and define visit (v) and occurrence (o) using $\mathbb{T}$. Next, we define the revisit interval (RV_{days}) and the revisit intention (RV_{bin}), which are labels in our prediction model. Finally, we introduce the revisit prediction problem.

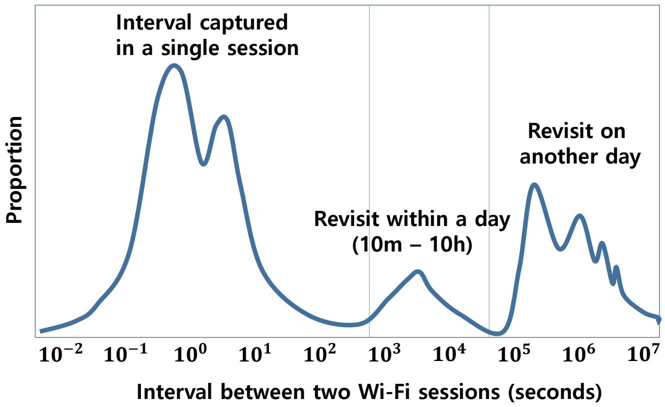


Figure 2.6: A session interval distribution.

2.4.1 Key Terms and Concepts

Multilevel Semantic Trajectory

Multilevel semantic trajectory describes a customer's motion pattern with multiple levels of semantic areas.

Definition 2.4.1. *(Semantic trajectory [117]) A* semantic trajectory $\mathcal{T}$ *is a structured trajectory of size* $n\,(n \geq 1)$ *in which the spatial data (the coordinates) are replaced by semantic areas, that is,* $\mathcal{T} = \{s_1, s_2, \ldots, s_n\}$*, where each element (= a session) is defined by* $s_i = (sp_i, t_{in}^{(sp_i)}, t_{out}^{(sp_i)})$*. Here,* sp_i *represents the semantic area.* $t_{in}^{(sp_i)}$ *is the incoming timestamp for entering* sp_i*, and* $t_{out}^{(sp_i)}$ *is the outgoing timestamp for leaving* sp_i*.* □

If the dwell time of each area $t_{out}^{(sp_i)} - t_{in}^{(sp_i)}$ is shorter than 5 s considering walking speed and the 5 m distance between adjacent sensors, a customer is likely to pass that area without consideration, and thus, we delete the element from the trajectory. Every timestamp in a semantic trajectory should appear within the same day.

Definition 2.4.2. *(Multilevel semantic trajectory) A* multilevel semantic trajectory $\mathbb{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_l\}$ *is a set of semantic trajectories with* l $(l \geq 1)$ *different semantic levels. Each semantic trajectory* $\mathcal{T}_i$ *represents the same motion pattern from a customer using semantic areas of level* i*.* □

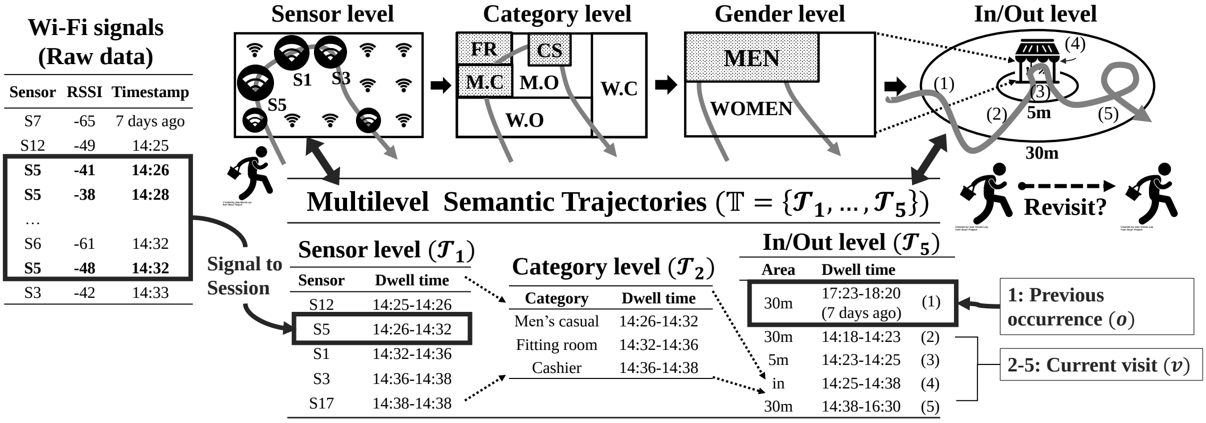
For our indoor environment, we utilized semantic levels inside the store, except for the highest level l indicating the in/out level. The total dwell time of $\mathcal{T}_l$ is always longer than $\mathcal{T}_1, \ldots, \mathcal{T}_{l-1}$, because the in/out mobility utilizes weak signals that can be captured for a longer period than the strong signals used for indoor behavior.

Visit and Occurrence

A set of semantic trajectories describes each visit and occurrence, as defined in Defs 2.4.3 and 2.4.4.

Definition 2.4.3. *(Visit) A* visit v *is a unit action of entering the store.* $v_k(c, [t_s, t_e], \mathbb{T})$ *is a* k^{th} *visit by customer* c*, who is sensed from* t_s *to* t_e*, of which the motion pattern is described with a multilevel semantic trajectory* $\mathbb{T}$*.* □

We consider only the visits that are long enough to represent meaningful behavior. For the sensor-level trajectory $\mathcal{T}_1$, the total dwell time $t_e - t_s$ should be greater than 1 min, because it takes less than



Wi-Fi signals (Raw data)

Sensor	RSSI	Timestamp
S7	-65	7 days ago
S12	-49	14:25
S5	**-41**	**14:26**
S5	**-38**	**14:28**
…		
S6	-61	14:32
S5	**-48**	**14:32**
S3	-42	14:33

Sensor level ($\mathcal{T}_1$)

Sensor	Dwell time
S12	14:25-14:26
S5	14:26-14:32
S1	14:32-14:36
S3	14:36-14:38
S17	14:38-14:38

Category level ($\mathcal{T}_2$)

Category	Dwell time
Men's casual	14:26-14:32
Fitting room	14:32-14:36
Cashier	14:36-14:38

In/Out level ($\mathcal{T}_5$)

Area	Dwell time	
30m	17:23-18:20 (7 days ago)	(1)
30m	14:18-14:23	(2)
5m	14:23-14:25	(3)
in	14:25-14:38	(4)
30m	14:38-16:30	(5)

Figure 2.7: Generation of multilevel trajectories to predict customer revisit: Using noninvasive monitoring, customer Wi-Fi signals are collected. These are then transformed into a sensor-based trajectory, and further summarized into categories, floors, genders, and surrounding areas. The features extracted from these multilevel trajectories effectively determine the characteristics related to customer behavior.

1 min to go through the store. The data preprocessing thresholds are empirically configured depending on the size of a store and the number of sensors.

Definition 2.4.4. *(Occurrence) An* occurrence *o is a special case of a visit that represents a unit action of lingering around the store without entering. $o_k(c, [t_s, t_e], \mathbb{T})$ is a k^{th} occurrence by customer c, who is sensed from t_s to t_e, of which the mobility is only captured in the outdoor area with $\mathbb{T} = \{\emptyset, \ldots, \emptyset, \mathcal{T}_l\}$.* □

Although we did not have any personal information such as the customer's residence, we could measure his/her accessibility to the store through the occurrence. For each visit v_i, we use a set O of previous occurrences $\{O \mid o_k \in O, \ \forall\ o_k(t_e) < v_i(t_s)\}$ as a reference to generate store accessibility features [**SA**], which will be explained in Section 3.2.2.

Revisit Interval and Revisit Intention

If a customer revisits the store after d days, the previous visit v of the customer has a d-day revisit interval, denoted by $RV_{days}(v) = d$, and a *positive* revisit intention, denoted by $RV_{bin}(v) = 1$, as in Definition 2.4.5.

Definition 2.4.5. *(Revisit interval and revisit intention) If two consecutive visits $v_k = v_k(c_i, [t_{k,s}, t_{k,e}], \mathbb{T}_k)$ and $v_{k+1} = v_{k+1}(c_i, [t_{k+1,s}, t_{k+1,e}], \mathbb{T}_{k+1})$ of customer c_i meet the condition $t_{k,e} < t_{k+1,s}$, the revisit interval $RV_{days}(v_k)$ and the revisit intention $RV_{bin}(v_k)$ of the former visit v_k are as follows:*

$$\begin{aligned} RV_{days}(v_k) &= \#\ days\ of\ t_{k+1,s} - t_{k,e} \\ RV_{bin}(v_k) &= 1 \end{aligned} \tag{2.1}$$

If a visit v_k does not have any following revisit, then

$$\begin{aligned} RV_{days}(v_k) &= \infty \\ RV_{bin}(v_k) &= 0 \qquad \square \end{aligned} \tag{2.2}$$

2.4.2 Predictive Analytics

Our predictive analytics problem is now formally defined as follows:

Customer Revisit prediction

Task: Given a set of visits $V_{train} = \{v_1, \ldots, v_n\}$ with *known* revisit intentions $RV_{bin}(v_i)$ and revisit intervals $RV_{days}(v_i)$ ($v_i \in V_{train}$), build a classifier C that predicts *unknown* revisit intention $RV_{bin}(v_{new})$ and revisit interval $RV_{days}(v_{new})$ for a new visit v_{new}.

Chapter 3. Revisit Prediction By Feature Engineering

Chapter based on work that appeared at ICDM 2018 [52] and the forthcoming KAIS journal [53].

In this chapter, we introduce our feature engineering framework to predict the revisit intention of customers. In a binary prediction task with 50 % baseline prediction accuracy, we achieved 67−80 % accuracy for all customers and 64−72 % accuracy for first-time visitors. The performance improvement by considering customer mobility was 4.7–24.3 % over the baselines. We tried various classifiers and confirm that LightGBM [49] was the most effective and efficient. Toward this goal, we study the predictive power of each group of features and the effectiveness of each semantic level to show whether or not the trajectory abstraction boost the predictability. Furthermore, we provide an in-depth analysis regarding the effect of data collection period and present the robustness of our model on missing customers. Another important thing to share is our efforts to resolve gaps that appear between data and actual phenomena. Lastly, we report the unexpected prediction challenges even when the two groups of data show inherent differences in a statistical sense.

3.1 Motivation

> *Coming up with features is difficult, time-consuming, requires expert knowledge. "Applied machine learning" is basically feature engineering.*
>
> — Andrew Ng

It is important to understand that predictive analytics is not a magic. In most cases, the machine learning algorithm can only extract meaning from the data that we give it. It does not have the wealth of intuition that a human has, and subsequently the success of the algorithm can often hinge on how you engineer the input features.

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. Feature engineering is fundamental to the application of machine learning and it is often the most important step in the whole machine learning pipeline. If feature engineering is done correctly, it increases the predictive power of any machine learning algorithm by creating features from raw data that help facilitate the whole process.

Still, many winning solutions on data mining competitions are established on carefully designed handcrafted features [72, 30, 46, 19]. Well-known champions in a data mining competition platform—*Kaggle* agreed that applying feature engineering as much as possible is of prime importance in succeeding in machine learning competition as well[1][2]. Any experienced professional can recall numerous times when a simple model trained on high-quality data was proven to be better than a complicated model built on data that was not clean.

To achieve our goal, we had to ensure that the Wi-Fi signals data contains relevant indicators for the customer revisit prediction. Throughout this chapter, we show how we discover the relevant patterns by elaborating our insights.

[1] `http://bit.ly/competition-tip-giba`

[2] `http://bit.ly/competition-tip-kazanova`

The remainder of this chapter is organized as follows. We describe the characteristics of the features in Section 3.2. In Section 3.3, we explain the experiment settings and present overall prediction results. After we discuss the lessons and challenges obtained through the experiments, we conclude this chapter.

3.2 Key Contribution: Handcrafted Features

To have a multiperspective view of customer movements, we construct each visit as a five-level[3] semantic trajectory, $\mathbb{T} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \ \mathcal{T}_5\}$, where the levels correspond to *sensor*, *category*, *floor*, *gender*, and *in/out*, respectively. We expect the pattern captured using multiple levels can be helpful in predicting customer revisits Thus, some features were created for each semantic level. For example, a customer c_1 who made a recent visit $v(c_1, [t_s, t_e], \mathbb{T})$ with $|\mathcal{T}_1| = 20, |\mathcal{T}_2| = 3$ are more likely to return than a customer c_2 with $|\mathcal{T}_1| = 1, |\mathcal{T}_2| = 1$, because c_1 has longer trajectories than c_2, which implies that c_1 has more interest in the store than c_2. If c_1's occurrences $\mathbb{T} = \{\emptyset, \emptyset, \emptyset, \emptyset, \mathcal{T}_5\}$ have been captured on a daily basis, then c_1 is likely to be a commuter to a nearby office, and their RV_{days} is likely to be much smaller than a noncommuter's $E[RV_{days}]$. In this example, $|\mathcal{T}_i|$ and the average interarrival time are features to predict a revisit, generated from visits and occurrences, respectively. In the rest of this section, we first give an overview of how all features are systematically generated and then introduce the important features of each group in detail.

3.2.1 Overview of Features

Table 3.1 gives a summary of the features in our framework, which are self-explanatory. The first two columns describe data sources used to extract features, leading to ten different feature groups. The first six feature groups—overall statistics [**OS**], travel distance/speed/acceleration [**TS**], area preference [**AP**], entrance and exit pattern [**EE**], heuristics [**HR**], and time of visit [**TV**]— are generated from the *visit* itself. Upcoming events [**UE**], store accessibility [**SA**], and group movement [**GM**] features are generated using certain references: Time of visit [**UE**] features use sales and holiday information for the near future, store accessibility [**SA**] features uses the *occurrences* of the customer before making this visit, and group movement [**GM**] features considers other visits at the same time.

For seven stores, the total number of generated features varies from 220 to 866 depending on the number of areas and the number of semantic levels used. $\mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4-$ level features are generated for two stores: E_GN and E_SC, which we continuously tracked their floor plans during data collection periods. In Table 3.1, we introduce 20 *representative* features to best describe the characteristic of each feature group. On the right side of the table, the corresponding semantic level for each feature is marked. The detailed number of features for each group and each semantic level is listed in Table 3.2, for two stores E_GN and E_SC.

Figure 3.1 and Figure 3.2 display meaningful relationships between the feature values of f_1, f_7, f_9, f_{15}, and f_{17} with the average revisit intention rate $E[RV_{bin}(v)]$. By dividing total visits into 20 equal bins according to feature value, we can identify the association between feature values and revisit rates without being affected by outliers.

[3] For ease of exposition, this setting is specific to the datasets we have. The set of levels depends on the dataset and application in hand as well as is orthogonal to our framework.

Table 3.1: Description of the representative features according to the data sources and feature groups. The ✓ indicates the best semantic level to describe each feature. For features with multiple ✓, the values of the features generated at each level are different, thus having different meanings.

Data sources	Data sources (low-level)	Feature groups	Twenty representative features (Among 866 features of store E_GN)	Semantic level of features					
				Sensor	Category	Floor	Gender	In/Out	None
Moving pattern of the visit	From the entire trajectory	Overall statistics **[OS]**	f_1 = Total dwell time					✓	
			f_2 = Trajectory length	✓	✓	✓	✓		
			f_3 = Skewness of dwell time of each area	✓	✓		✓		
		Travel Distance/ Speed/Acceleration **[TS]**	f_4 = Total distance traveled inside the store		✓				
			f_5 = Speed based on transition time	✓	✓	✓	✓		
			f_6 = First-k HWT coefficients of acceleration	✓	✓	✓	✓		
		Area preference **[AP]**	f_7 = Coherency of dwell time for each level		✓	✓	✓		
			f_8 = Top-k-area dwell time	✓	✓	✓	✓		
		Entrance and Exit pattern **[EE]**	f_9 = Exit gate	✓					
			f_{10} = Daily visit count of the customer					✓	
		Heuristics **[HR]**	f_{11} = Wears clothes but does not buy		✓				
	From the subsequence	Statistics of each area **[ST]**	f_{12} = Number of time sensed in the area	✓	✓	✓	✓		
			f_{13} = Stdev of dwell time for the area	✓	✓	✓	✓		
Temporal information of the visit	From the time of visit and events calender	Time of visit **[TV]**	f_{14} = Day of the week						✓
		Upcoming events **[UE]**	f_{15} = Remaining day until the next sale						✓
			f_{16} = Number of holidays for next 30 days						✓
Occurrences before the visit	From access intervals	Store accessibility **[SA]**	f_{17} = Number of days since the last access					✓	
			f_{18} = Average interarrival time					✓	
Simultaneous visits	From the entrance and exit time	Group movement **[GM]**	f_{19} = Presence of companions					✓	
			f_{20} = Number of companions					✓	

Table 3.2: The number of features per group for two stores, E_GN and E_SC.

(a) Number of features for each group.

Feature groups	Store E_GN	Store E_SC
[**OS**]	121	121
[**TS**]	90	90
[**AP**]	64	59
[**EE**]	3	3
[**HR**]	8	8
[**ST**]	**516**	**318**
[**TV**]	31	31
[**UE**]	7	7
[**SA**]	24	24
[**GM**]	2	2
ALL	866	663

(b) Num. of features for each semantic level.

Semantic levels	Store E_GN	Store E_SC
Sensor level	297	189
Category level	244	236
Floor level	131	68
Gender level	82	82
In/Out level	74	50
(Temporals)	38	38
ALL	866	663

3.2.2 Feature Descriptions

In this section, we introduce the detail of each feature group used in our model. With the background information for designing each feature, we show some correlations between features and customer revisits.

Overall Statistics [OS]

[**OS**] features represent the high-level view of a customer's indoor movement patterns, and therefore, the predictive power is relatively strong. By considering the trajectory as a whole, we can extract features such as total dwell time ($f1$), trajectory length (f_2), and average frequency of each area. We also apply skewness (f_3) or kurtosis to measure asymmetric or fat-tail behavior of the dwell-time distribution of each area.

Travel distance, Speed and Acceleration [TS]

[**TS**] features are in-depth information that need to be explored [85]. To approximate the physical distance (f_4) traveled by the customer, we created a network based on the physical connectivity between areas. We used the transition time to obtain the shopping speed (f_5), and we modeled the acceleration from the speed variation between consecutive areas. A time series analysis using the Haar Wavelet Transform (HWT) [104] was performed, as well as statistical analysis, to determine how the customer's interests changed with time. We included the first-16 HWT coefficient (f_6) in our feature set.

Area Preference [AP]

With [**AP**] features, it is possible to identify the difference between a customer viewing a specific area with a high degree of concentration and a person shopping lightly throughout the store. The area name and dwell time (f_8), and its proportion over total dwell time of the top-3 areas of each level are included in the basic features. The coherency of each level (f_7) determines the consistency of the customer's behavior. The definition of the coherency metric is the proportion of time spent in the longest

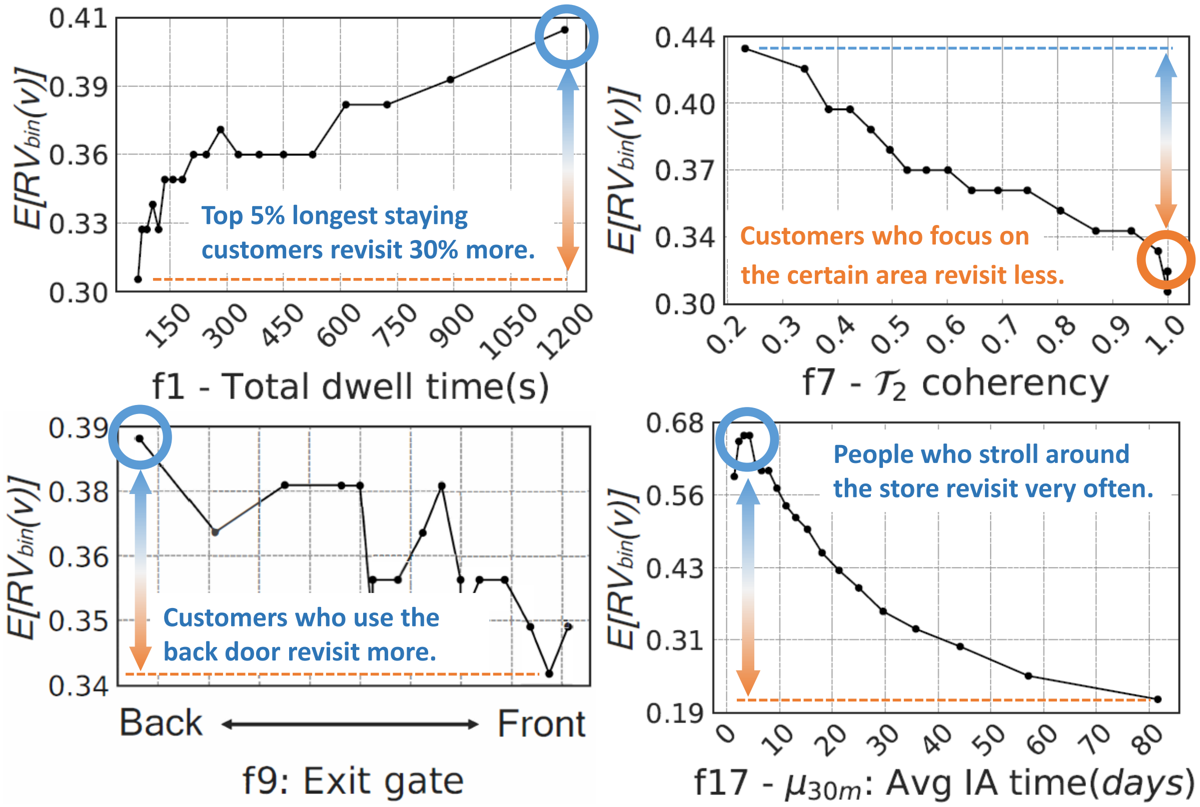


Figure 3.1: The relationship between the selected features and RV_{bin} in store E_SC ($E[RV_{bin}(v)\ (v \in V_{all})] = 0.3616$). Each marker point represents the average revisit intention $E[RV_{bin}(v)]\ (v \in V_b)$ of the set V_b of visits obtained by equal-frequency-binning the entire data according to feature values. Indoor moving pattern features f_1, f_7, and f_9 shows at most 40 % deviation of $E[RV_{bin}(v)]$ according to the feature value. The store accessibility feature f_{17} shows 325 % deviation, which is the highest among the selected features. For f_9, the group of customers who are most likely to use the back door are located on the left side of the x-axis.

staying area. This metric is effective to capture regular customers who know the store's layout and go directly to the desired area.

Entrance and Exit pattern [EE]

Interestingly, customers leaving through the back door (f_9) revisited 13.6 % more than customers leaving through the front door, according to our data. Therefore, we positioned several sensors nearby the front and back doors to note their entrance and exit patterns, and use the estimated values as features. We expected that customers familiar with the store might have used a more convenient door for their next destination. Next, we add the number of previous reentries on that day (f_{10}) as a promising feature. For this feature, we used 10 min threshold to define independent visit, from the multimodal distribution explained in Figure 2.6. As the number of daily entrances increases by one, the rate of further revisits increases by 1 %.

Heuristics [HR]

To fully exploit the relation between customer trajectories and revisits, we interviewed the managers and part-timers of the stores to get intuitions on what kinds of patterns are likely to appear from the customers who are willing to revisit. In general, the interviewees agreed that staying in certain areas, trying an item, and purchasing or postponing the item can reflect customers' interest and purchase pattern that lead to revisits. These steps of actions, in fact, correspond to online shopping activities—i.e., browse, add to cart, checkout, and then revisit or churn [72]. As we do not know whether a customer actually tried an item in the fitting room or purchased it, we inferred those actions by tracking the dwell time in the fitting room and the checkout counter. Here are two representative heuristics anticipating the revisit of customers for future purchase.

- If a customer wears clothes in the fitting room without purchase ($\leq 1\,\text{min}$ in the checkout counter): $f_{11} = 1$, for all other cases: $f_{11} = 0$.
- If a customer stays in the store much longer ($= 10\,\text{min}$) than average visitors [37], without purchase: $f = 1$, if not: $f = 0$.

The reasons for these associations are as follows. If the customer tries an item or stays in the store for a long time, he/she is prone to purchase the item. However, the fact that the customer does not purchase the item right away implies that there is a possibility of purchasing that item at the next visit.

Statistics of Each Area [ST]

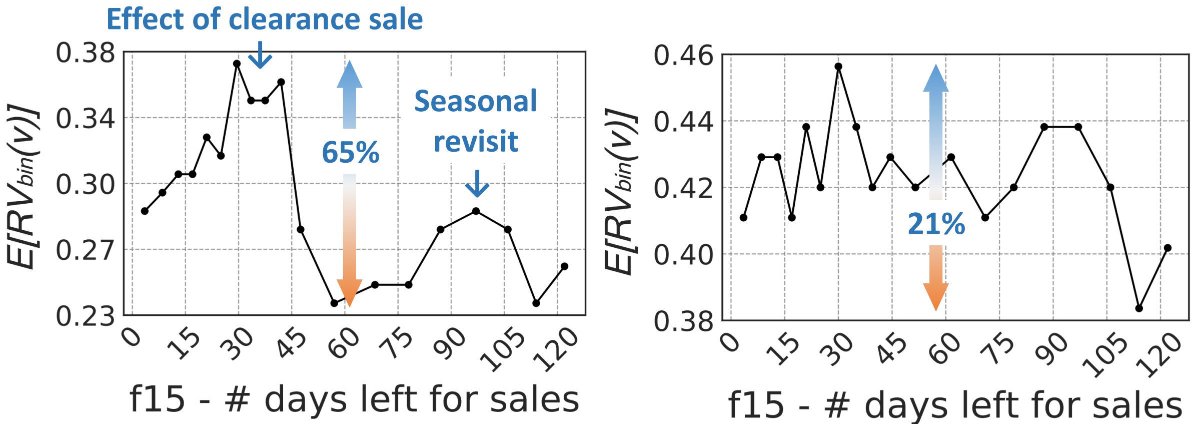
If a certain semantic area is highly relevant to revisit, the statistics from that area have higher predictability. For all semantic areas, we created six features including the number of times it was sensed (f_{12}), the percentage of the total time spent in the area (that is used for developing the coherency feature), and the standard deviation of the times sensed in the area (f_{13}). As explained before, the difference in the total number of features is mainly due to the difference in the number of areas that each store has. In our final model, the main difference in the total number of features originates from the difference in the number of zones. (Table 3.2)

Time of Visit [TV]

The temporal features include the time of visit such as the hour of the day and day of the week (f_{14}) as basic features. The feature values described above are ordinal and thus were transformed into multiple binaries by one-hot encoding. The value of a temporal feature is determined by the entrance time.

Upcoming Events [UE]

Customers are more likely to visit a store in the period of a clearance sale. However, they are less likely to visit the fashion district in the holiday seasons(e.g., Spring Festivals, Thanksgiving week) since they are out of the city center. For example, customers who visited one month before the clearance sale have higher chance to revisit since they would like to get a discount during the upcoming sales. By combining simple extrinsic information, the temporal features, particularly **[UE]**, becomes the second strongest predictive feature groups. It contains six features, including a number of days left for the next clearance sale (f_{15}) and a number of holidays for next 30 days (f_{16}), as numeric features.



(a) First-time visitors (v_1): Prone to events.

(b) All visitors: Indifferent to events.

Figure 3.2: Key factors of v_1's revisit: discount and seasonality.
Discount-sensitive: A set V_b of customers who visited between 30–45 days before a clearance sale showed a high $E[RV_{bin}(v)]$ ($v \in V_b$) compared to other customers; this difference was more apparent in first-time visitors than all visitors.
Seasonal-sensitive: Another peak of $E[RV_{bin}(v)]$ appeared on the set of customers who made a visit between 90–105 days before the sale. It described the seasonal revisit, and it was also more noticeable to first-time visitors than all visitors.

Store Accessibility [SA]

When installing sensors inside the store, could you imagine that the weak noise collected outside the store would provide the most important clue to predict revisit? Surprisingly, the revisit predictability increased dramatically when we included **[SA]** features using weak signals, which could have been overlooked as mere noises. The following settings are expected to be applicable to many studies when conducting research using in-store signals that do not contain customer address information.

The features are designed to capture various aspects from interarrival times. We utilized two additional outdoor areas nearby the store—5 m and 30 m zone—to detect the customer occurrences. Considering a customer's arrival process to 5 m zone, let us denote the time of the first occurrence by T_1. For $k > 1$, let T_k denote the elapsed time between $k - 1^{\text{th}}$ and the k^{th} event. We call the sequence $\{T_k, k = 1, 2, ..., \}$ as the *sequence of interarrival times.* Considering the target visit as n^{th} event of the arrival process, we use the following features:

- $n - 1$: Number of occurrences before the visit;
- T_n: Number of days from the last occurrence (f_{17});
- $\mathbb{1}_{n>1}$: Existence of having any occurrence before the visit;
- $\mu = \sum_{k=2}^{n} T_k/(n-1)$: Average interarrival time (f_{18});
- $\sigma = \sqrt{\sum_{k=2}^{n}(T_k - \mu)^2/(n-1)}$: Standard deviation of interarrival times;

In addition to these five features from T_k, we added the average sensed time for previous occurrences.

Group Movement [GM]

Unlike previous features, **[GM]** features were extracted by considering multiple trajectories. This is a representative feature that can only be captured by analyzing surrounding trajectories that happened simultaneously with the main trajectory. In our feature extraction framework, we considered the presence of companions (f_{19}) and the number of companions (f_{20}). One of the biggest characteristics of judging whether or not to be a companion is to enter the store at the same time. Based on the information obtained through the field study, we considered that two visitors are in a group when their entrance time and exit time are both within 30 s.

We decided 30 s group movement threshold by the following logic. According to our observation at store E_GN in the afternoon of June 24 and June 26, 2017, 56 % of 105 customers entered the store with their companions, which was more than half. Considering $p_x = 39.2\,\%$ as the on-site Wi-Fi turn on rate (Always-on: 29.2 %, Conditionally-on: 10 %) [84] and $p_y = 56\,\%$ as the actual proportion of customers in a group, we expected that $p_{yo} = 15.5\,\%$ of the total visitors were represented as having companions in our collected data of store E_GN (by Eq. 3.3 later in Section 3.3.3). By setting 30 s as a threshold of accompaniment, we also obtained 15 % of the total visitors were considered as having companions in the same data. By considering a gap between actual group ratio and observed group ratio, we claim that 30 s is an appropriate threshold to distinguish group movement.

3.2.3 Unused Features

Some potentially useful features were not included in our final model because their effect on the accuracy was marginal. However, we would like to mention them since they could be useful in other types of predictive analytics [61, 72].

Sequential Patterns

Semantic trajectories of customers can be represented as sequential patterns. But sequential patterns [27, 61] were not effective for the revisit prediction task on our datasets, so we omitted them from the final framework. To briefly describe our approach, we retrieved top-k discriminative sequential patterns by the information gain and generated k features. Each feature $f_i(v)$ denotes the number of times a trajectory of visit v contains i^{th} patterns. Starting from the simplest association rule mining, we elaborated a partial sequence mining technique from [61], by incorporating discretized interval between areas. We discretized a dwell time and an interval into four levels: *veryshort*, *short*, *medium*, and *long*. With this approach, our pattern can hold temporal information, which is expected to be more effective than [27] considering continuous intervals. Table 3.3 shows an example of each level of sequential patterns. We attempted to use diverse level of sequential patterns but the result was not satisfactory. Despite that it was expensive to generate the features, their information gains were typically low.

We developed a software[4] by modifying an open-sourced frequent sequence mining package[5]. The output of the reference software were a set of partial sequence pattern without time constraints, $A \overset{*}{-\,-\!\!\rightarrow} B \overset{*}{-\,-\!\!\rightarrow} C$. We added three functionality to advance the software. First, we modified the algorithm to consider time constraints for intervals. Second, we added fast counting module for feature engineering, to calculate feature values for both training & test sets. Third, we made our software to calculate information gain as well as support, to find top-k discriminative partial sequences at once. Although we exclude sequential pattern features from our final model, we believe it is worthwhile to report our effort in the thesis.

[4] https://github.com/Seondong/mtraj

[5] https://github.com/bartdag/pymining

Table 3.3: Types of sequential patterns.

Pattern type	Description
ABC	A pattern without an order, obtained by association rule mining.
$A \rightarrow B \rightarrow C$	A sequential pattern having an order, where the following element appears immediately after the previous element.
$A \overset{*}{\dashrightarrow} B \overset{*}{\dashrightarrow} C$	A partial sequential pattern [61], an arrow $A \overset{*}{\dashrightarrow} B$ denotes that there might exist additional elements between A and B.
$A_{short} \overset{*}{\dashrightarrow} B_{long} \overset{*}{\dashrightarrow} C_{medium}$	A partial sequential pattern which has a time constraint for the dwell time of each element.
$Enter \overset{veryshort}{\dashrightarrow} A_{short} \overset{short}{\dashrightarrow} B_{long} \overset{medium}{\dashrightarrow} C_{medium}$	A partial sequential pattern with time constraint for the dwell time of the element and the interval between elements.

Past Indoor Information

We excluded the features that average up the customer's previous indoor mobility statistics, as well as those that represent the amount of changes from past statistics [72]. By nature, the number of features becomes doubled per revisit by considering that information. However, they were not a strong indicator of revisits unlike [**SA**] and thus were removed.

Features That May Interfere with Fair Evaluation

Since most customers have a small number of visits, we developed a general model that considers the mobility of the entire set of customers. According to this principle, we considered each visit separately, by removing customer identifiers. In this way, we can also ensure that our model is robust to general cross-validation settings. We excluded the visit date to avoid a biased evaluation that favors the customers who visited in an earlier period. We also ignored the explicit visit count information.

3.3 Experiments

In our experiments, we verify that our feature set designed from customer mobility patterns is effective in predicting customer revisit, especially for newcomers. In addition, we verify the performance of individual feature groups and semantic levels. Throughout the discussion section, we provide more detailed analyses regarding the revisit prediction. The key contents include the demonstration of the performance change over the length of data collection period and model robustness on missing customers. We conclude this section by sharing the difficulties of securing accuracy in line with the gap between the predictive power and the statistical significance of each feature.

3.3.1 Settings

Prediction Tasks

We designed prediction tasks to explore customers' revisit behaviors. The first task is a binary classification task to predict customers' revisit intention RV_{bin}. The second task is a regression task to predict the revisit interval RV_{days} between two consecutive visits. For each task, we conducted experiments on two different data subsets. First, we see the performance of our model on the entire customer dataset. Second, we used a dataset consisting of only the first-time visitors to show that our prediction framework is effective in determining the willingness of first-time visitors to revisit.

Scoring Metrics

We used two scoring metrics: *accuracy* and *root mean squared error (RMSE)* for the classification and regression tasks, respectively.

- The *accuracy* is the ratio of the number of correct predictions to that of all predictions. We used it for the classification task because it is considered to be the most intuitive metric for store managers and practitioners. To fairly compare the model performance in seven imbalanced datasets with different revisit rates, we downsampled non-revisited customers for each dataset. In this way, we designed the task as a binary classification on balanced classes having 50 % as a random baseline. To mitigate the risk of the sampling bias, we prepared *ten* different downsampled train/test sets with random seeds. The averages of ten executions were reported in the paper.
- The RMSE is measured between the actual interval and the predicted interval. To make the RMSE values of two stores with different data collection periods comparable, a RMSE value was normalized by the length T of the data collection period, providing the same result as the RMSE calculated by considering the ratio of an interval to the total period $y_i^* = y_i/T$, as follows:

$$\begin{aligned}\frac{RMSE}{T} &= \frac{1}{T}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_i)^2} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\frac{y_i}{T} - \frac{\bar{y}_i}{T})^2} \\ &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i^* - \bar{y}_i^*)^2} = RMSE^*.\end{aligned} \tag{3.1}$$

  Because we cannot calculate the revisit interval for the last visit, we excluded the customers' last visits for the regression task.

Data Preparation

The training and testing data were prepared with three settings:

- S1: 5-fold cross-validation by dividing *customers*, where each customer data can be included only in a single fold.
- S2: 5-fold cross-validation by dividing *visits*[6], where each visit is handled independently.
- S3: First 50 % visits as the training data, and other 50 % as the testing data.

The accuracy difference between S1 and S2 was insignificant to the fourth decimal place. In S3, there was an accuracy loss of about 2.5 % on average compared to S1 and S2, due to floor plan changes of the stores and inaccurate labels caused by truncation in time (Section 3.3.3). Because of the page limit, we report the main results using the configuration S1.

[6] As a result of Section 3.2.3, our model is considered to be safe to perform cross-validation.

For all settings, we did not consider the data which occurred in the final two weeks of the data collection period, because their revisit behavior might not be captured owing to the termination of the data collection. In the later chapter, we introduce survival analysis and how to learn correctly from these *partial observations*. The experiment is performed on the entire trajectory collected, and we have not made any efforts to cherry-pick parameters to find well-performing data.

Classifier

All results described in this section were obtained using Python API of the XGBoost [14] library that optimized the gradient boosting tree [25] framework. In our preliminary experiments, XGBoost gave the *best* performance among logistic regression, decision trees, random forests, AdaBoost, and gradient boosting trees implemented in the Python Scikit-learn [86] library.

We used *all* features for training and testing the model, since using all features gives the best performance and the boosting tree classifier is robust to potential correlations between features. The elapsed time for each fold with 200,000 visits and 660 features took no longer than 1 min in a single machine (Intel i7-6700 with 16 GB RAM, without GPU). Besides, we did *not* focus on fine-tuning the prediction model, but used the basic hyperparameter settings. Obviously, our prediction framework can benefit from *any* state-of-the-art classifier.

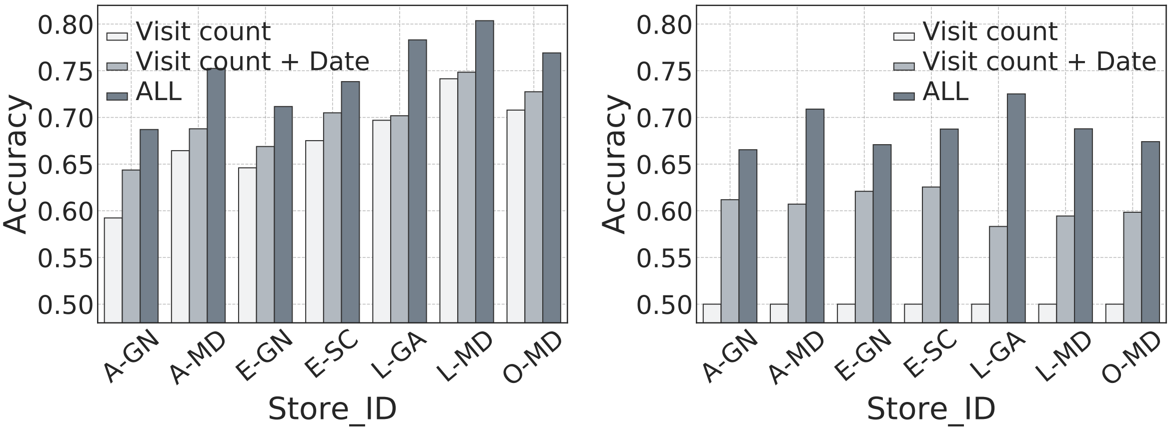
Table 3.4: Performance of classification and regression tasks.

Store	Period (days)	# features	Customer type	# data (# revisitors)	Accuracy	RMSE
A_GN	222	256	First	99,497 (9,514)	0.6336	0.2132
			All	112,672 (13,222)	0.6689	0.2000
A_MD	220	328	First	223,103 (47,917)	0.6930	0.1865
			All	327,940 (104,913)	0.7412	0.1622
E_GN	300	866	First	144,610 (21,701)	0.6663	0.1862
			All	183,246 (38,817)	0.7050	0.1627
E_SC	373	663	First	172,551 (41,036)	0.6818	0.1824
			All	270,366 (98,818)	0.7288	0.1475
L_GA	990	244	First	838,241 (107,925)	0.7173	0.1403
			All	1,062,226 (225,409)	0.7789	0.1244
L_MD	747	220	First	1,154,486 (197,476)	0.6799	0.1416
			All	1,718,359 (566,701)	0.7991	0.1146
O_MD	698	316	First	1,033,253 (294,949)	0.6645	0.1311
			All	2,008,384 (978,699)	0.7599	0.1028

3.3.2 Results

Overall Results on Seven Stores

Table 3.4 shows the overall accuracy and RMSE using XGBoost classifier. First, the prediction accuracy for first-time visitors is 67 % averaged over seven stores. By only using mobility data captured by in-store sensors, *two* out of *three* customer's revisit is predictable without having *any* historical data in the store. Second, the average prediction accuracy increases to 74 % by considering all customers. The primary reason for this result is that the dataset of all customers contains *frequently* visiting customers,



(a) On all visitors.

(b) On first-time visitors.

Figure 3.3: Effectiveness of analyzing customer trajectories.

who are mostly predicted to revisit. Third, the stores with a long data collection period and abundant user logs generally show high performance, while this trend might not happen depending on the characteristics of the stores. Last, the regression performance shows a similar tendency with classification performance. For instance, RMSE error of store L_GA, L_MD and O_MD are much lower than the first four stores. As shown in Eq. (3.1), the RMSE here can be explained as a mean error rate with respect to the data collection period, the meaning of the error 0.1028 is that the difference between predicted and true interval divided by the data length is 10.28 % in average.

Performance Improvement by Analyzing Trajectories

To measure the performance improvement using our features, we developed two different baselines for comparison. The first baseline is a theoretical lower bound of the prediction accuracy obtained by only using revisit statistics conditioned on the number of (prior)[7] visits. Since we fully ignored any other information here, the prediction accuracy with this limited information must be lower than that of using full trajectories. To continue the flow, the procedure of deriving lower bounds will be introduced at the end of this section.

The second baseline is a model to which the visit date is added. Since our task utilizes finite time-series datasets with time-dependent objectives, the earlier collected logs tend to have the higher revisit rate. Therefore, by including a visit date as an additional feature, the baseline accuracy improves naturally. If there existed infinite data, the performance increase by this factor would disappear. The benefit of using customer mobility can be considered as the gap between our final model and the second baseline.

Figure 3.3 reports the accuracy of our model[8] against two baselines. We note that our final model is more effective than the second baseline by 4.7–11.6 % in terms of accuracy. Among the first-time visitors, the effectiveness of trajectory analysis increases, showing a performance improvement of 8.0–24.3 %.

[7] We said ‘prior’ to make reader understand the concept easily. Actually, we used the number of visits including the current one.

[8] For this experiment, we included visit count and date to our feature set, so the overall accuracy is slightly higher than the values reported from Table 3.4.

Table 3.5: Prediction accuracy conditionally measured on groups of customers with the same number of visits. We only reported the result where $|v_n| \geq 50$ on the test set.

Store ID	A_GN	A_MD	E_GN	E_SC	L_GA	L_MD	O_MD
# visits							
v_1	0.661	0.741	0.681	0.716	0.763	0.778	0.758
v_2	0.732	0.735	0.716	0.691	0.795	0.773	0.706
v_3	0.824	0.786	0.791	0.751	0.840	0.848	0.757
v_4	0.856	0.808	0.845	0.800	0.848	0.879	0.801
v_5	-	0.803	0.865	0.831	0.847	0.885	0.820
v_6	-	0.810	0.884	0.852	0.846	0.883	0.829
v_7	-	0.808	0.907	0.861	0.856	0.879	0.834
v_8	-	0.814	0.911	0.866	0.836	0.878	0.838
v_9	-	0.802	0.903	0.875	0.863	0.874	0.837
v_{10}	-	0.789	-	0.900	0.867	0.870	0.839

Prediction Accuracy According to the Number of Visits

For further analysis, we measured the prediction accuracy for each customer group determined by their number of visits. For this experiment, we used the model trained on all customers.

Customers who visit more than a certain number of times usually have a high chance to revisit. Thus, we expect that our model can predict their revisits with high accuracy. The results in Table 3.5 confirm this expectation. As customers visited more often, the prediction accuracy tended to increase in all stores. Interestingly, we found that the prediction accuracy sometimes was the lowest in the case of v_2 since those groups of customers seemed to have the most uncertain behavior on their revisits.

Table 3.6 shows the improvement of our model compared with the two baselines in Section 3.3.2 for each customer group. It indicates that our model is more effective than the baselines by over 10 %, especially on v_1 and v_2. Thus, our feature set is shown to be effective in predicting customers' revisits even when they are newcomers.

Deriving the First Baseline Analytically The lower bounds can be derived either experimentally or analytically. Here is how we derived it analytically. The visit logs v_k with the same visit count k are considered to have the same information. To maximize the accuracy, we must predict the label l of v_k by the following criteria:

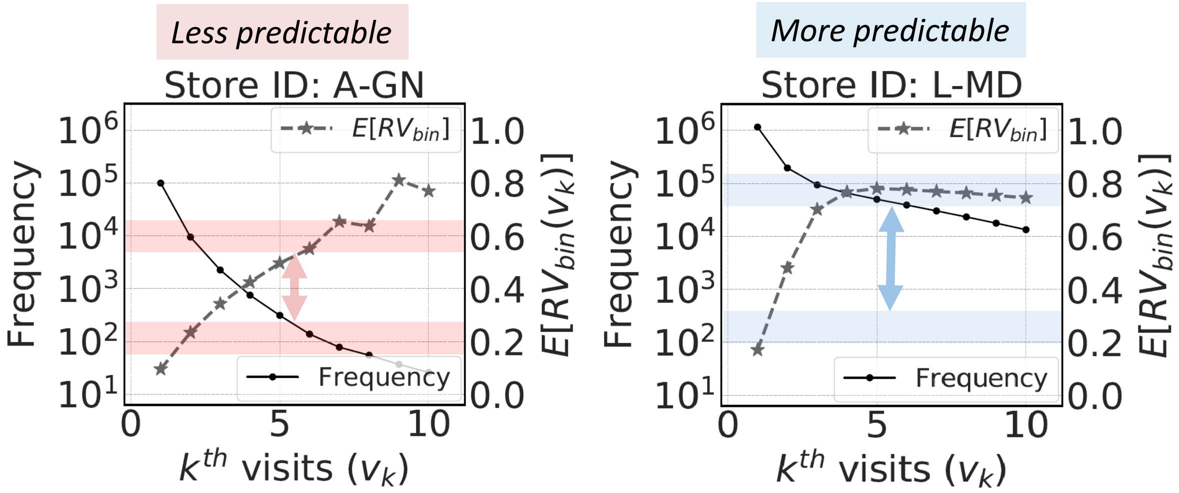
$$\forall v : l(v \in v_k) = \begin{cases} 1, & \text{if } E[RV_{bin}(v_k)] \geq 1/2 \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

Considering each proportion $p_k = |v_k|/\sum_k |v_k|$ and simplifying $E[RV_{bin}(v_k)]$ as r_k, the lower bound accuracy of a model can be represented as $LB = \sum_k p_k \cdot \max(r_k, 1 - r_k)$. In the experiment of only first-time visitors, $LB = 1/2$ since $p_1 = 1$ and $r_1 = 1/2$.

The interpretation with the lower bound is as follows. For higher predictability, the revisit tendency of each v_k should be homogeneous. In Figure 3.4, we can notice that store L_MD is more predictable than A_GN, because $|r_k - 0.5|$ of L_MD is larger than that of A_GN for the majority of k.

Table 3.6: Improvement of our model against the two baselines. The first number represents the improvement of prediction accuracy over the first baseline (%), and the second number represents the improvement over the second baseline (%).

Store ID	A_GN	A_MD	E_GN	E_SC	L_GA	L_MD	O_MD
# visits							
v_1	18.6/7.7	17.1/14.7	12.9/9.1	10.4/7.1	18.2/17.6	10.5/10.4	7.6/7.4
v_2	4.9/1.2	13.5/5.0	7.5/2.0	15.1/3.1	4.6/3.0	18.4/12.5	29.7/13.0
v_3	1.7/0.4	4.2/1.3	3.0/0.4	7.5/1.3	0.9/0.3	2.5/1.2	8.0/3.5
v_4	1.3/0.3	3.5/0.5	2.8/1.1	5.5/0.7	1.0/0.1	0.9/0.2	3.7/1.0
v_5	-	3.2/0.3	1.3/-0.4	3.8/0.8	1.1/0.1	0.7/0.0	2.7/0.5
v_6	-	2.3/0.2	1.6/0.8	3.3/0.4	1.3/0.2	0.8/0.0	2.4/0.2
v_7	-	3.8/0.8	1.8/-0.1	2.7/1.0	1.3/0.3	0.8/0.0	2.2/0.2
v_8	-	4.0/-0.2	1.7/0.5	2.4/0.0	1.4/0.2	1.2/0.0	2.2/0.2
v_9	-	3.6/0.0	1.5/0.9	3.2/0.6	1.8/0.6	1.4/0.2	2.0/0.0
v_{10}	-	3.1/0.0	-	2.1/0.2	0.9/0.2	1.6/-0.1	2.5/0.2



(a) The case of a less predictable store with LB 0.595.

(b) The case of a more predictable store with LB 0.741.

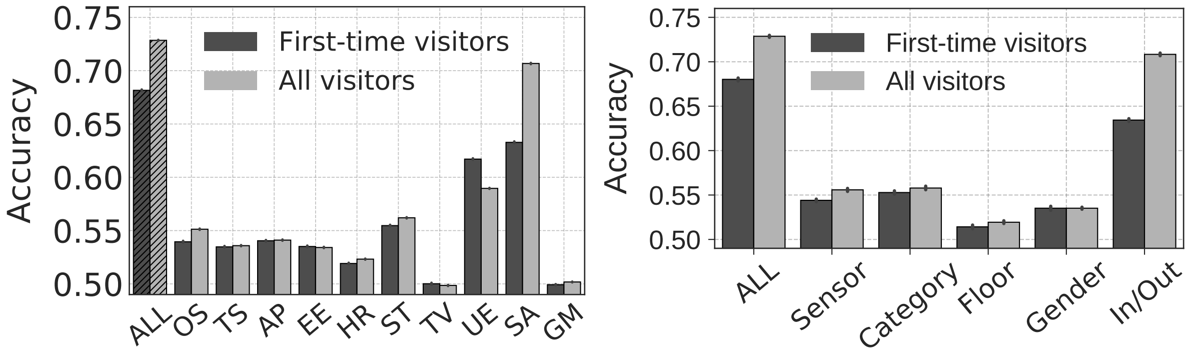
Figure 3.4: Lower bound accuracies of two stores.

Predictive Power of Feature Groups

Figure 3.5(a) investigates the predictive power of each group of features in store E_SC. Each bar corresponds to the prediction results using the features of only a specific group. The labels of the x-axis are the abbreviations of the feature groups categorized in Table 3.1. For the convenience of comparison, the leftmost bar on the figure represents the results when all features in Table 3.4 are used. It was observed that the *store accessibility* [**SA**] features have the strongest predictive power, especially for the prediction with all visitors, followed by the *upcoming event* [**UE**] features for the first-time visitors.

Predictive Power of Semantic Levels

As opposed to our intuition, a performance of semantic levels inside the store did not boost the performance that much. As in Figure 3.5(b), the performance of the features generated from the category level ($\mathcal{T}_2$) only beats the features from the sensor level ($\mathcal{T}_1$). Besides, the semantic trajectories generated from the floor-level ($\mathcal{T}_3$) and the gender level ($\mathcal{T}_4$) were not effective to predict customer revisit in the store E_SC. We can conclude that finding effective trajectory abstraction is difficult even if the hierarchical information is provided.



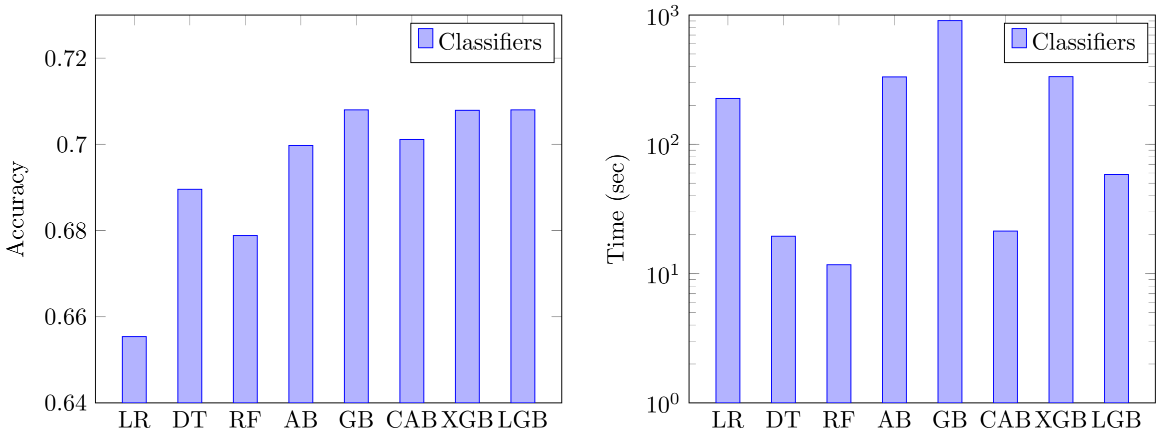
(a) On feature groups.

(b) On semantic levels.

Figure 3.5: Performance comparison on feature groups and semantic levels for store E_SC. Each bar represents the predictive power that can be obtained when using only the feature group or the semantic level shown on the x-axis. In (a), store accessibility [**SA**] and upcoming events [**UE**] show strong predictive power. In (b), [**In/Out**] level shows the strongest predictive power followed by [**sensor**] and [**category**] level. (Acronym in (a): [ALL] = All features, [SA] = Store accessibility, [UE] = Upcoming events, [ST] = Statistics of each area, [OS] = Overall statistics, [AP] = Area preference, [SP] = Speed/Acceleration, [EE] = Entrance and exit pattern, [HR] = Heuristics, [TV] = Time of visit, [GM] = Group movement.)

Comparison On Various Classifiers

During the revision, we also compared the performance of the XGBoost results with up-to-date boosting classifiers such as LightGBM [49] and CatBoost [89], and LightGBM was 5.7 times faster than CatBoost with similar performance. To further improve performance, we also tried a two-level stacking by incorporating the top-3 individual models, but the performance improvement was marginal. We first introduce the performances between *eight* classifiers. We used default parameter settings for classifiers and some tuned parameters are listed below.



(a) Performance comparison.

(b) Running time comparison.

Figure 3.6: Comparison between classifiers. LGB turns out to be the most effective among all classifiers. (a) Average accuracy on all experiments, (b) Average running time on all experiments.

- Classifiers provided by Scikit-learn [86][9]. The parameters used are summarized as follows.
  - LR (Logistic Regression): default settings.
  - DT (Decision Tree): max_depth = 4.
  - RF (Random Forests): n_estimator = 10.
  - AB (AdaBoost): default settings.
  - GB (Gradient Boosting): max_depth = 4.
- Up-to-date boosting classifiers:
  - CAB (CatBoost): depth = 4, learning_rate = 0.1, iterations = 30.
  - XGB (XGBoost): max_depth = 4, learning_rate = 0.1.
  - LGB (LightGBM): max_depth = 4, learning_rate = 0.1.

Figure 3.6 summarizes the comparison results for the eight classifiers in terms of prediction accuracy and running time. To obtain stable results, we repeated 5-fold cross-validation 25 times and then reported the averages by aggregating the results of the seven stores. As a result, LGB turned out to be the fastest classifier among the three best-performing classifiers—GB, XGB, and LGB. CAB was very fast as well as gave comparable results. Interestingly, DT took more time than RF and showed a better result in the default setting. Table 3.7 shows the details of Figure 3.6 by showing the accuracy for each of the seven stores. The mean and standard deviation were calculated from the average accuracies of 25 different 5-fold cross-validations.

Comparison On Stacking Models

To achieve additional performance improvement, we applied stacking (meta ensembling) with eight strategies. *Stacking* is a model ensembling technique used to combine multiple predictive models to generate a better model [113]. Usually, the stacked model is known to outperform each of the individual models owing to its smoothing nature and its ability to highlight each base model. The main point of the stacking is to utilize the prediction results of the base models as features for the stacking model in the second layer.

[9] Scikit-learn 0.20, which is the latest version at the time of this submission, was used for the experiments.

Table 3.7: Prediction accuracy (%) of various classifiers for the revisit prediction task.

(a) Experimental results on the models trained by first-time visitors.

Store ID	A_GN	A_MD	E_GN	E_SC	L_GA	L_MD	O_MD
LR	59.56±0.27	66.39±0.13	61.80±0.22	60.94±0.21	69.08±0.08	65.11±0.09	63.48±0.07
DT	62.42±0.26	66.42±0.11	64.97±0.25	66.31±0.09	69.98±0.06	65.33±0.05	63.94±0.04
RF	61.63±0.25	66.74±0.13	61.84±0.32	62.50±0.20	69.34±0.16	65.57±0.11	63.58±0.13
AB	62.51±0.31	68.52±0.12	65.39±0.16	66.83±0.14	71.05±0.06	67.26±0.05	65.68±0.03
GB	63.13±0.20	69.30±0.10	66.69±0.19	68.29±0.10	71.83±0.06	67.77±0.05	66.21±0.04
CAB	63.12±0.27	68.43±0.11	65.78±0.18	67.44±0.10	70.84±0.07	66.94±0.05	65.32±0.05
XGB	63.14±0.23	69.29±0.10	66.67±0.15	68.28±0.10	71.79±0.06	67.76±0.04	66.19±0.03
LGB	63.18±0.25	69.31±0.11	66.68±0.18	68.28±0.11	71.80±0.06	67.77±0.05	66.19±0.03

(b) Experimental results on the models trained by all visitors.

Store ID	A_GN	A_MD	E_GN	E_SC	L_GA	L_MD	O_MD
LR	61.58±0.34	69.30±0.16	62.43±0.48	60.15±1.43	72.64±0.05	75.41±0.12	69.69±0.20
DT	66.10±0.27	72.18±0.05	68.30±0.11	70.85±0.05	76.38±0.04	78.29±0.04	73.98±0.01
RF	65.13±0.26	71.38±0.10	66.91±0.24	68.84±0.20	75.68±0.11	77.74±0.20	73.47±0.18
AB	66.25±0.25	73.19±0.07	69.78±0.10	72.02±0.04	76.85±0.05	79.12±0.02	75.07±0.01
GB	66.67±0.21	74.11±0.05	70.69±0.09	73.06±0.05	77.87±0.03	79.75±0.07	75.82±0.01
CAB	66.62±0.23	73.53±0.05	69.96±0.10	72.15±0.06	77.14±0.04	79.11±0.09	75.16±0.01
XGB	66.69±0.21	74.09±0.06	70.67±0.07	73.05±0.05	77.85±0.03	79.74±0.08	75.81±0.01
LGB	66.70±0.20	74.10±0.05	70.69±0.09	73.05±0.06	77.86±0.04	79.74±0.08	75.81±0.01

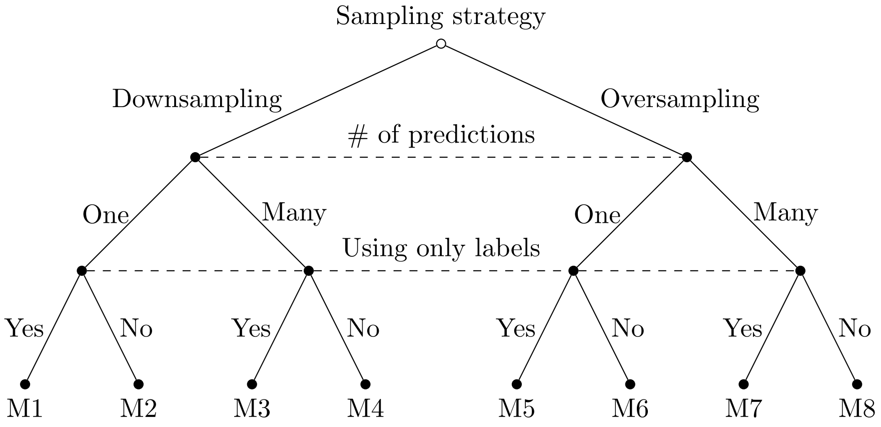


Figure 3.7: Stacking options.

Table 3.8: Prediction accuracy (%) of stacking models for the revisit prediction task with the data of all visitors.

Store ID	A_GN	A_MD	E_GN	E_SC	L_GA	L_MD	O_MD
Single Model							
LR	61.57±0.17	69.33±0.07	62.61±0.43	60.92±0.96	72.65±0.04	75.34±0.14	69.64±0.18
DT	66.17±0.26	72.21±0.03	68.33±0.17	70.86±0.06	76.37±0.03	78.27±0.01	73.98±0.01
RF	65.17±0.16	71.36±0.13	66.84±0.26	68.63±0.24	75.68±0.12	77.68±0.17	73.37±0.39
AB	66.40±0.29	73.18±0.05	69.82±0.07	72.00±0.05	76.82±0.03	79.12±0.01	75.06±0.01
CAB	66.84±0.11	73.80±0.02	70.44±0.15	72.61±0.06	77.39±0.03	79.38±0.01	75.51±0.01
XGB	66.78±0.15	74.10±0.04	70.67±0.12	73.03±0.07	77.83±0.04	79.71±0.01	75.81±0.01
LGB	66.88±0.23	74.11±0.03	70.64±0.13	73.04±0.07	77.83±0.02	79.71±0.01	75.82±0.02
Stacking Model							
M1	66.56±0.12	73.88±0.03	70.49±0.07	72.91±0.09	77.56±0.02	79.52±0.01	75.66±0.01
M2	66.70±0.13	73.95±0.03	70.52±0.08	72.95±0.08	77.62±0.02	79.59±0.01	75.69±0.01
M3	66.57±0.15	74.01±0.02	70.55±0.11	72.97±0.08	77.79±0.02	79.66±0.01	75.77±0.01
M4	66.78±0.22	74.07±0.02	70.65±0.11	73.07±0.06	77.82±0.02	79.69±0.01	75.80±0.01
M5	67.04±0.19	73.91±0.05	70.62±0.13	72.95±0.10	77.58±0.02	79.52±0.01	75.65±0.01
M6	67.00±0.28	73.96±0.04	70.64±0.14	72.99±0.09	77.64±0.02	79.58±0.01	75.69±0.01
M7	66.88±0.19	74.06±0.04	70.67±0.11	73.01±0.08	77.80±0.02	79.66±0.01	75.77±0.01
M8	66.97±0.15	74.10±0.04	70.71±0.11	73.10±0.07	77.83±0.02	79.70±0.01	75.80±0.01

To do this, we selected CAB, XGB, and LGB as the base models. We further separated a training set into three subsets and used two subsets to make the prediction labels for the remaining subset. The prediction labels for the testing set were also calculated together three $(={}_3C_2)$ times, and the three sets of the labels for the testing set were averaged for the final use. In this way, we generated the label features for both training and testing sets. These additional features are fed to the final LGB stacking model. We followed a general procedure from the reference[10] and added three options. Figure 3.7 illustrates the process of creating eight stacking models (M_1–M_8) through the choice of the three options. The description of the three options is as follows.

- Sampling strategy: A parameter that determines whether to use either random oversampling [63] or downsampling. This option is not directly related to the stacking, but we added it to improve the accuracy by treating the class imbalance problem.
- # of predictions: A parameter that determines whether to use one model or multiple models for each fold. The former case generates a single additional feature, and the latter case generates three additional features.
- Using only labels: A parameter that determines whether to use only the prediction labels (one or three features) or to use all existing features with the prediction labels (n+1 or n+3 features where n is the total number of hand-engineered features used).

[10] http://bit.ly/Kaggle_Guide_Stacking

Table 3.9: Elapsed time (min) of stacking models on revisit prediction task.

Store ID	A_GN	A_MD	E_GN	E_SC	L_GA	L_MD	O_MD
Single Model							
LR	0.14±0.03	2.28±0.44	1.91±0.48	2.54±0.92	2.86±2.53	10.77±1.14	18.92±6.76
DT	0.02±0.00	0.19±0.00	0.16±0.00	0.34±0.02	0.32±0.02	0.82±0.01	2.50±0.20
RF	0.01±0.00	0.09±0.00	0.06±0.00	0.13±0.01	0.19±0.05	0.53±0.00	1.51±0.18
AB	0.17±0.00	1.97±0.07	1.61±0.05	3.41±0.14	3.64±0.22	12.61±0.10	41.94±5.57
CAB	0.10±0.01	0.26±0.01	0.32±0.01	0.48±0.01	0.36±0.02	0.65±0.01	1.86±0.07
XGB	0.36±0.01	4.01±0.16	3.78±0.17	7.33±0.18	6.70±0.34	16.01±0.53	47.25±4.63
LGB	0.06±0.00	0.62±0.03	0.56±0.02	1.07±0.04	0.80±0.06	1.55±0.01	5.04±0.28
Stacking Model							
M1	0.73±0.02	7.77±0.44	7.01±0.25	13.81±0.48	12.15±0.57	29.16±0.51	83.45±10.04
M2	0.72±0.02	7.72±0.43	6.94±0.25	13.71±0.48	12.06±0.57	28.82±0.49	82.94±10.00
M3	1.11±0.02	12.16±0.69	10.93±0.39	21.47±0.74	19.25±0.98	46.00±0.57	134.76±14.39
M4	1.09±0.02	12.05±0.68	10.84±0.39	21.30±0.72	19.04±0.97	45.40±0.57	133.58±14.33
M5	5.04±0.24	16.31±0.87	25.60±1.65	23.62±0.65	49.06±2.61	72.13±2.99	87.32±8.27
M6	5.02±0.24	16.26±0.86	25.56±1.65	23.58±0.65	48.90±2.62	71.90±2.99	87.05±8.26
M7	7.82±0.25	25.57±1.21	39.86±1.91	36.90±1.19	78.76±4.07	117.41±5.02	139.91±14.17
M8	7.74±0.25	25.40±1.20	39.74±1.92	36.76±1.18	78.11±4.06	116.52±5.00	138.92±14.09

Table 3.8 and Table 3.9 shows the average accuracy results and the average running time obtained for each of the seven stores in details[11]. We observed that the performance improvement was not so high despite the long running time of the stacking model. Thus, we conjecture that each of the best-performing classifiers achieved almost the highest accuracy by itself.

3.3.3 Discussions

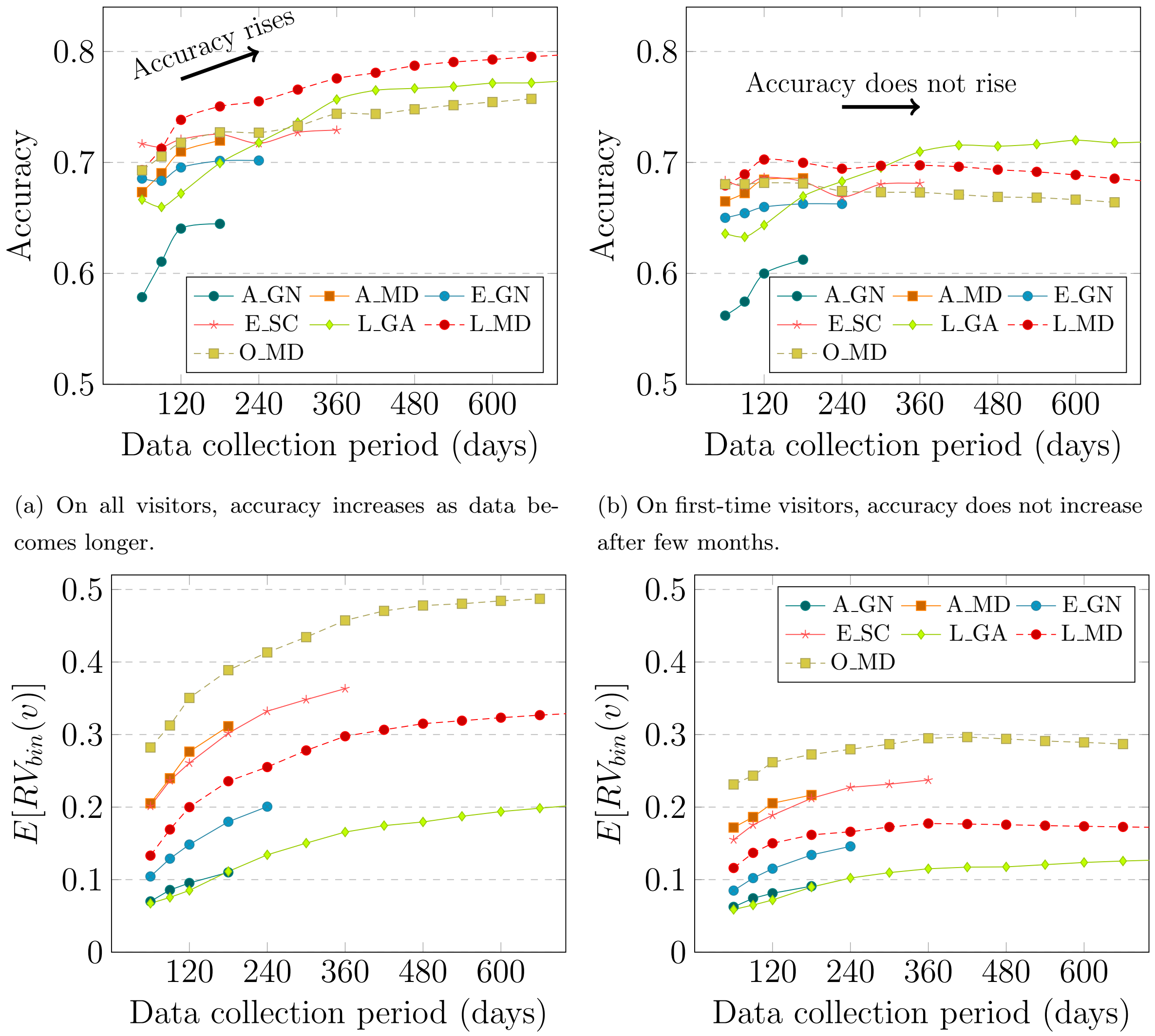
Growing Data—Is It Enough?

Proposing an appropriate data collection period is a very important business decision. From a customer's point of view, you will want to use the customer monitoring service for as short as possible to reduce cost, if there are not any benefit of using the service over certain time horizon. On the other hand, in terms of providing services, you want to make a long-term contract to make a steady profit. We would like to suggest some pointers to solve the following questions.

- In terms of predicting customer revisit, how long should we collect the data?
- Is the collection period required for each store different?

Figure 3.8(a) shows that the overall prediction accuracy increases as the length of the data collection period increases. The performance rapidly increases over the first few months, and then the increment is getting smaller. The main reason for the poor performance in the first few months is the lack of information on revisiting customers. Suppose that we have the data only for the first three months and consider a customer who did not revisit within the three months. His/her revisit intention is labeled as being false. However, what if he/she revisits after four months? Therefore, the labels in the training data

[11] We ran another five sets of 5-fold cross-validation for this experiment. Thus, the values of the baselines in Table 3.8 are slightly different from those in Table 3.7 within the margin of error.



(a) On all visitors, accuracy increases as data becomes longer.

(b) On first-time visitors, accuracy does not increase after few months.

(c) On all visitors, average revisit rates keep increasing for all cases.

(d) On first-time visitors, average revisit rates decrease in some cases.

Figure 3.8: Impact of the data collection period.

could be inaccurate if we collected the information for an *insufficient period*. To confirm our conjecture, we also examined the proportion of customers' revisit intention as the data collection progressed, as in Figure 3.8(c). The proportion, $E[RV_{bin}(v)]$, indeed increased as the data collection period increased. Thus, we confirmed that more customers were turned to repeating visitors with more data. However, prediction accuracy on first-time visitors did not always increase. We notice that the average revisit rate also decreases for those cases, i.e., O_MD and L_MD, which implies that recently visited customers do not tend to revisit the store. Overall, with a longer data collection period, performance improvement occurs by having more positive cases for regular customers. But, we should not make hasty generalizations since we only look at the data from seven stores. Therefore, the period for data collection, especially for predictive analytics, should be carefully determined. In the next chapter, we will share some of our thoughts on determining the data collection period by comparing the power-law distribution coefficient as time progresses.

Deciding Data Collection Period by Power-law Coefficient

Since $E[RV_{bin}(v)]$ and $E[RV_{days}(v)]$ vary from store to store, it may be difficult to determine whether the collected data is sufficient or not. We suggest that the appropriate data collection period can be set using the variation of the power-law coefficient. Similar to the previous study on the number of visits to an internet website [2], the number of visits to a store k is close to a straight line in the log-log plot, and can be approximated by a power-law distribution: $p(k) \sim Cx^{-\alpha}$. By definition of the power-law distribution, a store with a smaller power-law coefficient α has a higher average revisit rate $E[RV_{bin}(v)]$. The power-law coefficient for each store vary widely[12]. However, as the data collection period gets longer, the visit frequency distribution converges, and thus the coefficients decrease and converge. Table 3.10 shows how the power-law coefficient ratio $\alpha_t/\alpha_{t+30days}$ of seven stores converges to 1 over time. Here, we can state that the visit frequency distribution of the 180-day O_MD data is more close to stationary distribution than that of the 240-day L_GA data. By looking at the plots in Figure 3.8, we can confirm that the predictive power and average revisit rates of store O_MD saturates earlier than those of store L_GA. Although we could not succeed in explaining the relationship theoretically, we experimentally confirmed that as the number of power law coefficients rapidly approaches 1, the distribution and predictive power rapidly saturate. So we encourage practitioners to decide an appropriate data collection length by monitoring the convergence rate.

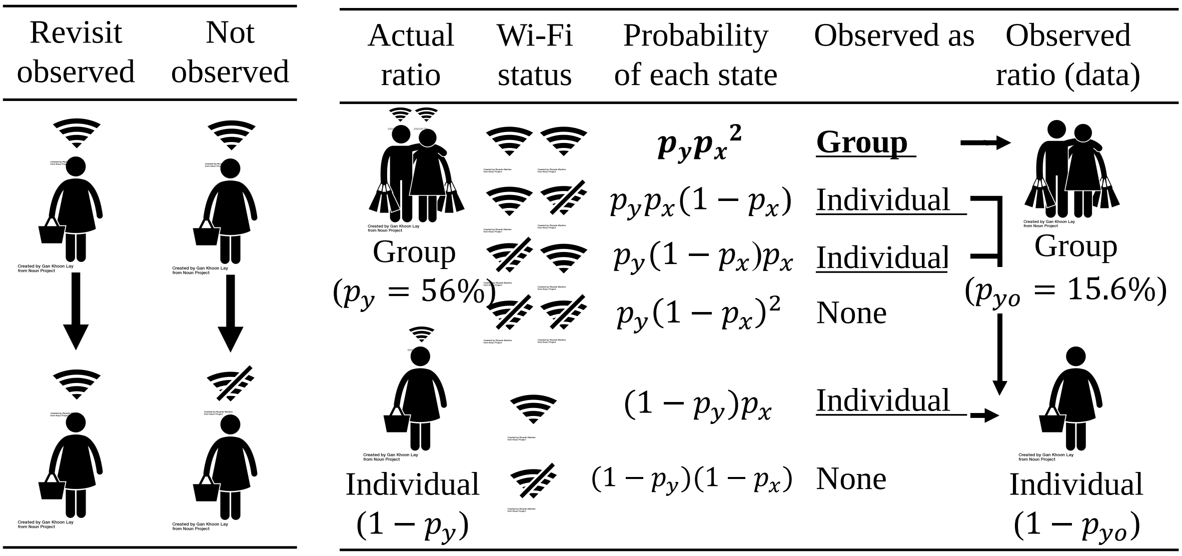
Table 3.10: Data sufficiency depending on the data collection period. The closer the value is to 1, the more stationary the distribution of data is.

Shop ID	t = 30 days	90 days	180 days	240 days	360 days	660 days
A_GN	1.609	1.119	1.070	-	-	-
A_MD	1.365	1.165	1.051	-	-	-
E_GN	1.479	1.162	1.078	1.036	-	-
E_SC	1.482	1.113	1.057	1.035	-	-
L_GA	1.517	1.138	1.115	1.075	1.038	1.013
L_MD	1.661	1.213	1.069	1.060	1.027	1.008
O_MD	1.303	1.133	1.048	1.026	1.022	1.002

Real Behavior and Collected Data—Are They Same?

Although the Wi-Fi positioning system enabled *noninvasive* monitoring, it is also limited, considering that not all users turn on Wi-Fi of their mobile device. Since the 4G LTE connection is very fast and ubiquitous in Korea [84], the proportion of "always-on" users is just 30 % [128]. This limitation implies that the datasets were missing some customer behaviors in the real world. Figure 3.9(a) shows untraceable revisits due to the conditional Wi-Fi usage of the customer, and Figure 3.9(b) shows a gap between actual/observed proportion of group movements caused by low Wi-Fi usage. The reason for the difference is that both companions must use Wi-Fi to verify the accompanying records on the data. p_x denotes the probability of customers who turn on Wi-Fi on-site including 'conditionally-on' users, and p_y denotes the actual proportion of customers in a group of size two. Here we ignore groups more than

[12] The power-law coefficient from each store calculated by [5] is listed in Table 2.1.



(a) Revisit.

(b) Group movement ($p_x = 0.392, p_y = 0.56$).

Figure 3.9: Missing behaviors in noninvasively collected data: (a) Customers' revisits were untraceable if they did not have Wi-Fi turned on. (b) The actual group movement ratio was 56 % instead of 15.6 %. Researchers must not interpret the data as it is, when explaining the real behavior.

two customers, which are not that common. Then the proportion p_{yo} of group customers observed in the data can be represented as Eq. (3.3).

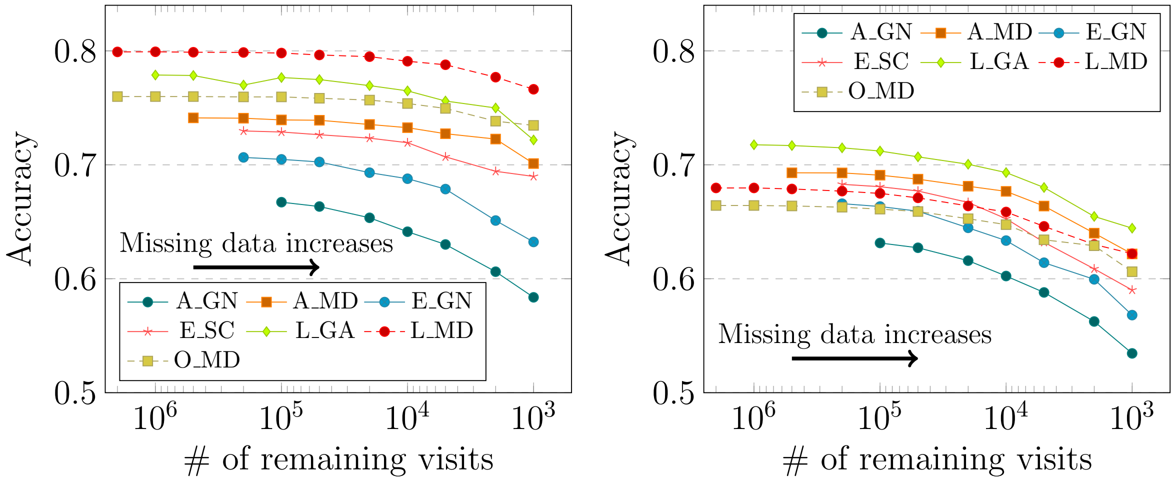
$$\begin{aligned} p_{yo} &= \frac{Observed(Group)}{Observed(Group) + Observed(Individual)} \\ &= \frac{p_y(p_x)^2}{p_y(p_x)^2 + 2p_y p_x(1-p_x) + (1-p_y)p_x} = \frac{p_x p_y}{1 + p_y - (p_x)^2}. \end{aligned} \tag{3.3}$$

Therefore, readers should recognize that the observed movement ratio can be very different from the actual movement ratio. We encourage readers to go back to Section 3.2.2 to check how to utilize this gap to decide the 30 s threshold to determine group movements. In the future, if customers' behaviors are more traceable with additional sensing technologies, we believe that *noninvasively* collected data will better reflect actual customer behaviors.

Assumptions to Interpret the Data

Continuing the context, we would like to clarify how we count the first-time visitors and explain several underlying assumptions to consider.

- Assumption 1: Because we do not know whether customers visited a store before data was collected, we simply assume that the customers did not visit before the collection period. We believe that this assumption is reasonable because the stores in which we collected the data were relatively new at that time we began data collection.
- Assumption 2: Because customers are captured only when they turn on the Wi-Fi of their mobile device, we assume that the customers' Wi-Fi turn on behavior is consistent when they visit the store. Also, we assume that there is no correlation between Wi-Fi usage and customer groups (first-time visitors and VIP customers).



(a) On all visitors.

(b) On first-time visitors.

Figure 3.10: Model robustness on missing customers.

- Assumption 3: We assume that customers visit the store with a device having the same MAC address. This assumption holds if there are not many people carries several Wi-Fi devices during their shopping. Also, we removed devices that follows MAC-address randomization.

Rigorously speaking, the proportion of true first-time visitors would be less than 70 % by considering all the effects explained above. Nevertheless, these customers are also likely to be early stage visitors.

Performance on Incomplete Data

Assuming that some of the customers' data are completely gone, is the performance of our model reliable? We confirmed that over 95 % of the performance of our model is maintained with a very small fraction of the dataset (e.g., 0.5 % for L_MD). For each store, we randomly removed the records of a set of customers and measured the model performance using the remaining data. Figure 3.10 shows the averages for 20 different executions. The accuracy loss of the model was within 3 % if 10,000 visits were secured. This observation can be also interpreted as follows:

- For large-scale mobility data, a comparable prediction model can be built by using small data subsets.
- On the other hand, we can estimate the prediction performance when all customer data becomes traceable.
- High prediction accuracy of some stores may not be due to their large number of visitors.
- Our revisit prediction framework can be also effective to smaller stores.

Meaningful Insights but Low Predictability

We would like to point out that securing prediction accuracy can be difficult although the differences between the two groups may seem obvious. The values of handcrafted features significantly differ by the revisit status, each of which is helpful to explain customers' visit patterns in retail businesses. But from the perspective of a prediction task, the correlation coefficient between the feature and the revisit label was relatively small, and the prediction accuracy using the feature was not very high.

In Table 3.11, the relative difference diff_1 in the feature values depending on the future revisit status is noticeable (2.7–104.2 %). Besides, the p-value $(p < 10^{-100})$ from Mann-Whitney U test supports that samples selected from revisited group $\{V_1 \mid v \in V_1,\ RV_{bin}(v) = 1\}$ and non-revisited group $\{V_0 \mid v \in V_0,\ RV_{bin}(v) = 0\}$ are not from the same distribution. The relative difference diff_2 of the revisit rate between the top 5 % and the bottom 5 % of customers in terms of feature values also shows clear distinction by 43.5–134.7 %.

However, the correlation coefficient and the final prediction accuracy using the feature are not as impressive as diff_1 and diff_2. Practitioners should note that the behavioral difference between the two groups is obvious and the p-value is high, but not in terms of the metric of correlation and prediction accuracy. Also, the feature should not be discarded because of the low correlation coefficient. If the feature has a nonlinear tendency, its predictive power can be strong. The statistics of f_b and f_c in Table 3.11 confirms our argument. We assert that our high-quality prediction came from a combination of various kinds of features which behave differently.

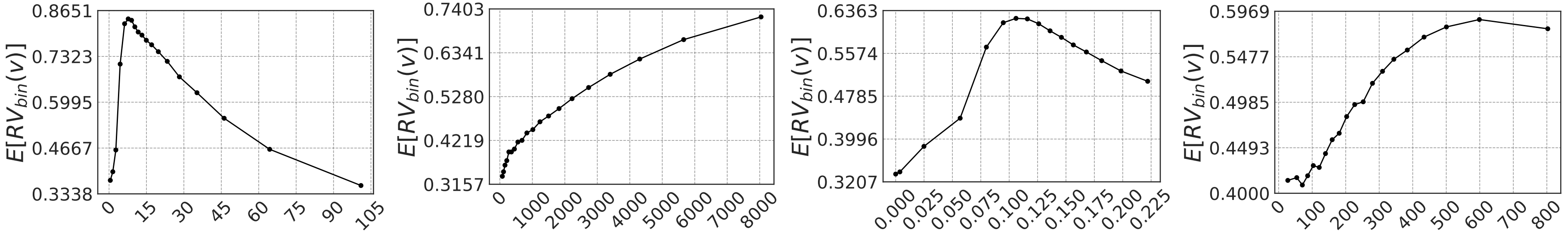
3.4 Summary

Various retail analytics companies have set up sensors to monitor customer mobility in offline stores. Although it was difficult to connect with other kinds of data because of legal issues, we confirmed that customer mobility indeed involves diverse meanings. Without having access to customer purchase data or customer profile, we have found that revisit intention of customers are predictable by up to 80 %, using only Wi-Fi signals collected by in-store sensors. Toward this goal, we suggested guidelines for setting the collection period of indoor data for revisit prediction. We also showed our model is robust even if a majority of customer data is missing. Moreover, we demonstrated that significant observations may be in disagreement with the final predictive power. The proposed set of features has enough generality to be applied in any offline stores tracking customer foot prints. We expect that our findings will help data scientists and marketers from both retail analytics companies and their clients make important decisions.

Although we did not point it out, several interesting questions are remained. What if we train our current model on the original datasets with huge class imbalance? How to use partial observations captured at the end of the data collection period? Is our model still good enough if we divide training and testing sets by time after preserving class imbalanced? Although we performed quite exhaustive experimental evaluation and the results were aligned with the initial claims, we cannot affirm that we answered those questions in this chapter thoroughly. In the next chapter, we introduce our new model *SurvRev* to answer those questions.

Table 3.11: Statistics of feature values with revisit status, and their final predictability: statistics from the store O_MD.
($FV_1 = E[FV(v)|RV_{bin}(v) = 1]$, $FV_0 = E[FV(v)|RV_{bin}(v) = 0]$, r_{pb}: Point-biserial correlation)

	Feature value difference by revisit status				Revisit rate difference by feature value interval (Figure 3.12)				
Feature Name	FV_1	FV_0	diff$_1$	p-value	$max(E[RV_{bin}(v)])$	$min(E[RV_{bin}(v)])$	diff$_2$	r_{pb}	Accuracy
f_a: Avg interarrival time (5 m)	21.752 days	44.410 days	104.2 %	0****	0.8408	0.3582	134.7 %	-0.207	0.7346
f_b: Total dwell time	3211 s	1612 s	99.2 %	0****	0.7206	0.3348	115.2 %	0.216	0.6005
f_c: 3rd longest area dwell time ratio	0.112	0.087	28.5 %	0****	0.6223	0.3350	85.8 %	0.152	0.6035
f_d: Avg dwell time for each area	357.53 s	348.16 s	2.7 %	0****	0.5880	0.4098	43.5 %	0.007	0.5584



(a) f_a

(b) f_b

(c) f_c

(d) f_d

Table 3.12: Detailed relationship between four features and $E[RV_{bin}(v)]$ mentioned in Table 3.11.

Chapter 4. Revisit Prediction By Deep Learning

Chapter based on the recent work after the proposal.

In this chapter, I introduce *SurvRev*, a next-generation revisit prediction model that can be tested directly in the business. Through an appropriate combination of survival analysis and deep learning with our domain knowledge, the new *SurvRev* model covers the predictive analytics on imbalanced class with partial observations.

Our *SurvRev* model has many advantages. First, *SurvRev* can use *partial observations* which were considered as missing data and removed in the previous regression framework. By using deep survival analysis, we are able to estimate the next customer arrival from unknown distribution. Accordingly, *SurvRev* is robust on huge class imbalance occurred by the censoring effect. Second, *SurvRev* is an event rate prediction model. It generates the predicted event rate of the next k days rather than predicting revisit interval $RV_{days}(v)$ and revisit intention $RV_{bin}(v)$ directly. This design enables *SurvRev* to work well on testing sets with unknown probability distributions.

We showed the superiority of the *SurvRev* model by comparing with diverse baselines including our feature engineering model and the state-of-the-art deep survival models. we reconfirm that in-store signals captured by customer mobility can be an important clue for predicting their future behavior.

For fertilizing this field, we also released more realistic benchmark dataset for revisit prediction, which will be the first publicly available dataset as far as I know. We believe this dataset can be used for diverse topics—predicting stickiness[1] of the customer, next area prediction, or funnel analysis to increase an inflow rate.

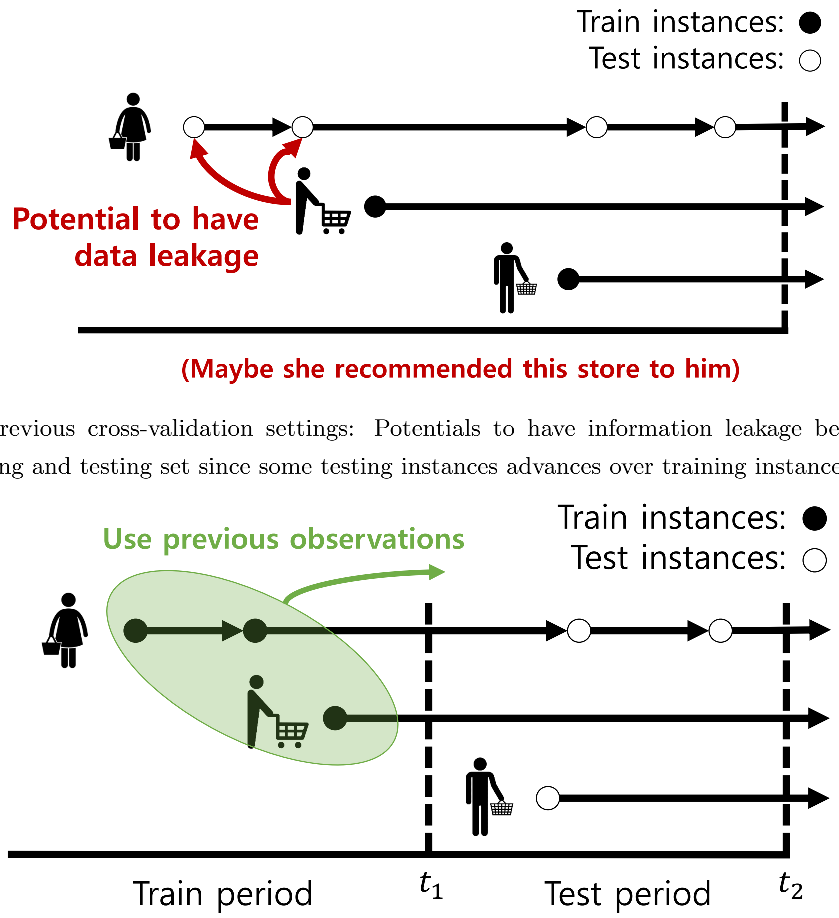
4.1 Motivation

> *Deep learning is when students create connections between the course material and their own lives.*
>
> — James Lang

In the previous chapter, we introduce our revisit prediction framework powered by feature engineering. We show the effectiveness of our framework by exhaustive experiments. However, we received some concerns about our evaluation protocol. We would like to take a look at the opinions one by one and introduce our effort to set up more principled evaluations to represent the real-world application.

Towards Practical Application Settings

The first concern is the evaluation method using cross-validation. We removed test users entirely and train with the remainder users and test on the removed users. This policy perfectly makes sense for prediction tasks in a static dataset unrelated to time. However, in a longitudinal prediction setup, this policy leads to an implicit data leakage because the testing dataset is not guaranteed to be collected



(a) Previous cross-validation settings: Potentials to have information leakage between training and testing set since some testing instances advances over training instances.

(b) New prediction settings: Data is splitted by time so there is no potential leakage between training and testing set. Previous observations are used to predict the future.

Figure 4.1: Updated data splitting rules.

later than the training set. We also agreed to split the data according to a particular date and train on the former part and test on the latter part (Figure 4.1).

The second concern is factitious downsampling. Although our feature set led to significant performance improvement on a downsampled dataset, we cannot guarantee to observe the same amount of improvement if our framework is evaluated on the original imbalanced dataset. In the case of extreme class imbalance, the predictive power of each feature might disappear due to the dominance of the majority label. We accepted the comments and decided to use the original dataset without any adjustment.

There are several advantages by using the original dataset as well as applying time-based data splitting. First, we can fully use all instances without any information loss as compared to using downsampled datasets. Second, since the revisitation appears to have a huge imbalance, we can apply our method to the actual system without making any change. Third, this is equivalent to real-world prediction scenario where the model is asked to predict the user's next decision when the user's behavior up to the current time exists.

However, we also met some new challenges that were not considered before. After splitting the dataset by time, we had to find a way to deal with *partial observations*, which are also called as *train-*

[1] Stickiness is a marketing term to describe the average time per month at a site.

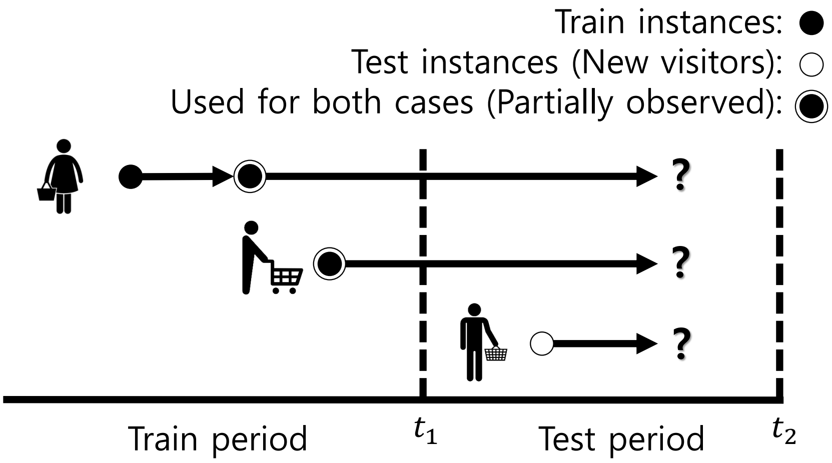
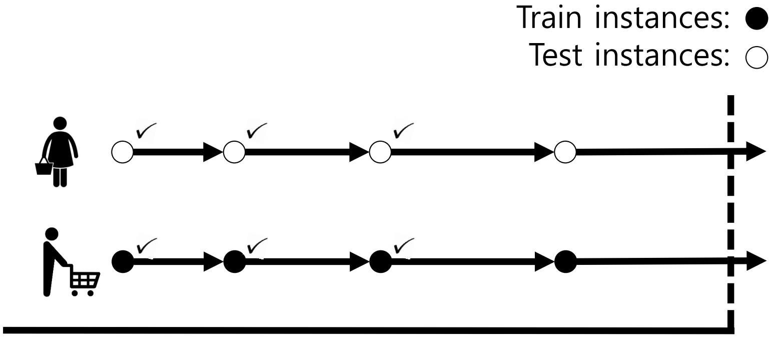

Train instances: ●
Test instances (New visitors): ○
Used for both cases (Partially observed): ◉
?
?
?
Train period
t_1
Test period
t_2

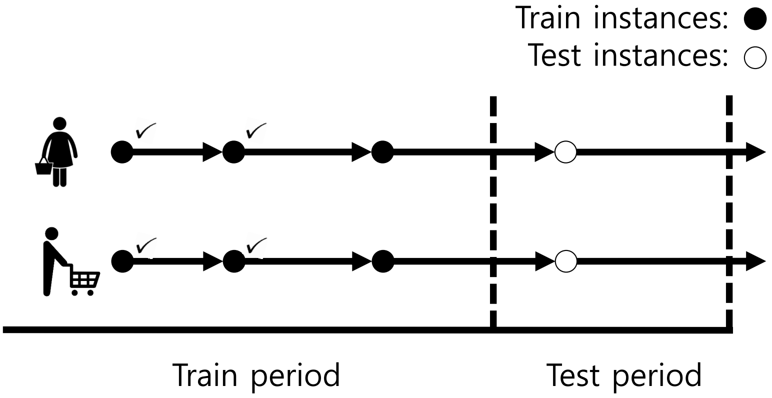
Figure 4.2: An example of categorizing training and testing cases in our revisit prediction task. Train-censored cases are marked with two circles and these instances are used for both training and testing.

censored visits. Train-censored visits are partially observed during the training period that require prediction for the remaining test period. In other words, train-censored visits are *the last visit* of customers collected during the training period. In most offline stores, the amount of censored visits is very large compared to revisited cases. However, the regression model in Chapter 3 cannot use any information from censored visits since we do not know when will they come back. This causes not only a huge information loss but also resulting in a biased prediction since a model is trained with revisited cases only. At the same time, train-censored visits are the subject of prediction. For companies, it is necessary to predict the possibility of revisit for train-censored visits since they would like to know their customers are completely churned or not [43]. Those customers already experienced the store, increasing their retention leads to a long-term benefit. In Figure 4.2, we illustrate an example of categorizing training and testing cases in our revisit prediction task. As illustrated, train-censored instances can be also used to train the model as well as they are used for testing. Since we split the data by time, partial information up to time t_1 can only be used for model training.

In addition to that, the good prediction model should predict the revisit of new visitors appeared during the testing period. In this thesis, we simply call the visits from the new visitors as testing instances. Predicting revisit of censored customers and new visitors together is very challenging, since the characteristics of those two groups are inherently different such as the remaining observation time and their visit histories. A big challenge comes from the difference of class distribution between the training, train-censored and testing instances. Figure 4.3 illustrates this phenomenon. A usual data mining classification task assumes that the class distributions between training and testing set are the same. However, it does not hold in our new setup and it gets worse by changing training length. Fard et al [22] also emphasized that predicting train-censored instances in longitudinal data is difficult owing to the event rate decaying with the observation time. They call this prediction setting as *early-stage event prediction* and make a distinction of it with a event prediction where training and testing data is collected during the same time period or same observation length. For example, if we assume the customer arrival process follows the *Poisson Process* [97], the expected revisit interval of train-censored instances is always larger than the expected revisit interval of testing instances appeared in the test period owing to the *memoryless property.*



(a) User based splitting: Although this was not the best prediction setup, the revisit interval distribution and the revisit ratio between training and testing sets were similar by the *law of large numbers*. Therefore, the classification model learned from the training set is effective to predicting testing instances. In this example, $E[RV_{bin}(v) \mid v \in V_{train})]$ and $E[RV_{bin}(v) \mid v \in V_{test}]$ are both $\frac{3}{4}$.



(b) Time-based splitting: In the new prediction settings, the revisit interval distribution and the revisit class distribution between training and testing sets are inherently different due to censoring effects. The classification model learned from training set might have difficulty to predicting testing instances. In this example, $E[RV_{bin}(v) \mid v \in V_{train}] = \frac{2}{3}$, whereas $E[RV_{bin}(v) \mid v \in V_{test}] = 0$.

Figure 4.3: Additional class imbalance occurred by new data splitting scheme.

In summary, we focus on the following challenges to make our prediction framework successful in real application settings:

- *Applied time-based splitting* instead of 5-fold CV based on users.
- *Used original imbalanced dataset* instead of downsampling it.
- *Considered both censored customers and new visitors* for testing our framework.

We believe these principles are crucial in applied data science research and a big advantages over the previous works which compromise difficulties. The *SurvRev* model is a new solution to handle these challenges. Before presenting our *SurvRev* model, we present some backgrounds on survival analysis.

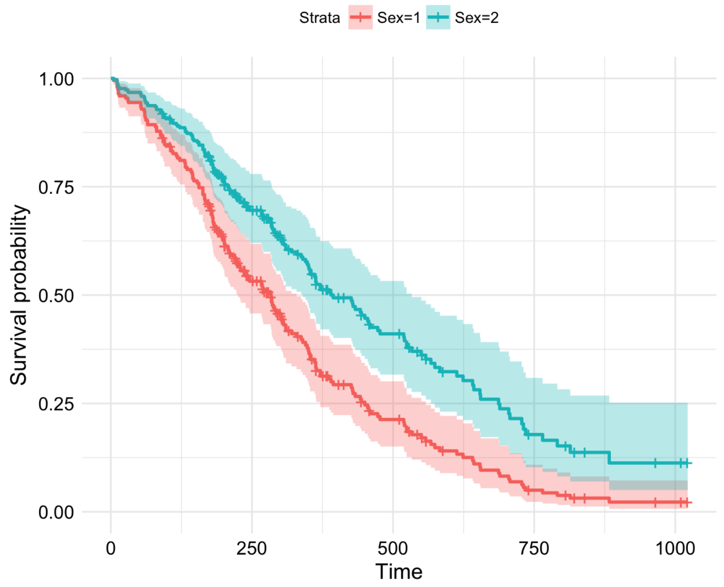


Figure 4.4: An example of survival analysis for life-time prediction[2].

4.2 Background on Survival Analysis

Survival Analysis [111] is a branch of statistics for analyzing the expected duration of time until certain events happen. It originates from the medical domain where new patient's expected time to events should be predicted from previous observations. Using survival analysis, one can ask the following questions: What is a ratio of survival after 5 years after being diagnosed as colon cancer? Of those who survive, what rates will they die? Researchers expanded the concept of an event and adopted the theory to diverse problems not only in the medical domain, but it is also called as reliability theory in engineering domain or duration analysis in economics. More generally, it involves the modeling of time to event data. Recent work include web browsing [12], churn analysis [43], and bidding prediction [95].

One of the biggest benefit by adopting survival analysis is to use *partial observations*, which are common in longitudinal studies. In our problem setting, not every customer revisits during the observation time. For some stores, the revisit ratio is below 50 % even if we collected signals for more than 2 years, which means that more than 50 % of the visits are considered as missing data in general regression settings. Surely one would not want to exclude all of those visits from the study by declaring them as missing data, while training a model to predict a revisit interval of new visitors. In our prediction model, we would like to interpret those footprints as "The customer visited our store twice during the summer, but he did not come again before the year ends" and get benefit from them by using the partial information. The presence of incomplete observations is called *censored observations* [32], which brings a unique challenge in survival analysis and differentiates survival analysis techniques from other standard regression methods [65]. In the following paragraph, we briefly introduce the basic concepts and notations in survival analysis.

Notation

In survival analysis, we divide data into two categories: *censored* and *uncensored*. *Censored data* comes out when an event has not occurred during the observation period. In our problem settings, the

[2] Image courtesy of `http://www.sthda.com/english/wiki/cox-proportional-hazards-model`.

Table 4.1: Summary of three types of statistical methods in survival analysis.

Type	Advantages	Disadvantages	Methods
Non-parametric	Efficient when no suitable theoretical distributions are known.	Difficult to interpret, Utilize labels only and ignore attributes.	Kaplan-Meier [47], Nelson-Aalen [83, 1], Life-Table [18].
Semi-parametric	The knowledge of the underlying distribution is not required.	Distribution is still unknown, not easy to interpret.	Cox model [17], Regularized Cox [107], Time-Dependent Cox [24].
Parametric	Easy to interpret, efficient when the survival times follows a particular distribution.	Fragile when the distribution assumption is violated.	Tobit [108], AFT [112], Buckley-James [11], Penalized Regression [127].

last visits of all customers consist of a set of censored visits. Since the data collection period is finite, we do not know the exact revisit interval for those censored visits. *Uncensored data* is a data which an event has occurred during the observation period. In our settings, all preceding visits before the last visit comprise a set of uncensored visits.

The *survival function* is conventionally denoted as $S(t)$, which is defined as $S(t) = \Pr(T > t)$. That is, the survival function means a probability that the event time T is later than the certain time t. Naturally, the function is decreasing and $S(0) = 1$. The *lifetime distribution function* $F(t)$ is a complement of the survival function, which is defined as $F(t) = \Pr(T \leq t) = 1 - S(t)$. Since it is a cumulative distribution function of time-to-event, we can also find the *event density function* $f(t)$ by differentiating F if it is differentiable, $f(t) = F'(t) = \frac{d}{dt}F(t)$, that is, a rate of events per unit time. The *hazard function* $\lambda(t)$ is one of the most important functions in survival analysis, which is defined as the event rate at time t conditioned that the item has been survived until time t.

$$\begin{aligned}\lambda(t) &= \lim_{dt\to 0} \frac{\Pr(t \leq T < t + dt | T > t)}{dt} \\ &= \lim_{dt\to 0} \frac{\Pr(t \leq T < t + dt)}{dt \cdot S(t)} \\ &= \frac{f(t)}{S(t)}.\end{aligned}$$

Consider the definition of $f(t)$, which can also be expressed as $f(t) = -\frac{d}{dt}S(t)$, the hazard function can be represented as:

$$\lambda(t) = \frac{f(t)}{S(t)} = -\frac{S'(t)}{S(t)} = -\ln S(t).$$

By integration, the survival function $S(t)$ can be written as $S(t) = \exp(-\int_0^t \lambda(u)du)$.

Traditional Statistical Methods

Traditionally, hazard function in survival analysis has been estimated by *three* types of statistical methods [60]: *non-parametric*, *semi-parametric*, and *parametric* methods. We borrow Table 4.1 from a comprehensive survey paper [111] to summarize the statistical methods in survival analysis.

Non-parametric methods are more efficient when there is no underlying distribution for the event time or the proportional hazard assumption does not hold. Kaplan-Meier [47], Nelson-Aalen [83, 1], and Life-table [18] methods are non-parametric approaches that have been widely used. Using these methods, one can easily get the empirical estimate of the survival function. By using only event time information, these methods are fast enough to catch overall distribution without setting any parameters. However, these methods ignore covariates of each instance, therefore not suitable for revisit prediction.

Semi-parametric methods are hybrid approaches that can obtain more precise estimators than the non-parametric methods, or obtain broader estimators than the parametric methods. In semi-parametric models, no underlying distribution is required, but the attributes assume to have exponential relations for outcome variable. Diverse variation of Cox model [17] has been proposed, such as adding L1 or L2 regularization term (Regularized Cox) [107], considering time-dependent covariates (Time-dependent Cox) [24], and considering mandatory covariates in a boosting scheme (CoxBoost) [8].

Parametric methods are efficient and effective when a time-to-event process follows a probability distribution. By specifying a parametric form of $S(t)$, we can get the benefit of concise equations already derived for each distribution. By specifying a parametric form, parametric methods can easily compute the expected failure time, also compute selected quantiles, and estimate survival functions more correctly than other methods assuming the parametric form is right [79]. Some popular distributions for estimating survival curves are Weibull, exponential, extreme value, log-normal, and log-logistic distribution [3]. Tobit [108], Accelerated Failure Time (AFT) [112], Buckley-James [11], and penalized regression models [127] are popular parametric methods for survival analysis.

Machine Learning on Survival Analysis

To understand behaviors from large-scale and high dimension data with complex survival function, *machine learning* and *deep learning* based survival analysis methods have been proposed with increasing computing power. While taking advantage of the concept of survival analysis, machine learning, and deep learning, approaches effectively handle censored datasets in an advanced way. Some algorithms also empower traditional statistical models by relaxing distributional assumptions and optimization constraints. Popular machine learning methods such as Bayesian methods [92, 22], SVM [50], boosting [36], neural networks [10], survival trees [9], and survival forests [41, 42] have been tailored to handle censored data. Among those approaches, we will summarize several notable work proposed in the past few years.

Recent Trends on Survival Analysis

In this part, we would like to explore the latest trends in survival analysis by their model and domain, then we introduce some work that focus on slightly different viewpoints.

By employing the elastic net as the regularization term, Yan et al [65] proposed regularized parametric censored regression for high-dimensional data. Gaussian processes are also successfully applied to survival analysis. Tamara et al [23] applied Gaussian processes to model non-parametric variations away from parametric baseline hazard, thus successfully handled left, right and interval censored cases.

Patients with reduced immunity are more susceptible to various diseases. In this case, a model should be designed for jointly assessing a patient's risk of multiple adverse outcomes. We call this setting as *competing risks*. Deep multi-task Gaussian process [4] with non-parametric Bayesian model showed its effectiveness over classical models. Lee et al [59] applied n separated fully connected neural networks with residual connections for segregating the effects by different sources of diseases. Like above-mentioned papers, deep survival analysis is applied for diverse problems in the medical domain. Some use cases are for analyzing kidney graft [77], for predicting clinical outcomes from cancer genomic profile [123], for personalized predictions [121], and for personalized treatment recommender system [48]. Usually, in this medical setting, we assume that event happens once during the lifetime so the predictive analytics model focuses on learning from other instances, rather than focuses on personalized training.

Survival analysis can also be used for time-series data where *multiple events* occur consecutively. Variation of *Recurrent Neural Networks (RNN)* [58] such as LSTM [35] is used as a building block for survival analysis with time-series data. Using LSTM and custom 3-way factor layer with multiple outputs, Jing et al [43] studied churn prediction and next action recommendation at the same time by multi-task learning. The idea of quantizing hazard rate is widely used for subsequent papers [59, 95]. In this scheme of music streaming services, each session is recorded in each LSTM cell and the output of each cell turns out to be a set of quantized hazard rates. Using this set of rates, the model minimizes the negative log-likelihood loss for representing gaps between events correctly. Zhou et al [126] used a customized fast-slow recurrent network to differentiate a session sequence and an item sequence in online shopping, and this method shows its effectiveness in click and dwell time prediction. Check-in time prediction in location-based social networks also gets benefit from recurrent-censored regression (RCR) model [120]. Their RCR model used a recurrent neural network to learn latent representations from historical check-ins of both actual and potential visitors, and use those outputs to censored regression for making a prediction. The research team tweaks their approach to recurrent spatiotemporal point process to take advantage of precedent location information on user trajectories [119]. The RNN model is also successfully adopted for personalized survival analysis in clinic datasets when the patient feature has been tracked over time [28]. Even though the feature values are measured once, Ren et al [95] successfully used LSTM in the sense that the hazard rate can be learned autoregressively through time. Although each session is not clearly separable in the above two settings, they were able to take advantage of RNN while modeling the temporal relevance wisely.

On the other side, some studies focus on the subtle difference to survival analysis and emphasize on their new problem settings. Criteo, a global display advertising company, released their solution for predicting conversion rates on their service [13, 12]. They focused that some visitors in their online shopping mall may not eventually convert to a potential customer, unlike survival time analysis where a patient will eventually die. From this intuition, they suggested a concept of *delayed feedback* to describe a time between ad-clicking and conversion. They combined two models for capturing conversion itself and for capturing delay before conversion. Fard et al [22] emphasize the difference between *early stage prediction* with survival analysis and time series forecasting. They focus on predicting an event occurrence at a future time point using only the information collected before certain observation time. By agreeing to this idea, we also divided the testing set into *train-censored* case and *first-time visitor* case. More recently, Zhang et al [125] train a neural network for remaining useful life prediction, similar to reliability theory, by providing a new censoring Kullback-Leibler divergence for evaluating the dissimilarity between the binary classification probabilities and the actual survival process. Researchers are also interested in studying evaluation metrics in survival analysis and perform an in-depth analysis of the well-known evaluation metric concordance index (c-index) [93], which is a metric for measuring pairwise ranking accuracy.

Lastly, I would like to introduce publicly available packages in survival analysis. Python lifeline package[3] and R survival package[4] contain the core survival analysis routines for implementation, and they can be used for understanding code bases and for baseline implementation. I hope this summary helps readers who are interested in predictive analytics with censored data.

[3] `https://github.com/CamDavidsonPilon/lifelines`

[4] `https://cran.r-project.org/web/packages/survival/survival.pdf`

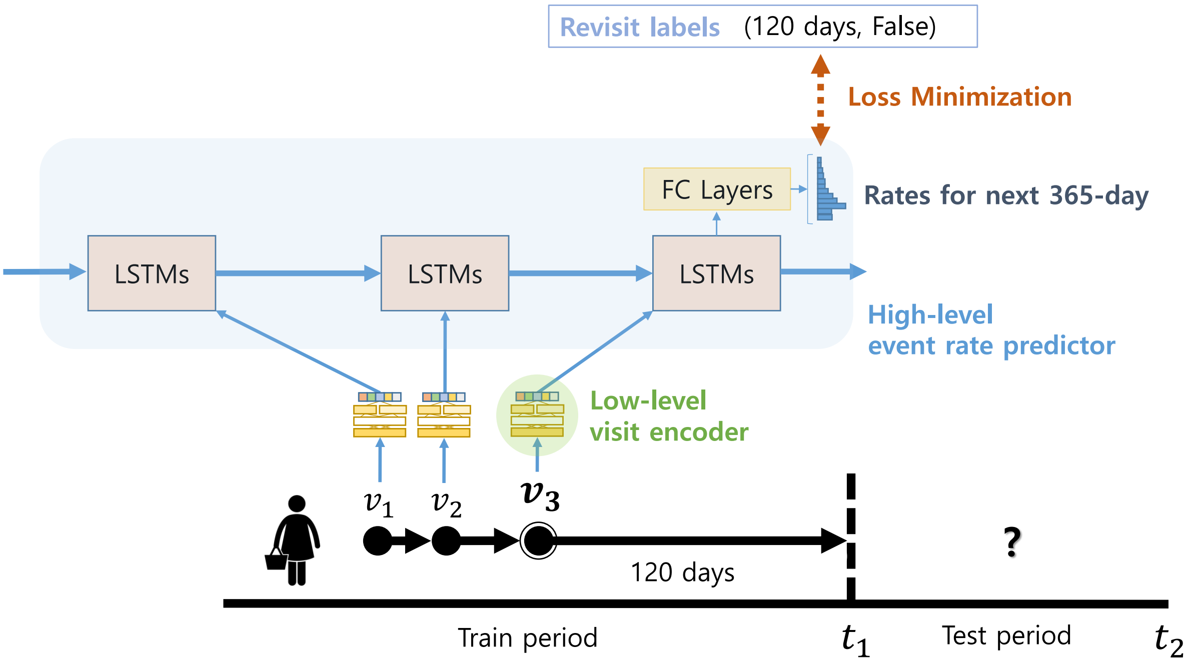


Figure 4.5: The architecture of our *SurvRev* model. Here is a training case for an instance v_3. The current visit data and its histories are passed through low-level encoders. Learned representations pass through a high-level event rate predictor consisting of LSTMs and fully connected layers. The output is the event rates for the next k days. After passing through several conversion steps, it minimizes the model loss. In this example, v_3 is a censored case that has not been revisited for 120 days. Therefore, the output event rates of v_3 that passed through the model are optimized to reflect this information. It is a big improvement since we ignored this censored case for training the regression model in the previous chapter.

4.3 Key Contribution: Deep Survival Model (*SurvRev*)

In this section, we introduce our approach to predict customer revisit. We named our model as *SurvRev*, where the meaning of an acronym is a ***S**urvival **Rev**isit* prediction model.

4.3.1 Overall Architecture

Figure 4.5 illustrates the overall architecture of our *SurvRev* model. The *SurvRev* model is designed as the combination of two modules: a *low-level visit encoder* (§ 4.3.2) and a *high-level event rate predictor* (§ 4.3.3). A low-level visit encoder is to learn hidden representation from each visit and a high-level event rate predictor is to estimate the event rates for the future by considering past information altogether. The final output of the high-level module is a set of predicted revisit rate for next k days. In order to calculate the loss function, we do some calculations for converting event rates (§ 4.3.4) to revisit probability at time t and the expected revisit interval. The whole model is trained by four different types of loss functions (§ 4.3.5), which are designed to optimize prediction results in various metrics.

4.3.2 Low-level Visit Encoder

Figure 4.6 illustrates the architecture of the low-level visit encoder. In the low-level visit encoder, the main area sequence inputs go through three consecutive layers and combined with auxiliary visit-level

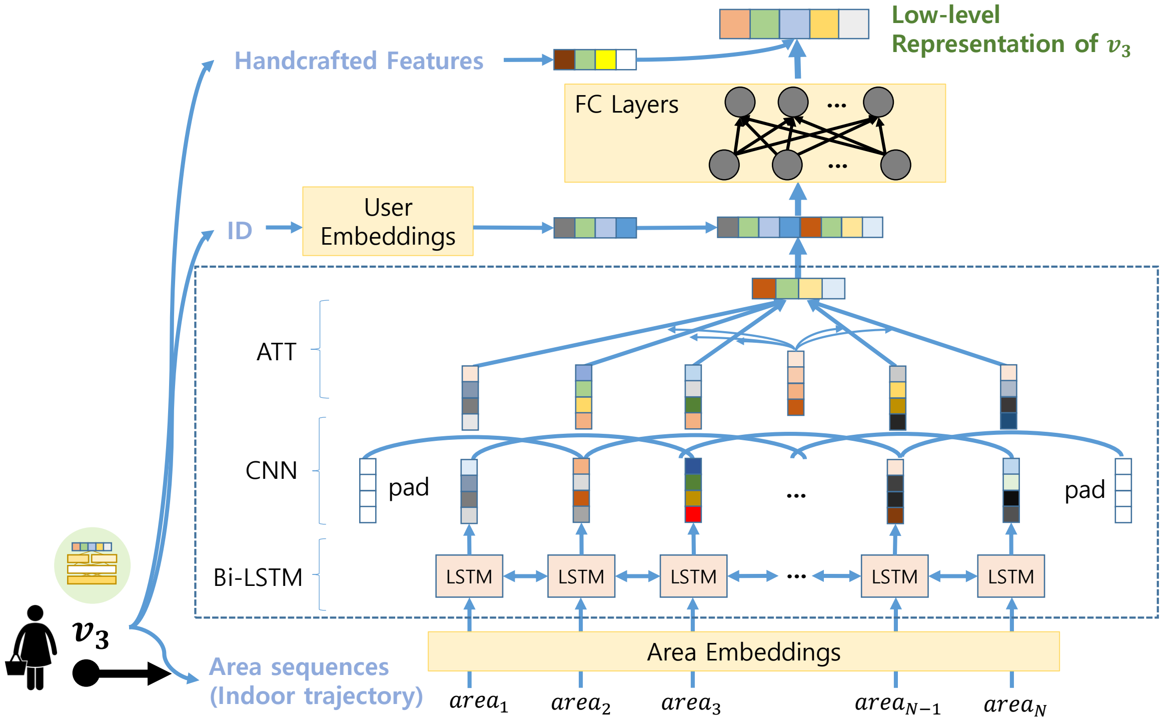


Figure 4.6: The low-level visit encoder of *SurvRev* model.

inputs—user embeddings and handcrafted features. We first introduce three-tiered main layers for area inputs, then introduce the process line of auxiliary visit-level inputs.

Processing Area Sequences

The first layer which an area sequence passes is a *pretrained area embedding* layer to get the dense representation for each sensor ID. We prepared the area pretrained embeddings and the user pretrained embeddings by using Doc2Vec [57] algorithm, implemented[5] in Gensim library [94]. After applying embeddings to each element of an area sequence, we concatenated it with the dwell time of each area. Then it goes through a *bidirectional LSTM (Bi-LSTM) [101]* to find relations back and forth. We expect the *Bi-LSTM* to learn meaningful sequential patterns that determine customer revisit. Each LSTM cell emits its learned representation, and the result sequences pass through a one-dimensional *Convolutional Neural Networks (CNN) [55]* to learn higher-level interaction. We expect CNN layers to learn higher-level representations from wider semantics, which are previously designed from the concept of multilevel location semantics such as category-level or gender-level. In business, the number of CNN layers can be determined depending on how many meaningful semantic levels that the store manager wants to observe. The output of the CNN layer goes through the *attention network [6]* to look over all the information that each visit contains. We expect the *attention layer* to highlight the specific part of the motion pattern which determines customer revisit. Through this sequence of processes, *SurvRev* can learn the diverse levels of hidden representations from the area sequences of each visit.

[5] *Doc2Vec* was popularized by *Gensim* (`https://radimrehurek.com/gensim`), a widely-used implementation of paragraph vectors.

Adding Visit-level Features

From here, we concatenate a user representation with a area sequence representation, then applied *fully connected layers* (FC) with ReLU [29] activation. We can implicitly control the importance of two representations by changing dimensions for both inputs. ReLU activation is done to align values into positives before combining with handcrafted features, which are positives too. Finally, we concatenate selected *handcrafted features* with the combination of user and area representations. The handcrafted features contain crucial information for summarizing visits and revisit prediction that cannot be directly captured by the boxed component in Figure 4.6. The selected handcrafted features are listed as follows.

- *Total dwell time*: How long does the customer stay during this visit?
- *Average dwell time*: How long does the customer stay on average in one area?
- *Number of areas visited*: How many times has the customer moved between different areas?
- *Number of unique areas visited*: How many unique areas captured during this visit?
- *Visit day*, *hour*, and its *combination*: What day of the week (Mon–Sun) and when did the customer visit (0–23 o'clock)? The combined feature is a 168-dim made from (day, hour) tuple, since the visit time may have different meanings even though the customer visit at the same hour or the same day. We used the numeric value instead of one-hot encoding.
- *Number of visits*: How many times have the customer visited the store?
- *Previous interval*: How long has it been since the last visit? If the visit is collected from the first-time visitor (left-censored), the interval between the first observation to the event occurrence point is used.

We applied batch normalization [40] before passing a final result through the high-level module of *SurvRev*. It is used to normalize the input layer by adjusting and scaling the activations, results to improve the speed, performance, and stability of artificial neural networks.

4.3.3 High-level Event Rate Predictor

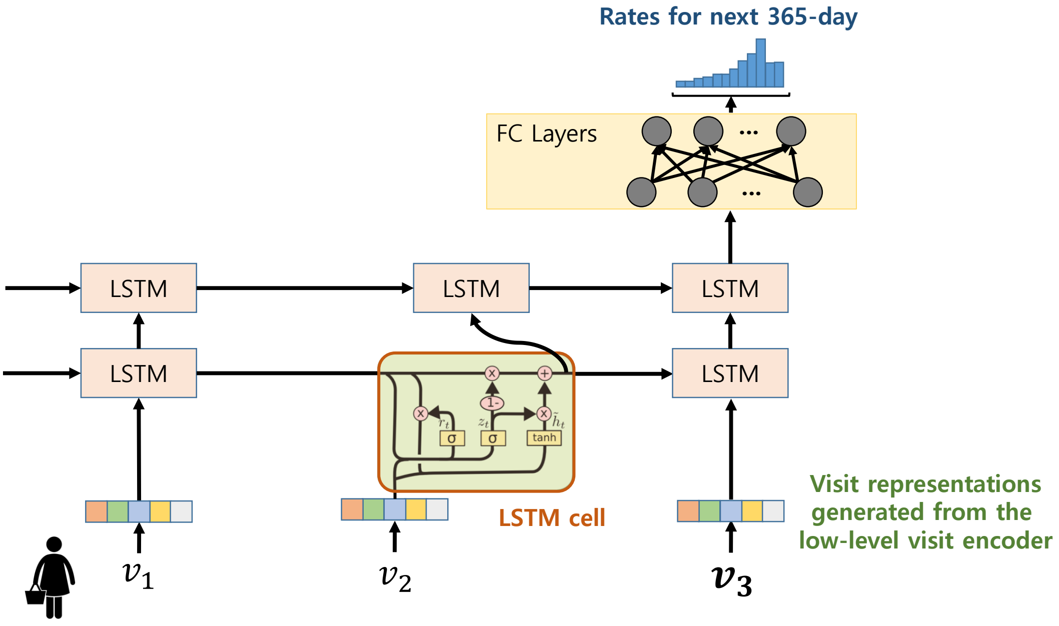


Figure 4.7: The high-level event rate predictor of *SurvRev* model.

Figure 4.7 illustrates the architecture of the *high-level event rate predictor*. The main functionality of the high-level event rate predictor is to consider the customer's *previous histories* by using dynamic LSTMs [35] and predict the revisit rate for next k days.

Assuming customers with multiple previous visits, we will describe how the data flow. For each customer, a sequence of outputs from low-level encoder becomes the input to the LSTM layers. We use dynamic LSTMs to allow for variable sequence lengths, which has a parameter to control the maximum number of events to consider. The sequential output from each LSTM cell goes through our final fully connected layer with softmax activation. The dimension of the final FC layer is k, which is a tuneable parameter. We set it as 365 in order to represent *quantized revisit rates* [43] for the next 365 days. Another reason to set k as 365 is that the total length of our dataset is 365 days. For convenience, we call this 365–dim revisit rate vector as $\hat{\boldsymbol{\lambda}} = [\hat{\lambda_t}, 0 \leq t < k, t \in \mathbb{N}]$. Each element $\hat{\lambda_t}$ indicates a quantized revisit rate in a unit time bin $[t, t+1)$.

4.3.4 Output Conversion

In this section, we explain how to convert 365-dim revisit rate $\hat{\boldsymbol{\lambda}}$ to other criteria such as *probability density function*, *expected value*, and *complementary cumulative distribution function* (CCDF). These criteria will be used for calculating diverse loss function in § 4.3.5. Remind that $RV_{days}(v)$ denotes a next revisit interval of visit v. This means that a revisit occurs after $RV_{days}(v)$ from the time of customer make a visit v to the store.

1. Substituting a quantized event rate $\hat{\boldsymbol{\lambda}}$ from 1 results in a survival rate. Survival rate $1 - \hat{\boldsymbol{\lambda}}$ denotes a rate at which revisit will not occur during the next unit time conditioned that the revisit has not happened so far. Therefore, cumulative product of the survival rate through time returns the quantized probability density function $p(RV_{days}(v) \in [t, t+1))$.

$$p(RV_{days}(v) \in [t, t+1)) = \hat{y_t} \cdot \prod_{r<t} (1 - \hat{\lambda_r}). \tag{4.1}$$

2. Then, the predicted revisit interval can be represented as a form of the expected value as in Eq. 4.2.

$$\hat{RV}_{days}(v) = \sum_{t=0}^{k} (t + 0.5) \cdot p(RV_{days}(v) \in [t, t+1)). \tag{4.2}$$

3. By using the last time of the observation period together, it is possible to predict whether or not a revisit is made within a period, denoted as $\hat{RV}_{bin}(v)$. Here, we define a *suppress time* $t_{supp}(v) = t_{end} - t_v$ where t_v denotes a visit time of v, and t_{end} denotes a time of observation ends. We name it as *suppress time* to convey the meaning that the customer suppresses his/her desire to revisit until the time of observation ends by not visiting the store again.

$$\hat{RV}_{bin}(v) = \begin{cases} 1 & \text{if } \hat{RV}_{days}(v) \leq t_{supp}(v) \\ 0 & \text{if } \hat{RV}_{days}(v) > t_{supp}(v). \end{cases} \tag{4.3}$$

4. Calculating survival rate with a suppress time gives CCDF and CDF. CCDF and CDF will be used to compute the cross entropy loss. When $t_{supp}(v)$ is a natural number, the following holds.

$$p(RV_{days}(v) \geq t_{supp}(v)) = \prod_{r<t_{supp}(v)} (1 - \hat{\lambda_r}). \tag{4.4}$$

$$p(RV_{days}(v) < t_{supp}(v)) = 1 - \prod_{r<t_{supp}(v)} (1 - \hat{\lambda_r}). \tag{4.5}$$

4.3.5 Loss Functions

We designed a custom loss function to learn parameters of our *SurvRev* model. We defined four types of losses—negative log-likelihood loss ($\mathcal{L}_{uc-nll}$), RMSE loss ($\mathcal{L}_{uc-rmse}$), cross entropy loss ($\mathcal{L}_{uc-ce} + \mathcal{L}_{c-ce}$), and pairwise ranking losses ($\mathcal{L}_{uc-c-rank}$). The custom loss function is a combination of each loss. The prefixes $\mathcal{L}_{uc}$, $\mathcal{L}_{c}$, and $\mathcal{L}_{uc-c}$ mean that each loss is calculated for uncensored, censored, both samples, respectively. Table 4.2 summarizes the main aspects of our loss functions which will be described in the following subsections.

Table 4.2: Summary of losses used in *SurvRev* model.

Notation	Meaning	Details	References
$\mathcal{L}_{uc-nll}$	Negative log likelihood	Maximizing likelihood of revisit probability in particular time bin. Designed five sub-losses $\mathcal{L}_{uc-nll-date}$, $\mathcal{L}_{uc-nll-week}$, $\mathcal{L}_{uc-nll-month}$, $\mathcal{L}_{uc-nll-season}$, and $\mathcal{L}_{uc-nll-day}$. e.g., $\mathcal{L}_{uc-nll-date}$: If the customer revisited in 29.5 days, the value of $p(RV_{days}(v) \in [29, 30))$ should be high.	Extension of Ren et al [95].
$\mathcal{L}_{uc-rmse}$	RMSE	The error between predicted revisit interval and actual revisit interval. Defined as $MSE(\hat{RV}_{days}(v), RV_{days}(v))$.	One of the evaluation metric in Kim et al [52].
$\mathcal{L}_{uc-ce}$	Binary cross entropy	Partial binary cross entropy value between $P(RV_{days}(v) \leq t_{end})$ and $RV_{bin}(v)$, for $v \in V_{uncensored}$.	$\mathcal{L}_{uncensored}$ in Ren et al [95].
$\mathcal{L}_{c-ce}$	Binary cross entropy	Partial binary cross entropy value between $P(RV_{days}(v) \geq t_{end})$ and $RV_{bin}(v)$, for $v \in V_{censored}$.	$\mathcal{L}_{censored}$ in Ren et al [95].
$\mathcal{L}_{uc-c-rank}$	Pairwise ranking loss	Minimize the number of wrong pairs. The pair is wrong if $\hat{RV}_{days}(v_i) > \hat{RV}_{days}(v_j)$ when $RV_{days}(v_i) \leq RV_{days}(v_j)$.	$\mathcal{L}_2$ in Lee et al [95].

Negative Log-likelihood Loss: $\mathcal{L}_{uc-nll}$

Since there is no ground truth revisit interval distribution, we would like to maximize the likelihood of the empirical data distribution. For consistency, we convert it to *negative log-likelihood loss* $\mathcal{L}_{uc-nll}$ (NLL loss) by setting a negative log on top of the likelihood. Minimizing the negative log-likelihood loss obtains the same effect of maximizing the likelihood value itself. $\mathcal{L}_{uc-nll}$ computation is only done for uncensored samples v in training set that have a valid value of next revisit interval $\forall RV_{days}(v) \in \mathbb{R}$.

For step-by-step optimization, we design five cases of $\mathcal{L}_{uc-nll}$ by changing interval parameters. Those are $\mathcal{L}_{uc-nll-date}$, $\mathcal{L}_{uc-nll-week}$, $\mathcal{L}_{uc-nll-month}$, $\mathcal{L}_{uc-nll-season}$, and $\mathcal{L}_{uc-nll-day}$. We explain it one by one by considering the case when $RV_{days}(v) = 29.5$.

- $\mathcal{L}_{uc-nll-date}$: If the customer revisited in 29.5 days, the model learns to increase the likelihood of daily interval $RV_{days}(v) \in [29, 30)$.
- $\mathcal{L}_{uc-nll-week}$: For the above case, the model considers that the customer revisits after 4–5 weeks, so it learns to increase the likelihood of weekly interval $RV_{days}(v) \in [28, 35)$.
- $\mathcal{L}_{uc-nll-month}$: Similarly, the model divides the interval into monthly bins, so it learns to increase the likelihood of monthly interval $RV_{days}(v) \in [30, 60)$.

- $\mathcal{L}_{uc\text{-}nll\text{-}season}$: For some applications (i.e., clothing), it is important to capture seasonal visitation pattern. This option allows the model to capture the likelihood of 3-month long interval, so it learns to increase the likelihood of the first interval $RV_{days}(v) \in [0, 90)$.
- $\mathcal{L}_{uc\text{-}nll\text{-}day}$: There may be a customer who visits the store only on weekdays or visits only on weekends. So it is necessary to consider the information about the day of the week for predicting $RV_{days}(v)$. If the revisited day is Saturday, then this loss function allows model to increase the likelihood of $t_v + RV_{days}(v) \in \text{Saturday}$.

Depending on the task domain, the losses to focus on will be slightly different. In total, the final *negative log-likelihood loss* $\mathcal{L}_{uc\text{-}nll}$ can be represented as a weighted sum of those five losses.

RMSE Loss: $\mathcal{L}_{uc\text{-}rmse}$

The second loss is a *Root Mean Squared Error* (RMSE) loss $\mathcal{L}_{uc\text{-}rmse}$ which minimizes the error between predicted revisit interval $\hat{RV}_{days}(v)$ and the actual interval $RV_{days}(v)$. $\mathcal{L}_{uc\text{-}rmse}$ makes the model to minimize its value for uncensored samples. One can think that the RMSE loss is a continuous expansion of negative log-likelihood loss. Unlike NLL loss, RMSE value can be computed even if the predicted value does not belong to a certain value range, which corresponds to a certain bin in $\mathcal{L}_{uc\text{-}nll}$.

Cross Entropy Loss: $\mathcal{L}_{uc\text{-}c\text{-}ce}$

The cross entropy is the first set of losses that can be measured for both censored and uncensored visits. The cross entropy loss $\mathcal{L}_{uc\text{-}c\text{-}ce}$ measures the performance of a classification model whose output is a probability value between 0 and 1. It decreases as the predicted probability converges to the actual label. We separate $\mathcal{L}_{uc\text{-}c\text{-}ce}$ into $\mathcal{L}_{uc\text{-}ce}$ and $\mathcal{L}_{c\text{-}ce}$ denoting the partial cross entropy value of uncensored set and censored set, respectively:

$$\mathcal{L}_{uc\text{-}c\text{-}ce} = \mathcal{L}_{uc\text{-}ce} + \mathcal{L}_{c\text{-}ce}. \tag{4.6}$$

Theorem 4.3.1. *The partial cross entropy loss $\mathcal{L}_{uc\text{-}ce}$ for a set of censored visits is equivalent to the negative log-likelihood. That is,*

$$\mathcal{L}_{c\text{-}ce} = -\sum_{v \in V_{censored}} \log p(RV_{days}(v) > t_{supp}(v)). \tag{4.7}$$

Proof. Let us derive the partial cross entropy loss $\mathcal{L}_{c\text{-}ce}$ for censored cases. From the definition of the cross entropy $H(p,q) = -\sum_{x \in X} p(x) \log q(x)$, we can interpret that $q(x)$ corresponds to the CCDF value for the censored case, which can be expressed as $p(RV_{days}(v) > t_{supp}(v))$. Also, $p(x) = 1$ because we know the $RV_{bin}(v)$ value of the training set for sure. After substitution, the result $-\sum_{x \in X} \log q(x)$ is equivalent to the definition of the negative log-likelihood.

$$\begin{aligned} H(p,q) &= -\sum_{x \in X} p(x) \log q(x) \\ &= -\sum_{v \in V_{censored}} 1 \cdot \log p(RV_{days}(v) > t_{supp}(v)). \end{aligned} \tag{4.8}$$

□

By minimizing $\mathcal{L}_{c\text{-}ce}$, the model is trained in the direction of CCDF increasing to one. This is desirable for censored cases where $RV_{bin}(v) = 0$. Although we do not have any information about the true event time for a censored case [95], in *SurvRev* we are able to calculate the CCDF, $p(RV_{days}(v) > t_{supp}(v))$, from the event rates $\hat{\boldsymbol{\lambda}}$, as explained in § 4.3.4.

Corollary 4.3.1.1. *The partial cross entropy loss $\mathcal{L}_{uc\text{-}ce}$ for a set of uncensored visits is equivalent to the negative log-likelihood. That is,*

$$\mathcal{L}_{uc\text{-}ce} = -\sum_{v \in V_{uncensored}} \log p(RV_{days}(v) \leq t_{supp}(v)). \tag{4.9}$$

Proof. It is self-evident by interpreting $q(x)$ as the CDF value for the censored case. Because the uncensored visit occurs when the customer returns before the observation time is over, the loss between the CDF value $q(x) = p(RV_{days}(v) \leq t_{supp}(v))$ and $p(x) = 1$ should be minimized. □

By minimizing $\mathcal{L}_{uc\text{-}ce}$, the model is trained in the direction of CDF increasing to one. This is for uncensored cases where $RV_{bin}(v) = 1$.

Pairwise Ranking Loss: $\mathcal{L}_{uc\text{-}c\text{-}rank}$

Motivated by ranking loss function [59] and a c-index [93], we introduce the pairwise ranking loss to compare the orderings between predicted revisit intervals. This loss function is to fine-tune the model by making the tendency of the predicted intervals and the actual intervals similarly. The loss function $\mathcal{L}_{uc\text{-}c\text{-}rank}$ is formally defined by following steps.

1. First, we define two matrices P and Q as follows:

$$\begin{aligned} P_{ij} &= sign(\hat{RV}_{days}(v_j) - \hat{RV}_{days}(v_i)) \\ Q_{ij} &= sign(RV_{days}(v_j) - RV_{days}(v_i)). \end{aligned} \tag{4.10}$$

For a censored visit v, we use a suppress time $t_{supp}(v)$ instead of using an actual revisit interval $RV_{days}(v)$. The substitution is for making a comparison between uncensored and censored cases. For example, two visits v_i and v_j are comparable when v_i satisfies $RV_{days}(v_i) = 3$ and v_j satisfies $t_{supp}(v_j) = 5$.

2. Then, we define a new matrix U as a Hadamard product of P and $-Q$.

$$U = P \odot -Q. \tag{4.11}$$

3. Next, we define a new matrix W as follows.

$$\hat{W}_{ij} = \begin{cases} 1 & \text{if } min(t_{supp}(v_j), RV_{days}(v_j)) \geq RV_{days}(v_i) \\ 0 & \text{else.} \end{cases} \tag{4.12}$$

4. The loss is defined as follows:

$$\mathcal{L}_{uc\text{-}c\text{-}rank} = \sum_{i<j, v_i \in V_{uncensored}} U_{ij} \cdot W_{ij}. \tag{4.13}$$

By minimizing $\mathcal{L}_{uc\text{-}c\text{-}rank}$, our model encourages correct ordering of pairs and discourage incorrect ordering of pairs. The constraint $v_i \in V_{uncensored}$ is added to ignore incomparable pairs. A binary variable W_{ij} also removes the effect of incomparable pairs due to the censoring effect such as v_i and v_j with $RV_{days}(v_i) = 3$ and $t_{supp}(v_j) = 2$, respectively. In this way, the final loss function behaves similar to c-index metric.

Final Loss

Combining all the losses, we can design our final objective $\mathcal{L}$ to train our *SurvRev* model.

$$\arg\min_{\theta} \mathcal{L} = \arg\min_{\theta} \mathcal{L}_{uc-nll} \cdot \mathcal{L}_{uc-rmse} \cdot \mathcal{L}_{uc-c-ce} \cdot \mathcal{L}_{uc-c-rank}.$$

where θ is a model parameter of *SurvRev.* The reason for using the product loss instead of using the sum of each loss is to reduce parameters to control the balance between losses to stabilize model training.

4.4 Experiments

To prove our model's excellence, we performed diverse experiments on real-world customer mobility dataset. We first introduce our efforts to create a revisit prediction benchmark dataset. After introducing the tuned parameter values of the *SurvRev* model, we briefly summarize the evaluation metrics needed for revisit prediction (All in § 4.4.1). We show the superiority of our *SurvRev* model by comparison with *seven* different baseline event prediction models (§ 4.4.2).

4.4.1 Settings

Data Preparation

We prepared another set of revisit prediction benchmark dataset by following principles introduced in Chapter 4.1. This new version of the dataset mimicked much more realistic prediction setting than the one in the previous chapter. The dataset consists of customer trajectory inside five different off-line stores. We consider each store independently since there are not many customer overlaps between the stores. Here is a brief step-by-step introduction on how to generate this dataset.

From the dataset of seven stores used in the previous chapter, we chose five featured stores L_GA, L_MD, O_MD, E_GN, E_SC and renamed as A, B, C, D, and E, respectively. We removed A_GN and A_MD due to their relatively short data collection periods, which are around 220 days. For the remaining five stores, we left only one year amount of data from 2017-01-01 to 2017-12-31. We decided to use the year 2017 data to keep coherency since the data provider changed the area sensing logics at the end of 2016, also we think the remaining data is long enough to study customer revisit.

We randomly selected $50,000$ device ID (customer) for the benchmark since we knew that footprints from $50,000$ users are large enough to guarantee the model performance (§ 3.3.3). We also prepared toy datasets with $1,000$ and $5,000$ customers, the smaller data is a subset of a bigger one. In this experiment, we followed a *10-minute rule* for grouping sessions into visits. If a customer session reappears within 10 minutes, we do not consider the subsequent visit as a new visit. We also made a several version of training and testing set by varying training length—$60, 120, 180, 240, 300$ days. Among these, we used two datasets trained for 180 days and 240 days. Due to the time constraint, we included results by running on smaller datasets with $5,000$ users.

We carefully designed the dataset to prevent leakage. While generating testing set, we only retained the first visit to prevent unjustified predictions. Otherwise, one can directly calculate the revisit interval of the former visit by comparing two visit dates. Besides, while generating labels for training data, we only used the observations until the end of the training period. For instance, the binary label is zero even if the customer revisited during a testing timeframe. On the other hand, the label is updated to one in a train-censored set.

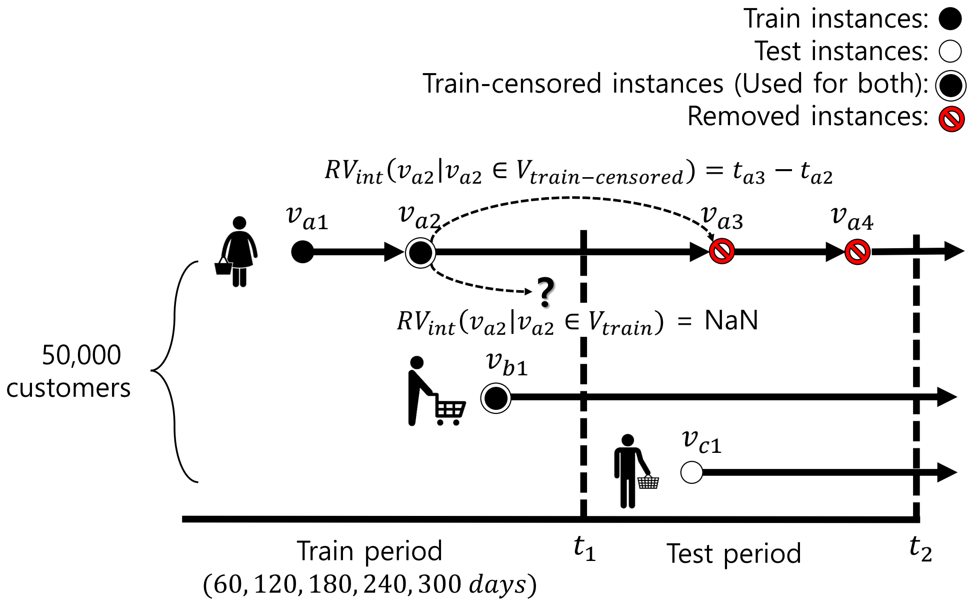

Train instances: ●
Test instances: ○
Train-censored instances (Used for both): ◉
Removed instances: 🚫
$RV_{int}(v_{a2}|v_{a2} \in V_{train-censored}) = t_{a3} - t_{a2}$
v_{a1}
v_{a2}
v_{a3}
v_{a4}
?
$RV_{int}(v_{a2}|v_{a2} \in V_{train})$ = NaN
50,000 customers
v_{b1}
v_{c1}
Train period
$(60, 120, 180, 240, 300\ days)$
t_1
Test period
t_2

Figure 4.8: Explaining revisit prediction benchmark dataset.

Example. Figure 4.8 illustrates our data settings by showing three customers a, b, c. Among six visits, three visits, v_{a1}, v_{a2} and v_{b1}, and their revisit status up to time t_1 are used as a training set, only v_{a1} is recorded as a visit with a revisit. The testing set includes a single event—v_{c1}—which visited the store for the first time after time t_1. The train-censored set includes two censored cases—v_{a2} and v_{b1}. To prevent data leakage, we exclude v_{a3} and v_{a4} from a testing instance. In the train-censored set, a revisit ratio is 1/2 since v_{a2} is recorded as a revisited case. In summary, the model requires to predict revisit of three testing instances—v_{c1}, v_{a2}, and v_{b1}—until time t_2 by having information from the training set. We conjecture that the performance of this off-line batch prediction can be the lower bound of the actual online testing. In the actual setting, the model can be continuously refined as the prediction progresses.

Data Statistics Table 4.3 describe several statistics of the datasets used in this chapter. V_{tr}, V_{te}, and V_{tc} are the acronyms of the training set, testing set, and train-censored set. We can observe the huge difference of average revisit rate $E[RV_{bin}(v)]$ and average revisit interval $E[RV_{days}(v)]$ between three sets. There are several things to note in order to understand the statistics.

- $E[RV_{bin}(v_{te})]$ is relatively smaller than $E[RV_{bin}(v_{tr})]$. The difference between two values are caused by data removal explained in the previous example. V_{tr} includes multiple visits for regular customer, whereas V_{tc} includes a single visit for each customer.
- Interestingly, $E[RV_{bin}(v_{tc})]$ is relatively smaller than $E[RV_{bin}(v_{te})]$. This statistics is counterintuitive since the censored instances v_{tc} secure longer time to be recognized as a revisit, and we expected that customers who visited earlier have more chance to revisit during the remaining time. To explain through Figure 4.8, censored instances v_{tc} can fully use the testing period $[t_1, t_2]$ until the observation ends. However, new visitors v_{te} have shorter remaining time than v_{tc} since they visit the store during the testing period $[t_1, t_2]$. These unexpected outcomes are due to the *decreasing customer interest* over time. We conjecture that some censored customers v_{tc} already lost the intention to revisit, so their average revisit probability is lower than v_{te}, in spite of having more available time to revisit. Figure A.3(c) in Appendix A proves this phenomenon by showing how $E[RV_{bin}(v)]$ in each dataset changes over time.

Table 4.3: Statistics of the datasets (training length = 180 days).

(a) Statistics on dividing first 180 days as training period.

Store ID	A (L_GA)	B (L_MD)	C (O_MD)	D (E_GN)	E (E_SC)
Length (days)	365	365	307	300	312
Sensors	14	11	27	40	22
# of sessions	745,182	948,763	1,322,130	1,572,592	1,491,874
$\lvert V_{tr}\rvert$	39,473	45,051	70,173	35,259	50,898
$\lvert V_{te}\rvert$	24,166	25,664	21,288	20,907	22,991
$\lvert V_{tc}\rvert$	31,409	29,511	36,462	28,069	32,208
$E[RV_{bin}(v_{tr})]$	0.204	0.345	0.480	0.204	0.367
$E[RV_{bin}(v_{te})]$	0.204	0.285	0.381	0.116	0.233
$E[RV_{bin}(v_{tc})]$	0.191	0.203	0.242	0.115	0.223
$E[RV_{days}(v_{tr})]$	38.7	26.4	24.3	33.9	31.0
$E[RV_{days}(v_{te})]$	45.7	30.2	19.0	28.3	30.6
$E[RV_{days}(v_{tc})]$	165.2	137.1	105.6	107.0	109.6

(b) Statistics on dividing first 240 days as training period.

Store ID	A (L_GA)	B (L_MD)	C (O_MD)	D (E_GN)	E (E_SC)
Length (days)	365	365	307	300	312
Sensors	14	11	27	40	22
# of sessions	794,635	1,061,244	1,534,905	1,769,480	1,723,720
$\lvert V_{tr}\rvert$	49,987	57,961	88,692	48,550	67,745
$\lvert V_{te}\rvert$	11,403	12,963	5,540	8,260	7,494
$\lvert V_{tc}\rvert$	38,179	36,193	43,378	37,562	40,474
$E[RV_{bin}(v_{tr})]$	0.236	0.376	0.511	0.226	0.403
$E[RV_{bin}(v_{te})]$	0.140	0.183	0.260	0.053	0.112
$E[RV_{bin}(v_{tc})]$	0.146	0.152	0.140	0.058	0.126
$E[RV_{days}(v_{tr})]$	52.5	34.2	30.7	40.9	37.4
$E[RV_{days}(v_{te})]$	30.0	15.2	7.1	13.4	17.2
$E[RV_{days}(v_{tc})]$	159.8	129.6	92.7	104.2	103.2

- The value $RV_{days}(v)$ is calculated among uncensored visits. $E[RV_{days}(v_{tc})$ is very long compared to $E[RV_{days}(v_{tr})$ and $E[RV_{days}(v_{te})$ since their possible revisit interval spans up to t_2.
- We leave the additional exploratory data analysis of our benchmark data in Appendix A.

Hyperparameter Settings

In our experiments, we prepared two sets of *pretrained embeddings* to feed inputs to our model. Since we limit the number of user subset for benchmark datasets, the pretrained embeddings are also learned from the trajectories generated by that user subset. The embedding dimension is set as 64 for both *area embeddings* and the *user embeddings*. A set of new IDs and a set of new areas in the testing set are mapped to [unk] and embedded to default values. Since it is finished within one minute, the code to generate those embeddings runs *on-the-fly*. For the low-level module, we use 64-dim Bi-LSTM unit with masking padded areas. The kernel size of CNN is 3, with 16-dim filters, and the number of neurons in the FC layer is 128. We use one dense layer. For a visit with long sequence, we considered up to k areas that can cover up to 95% of all cases, where k is dependent on each dataset. In the high-level module, the dynamic LSTM has 256-dim units and process up to 5 events. We used two layers of LSTM with tanh activation. The number of neurons in the final FC layer is 365. We used two FC layers with ReLU activation. For training the model, we used Adam [56] optimizer with learning rate of 0.001. We set the mini-batch size as 32 and run 10 epochs for 1k and 5k datasets and run one epoch for 50k datasets. NLL loss $\mathcal{L}_{uc-nll}$ is set as the averages of $\mathcal{L}_{uc-nll-month}$ and $\mathcal{L}_{uc-nll-season}$. Some of these hyperparameters were selected empirically by grid search.

Input Settings

We made a switch to control a number of user histories to use when training a *SurvRev* model. For predicting *last visits* (train-censored instances), we used all visits to train the model. For instance, if an input visit v_5 is followed by multiple previous visits, the logs prior to this visit are fed together in high-level event rate predictor. At the same time, each of his/her prior visits $(v_1, \cdots, v_4)$ is used as a separate input for the *SurvRev* model. For predicting *first-time visitors* (testing instances), only the first appearances $(v_1 \in V_{train})$ were used to train the model. Since there are not exist any prior log for each training instance, the LSTM length in a high-level event rate predictor is always one.

Evaluation Metrics

We used *three* evaluation metrics. *F-score*, and *concordance-index* (c-index) are used for evaluating uncensored and censored visits together. Also, *root mean squared error (RMSE)* is used for evaluating uncensored cases representing revisited customers. We removed *accuracy* from our evaluation criteria since it loses its effectiveness on measuring performance from the imbalanced dataset.

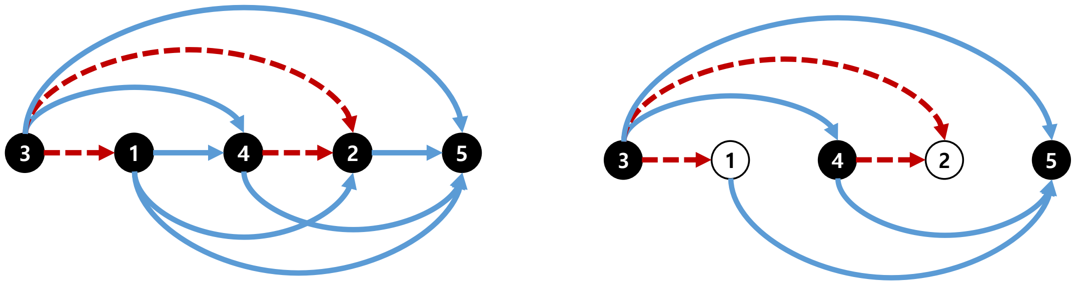
- *F-score*: An another metric for measuring binary revisit classification performance.
- *C-index* [93]: A metric for measuring global pairwise ordering performance, the most commonly used evaluation metric in survival analysis [59, 95]. Figure 4.9 illustrates how to calculate the c-index.
- *RMSE*: A metric for measuring error between predicted and golden revisit interval. RMSE can only be measured for cases when their golden revisit intervals exist.

4.4.2 Results

Comparison with Baselines

We verify the effectiveness of our *SurvRev* model on the large-scale in-store customer mobility data. For comparison, we implemented *eight* different event prediction models to fit our datasets. The baselines include well-known stochastic processes, a semi-parametric statistical model, a state-of-the-art gradient boosting model, and deep survival analysis models. Detailed explanations of these models are described as below:

Baselines Without Considering Covariates First three baselines focus on the distribution of revisit labels and consider them as an arrival process. These baselines do not consider the attributes (=covariates) obtained from each visit.



(a) Before censoring, c-index is $\frac{7}{10}$.

(b) After censoring, c-index is $\frac{4}{7}$.

Figure 4.9: An illustration to describe a c-index metric. Five numbers inside the nodes are predicted values—3, 1, 4, 2, 5 and the order of the nodes represents their actual value—1, 2, 3, 4, 5. The blue solid line means that the ordering of the predicted values of the two nodes at both ends of the line matches the ordering of the actual values. In opposite, the red dotted line represents the case when the ordering is violated. By censoring instances (black → white), some pairs are deleted since they cannot be compared anymore. As a result, the c-index value is changed from $\frac{7}{10}$ to $\frac{4}{7}$.

- *Majority Voting (Majority)*: Prediction results follow the majority class for classification, follow the average values for regression; this baseline is naïve but powerful for an imbalanced dataset.
- *Personalized Poisson Process (Poisson)* [97]: A stochastic process used to describe customer arrival patterns. We assume customer inter-arrival time follows an exponential distribution with a constant λ. To make it personalized, we control λ for each customer by regarding its visit frequency and observation time.
- *Personalized Hawkes Process (Hawkes)* [34]: The Hawkes process is an extended version of the Poisson Process which includes self-stimulation and time-decaying function on the rate λ. We referred the well-written post[6] and the repository[7] for our baseline implementation.

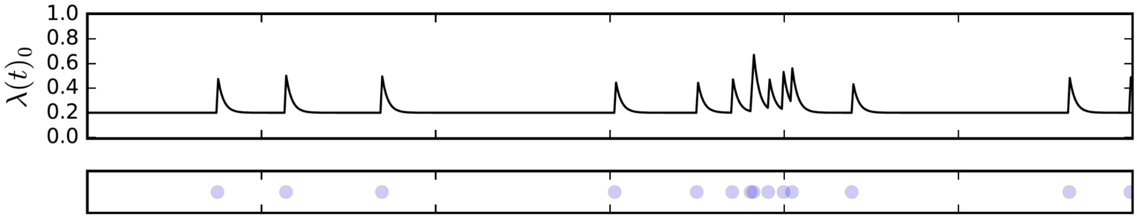


Figure 4.10: Hawkes process.

Baselines Considering Covariates Following two models considered covariates derived from each visit. The Cox proportional hazard model focuses on handling censored data and the gradient boosting tree model pointed out the interaction and correlation between features. For fairness, we used the same set of handcrafted feature for the latter baseline.

- *Cox Proportional Hazard model (Cox-PH)* [17]: Semi-parametric survival analysis model with proportional hazards assumption. Widely used baseline for survival analysis task.
- *Gradient Boosting Tree with Handcrafted Features (XGBoost)* [52]: Using carefully designed handcrafted features with XGBoost classifier [14].

[6] `https://stmorse.github.io/journal/Hawkes-python.html`
[7] `https://github.com/stmorse/hawkes`

Baselines Using Deep Survival Analysis The last two models are state-of-the-art survival analysis models that applied deep learning. The first one is specialized for event processes having long-term histories. The second model focuses on a death-or-survival scenario with quantized event rates. Deep learning models have strength in predicting event rate from an unknown distribution. From their model architecture, we expected the former model has strength for predicting train-censored case and the latter model has strength for predicting first-time visitor case newly appeared in a testing timeframe.

- *Neural Survival Recommender (NSR)* [43]: A deep multi-task learning model with LSTM and 3-way factor unit used for music data with sequential events. The downside of this model is that the input for each cell is simple, which did not consider lower-level interactions.

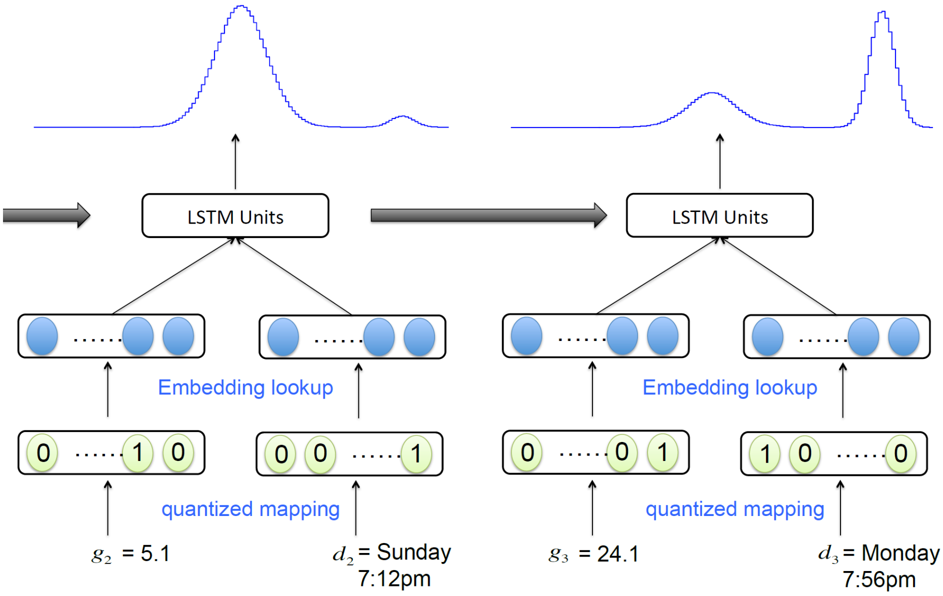


Figure 4.11: The architecture of the NSR baseline [43].

- *Deep Recurrent Survival Analysis (DRSA)* [95]: An auto-regressive model with LSTM. Each cell emits a hazard rate for each timestamp. The downside of this model is that each LSTM considers only a single event.

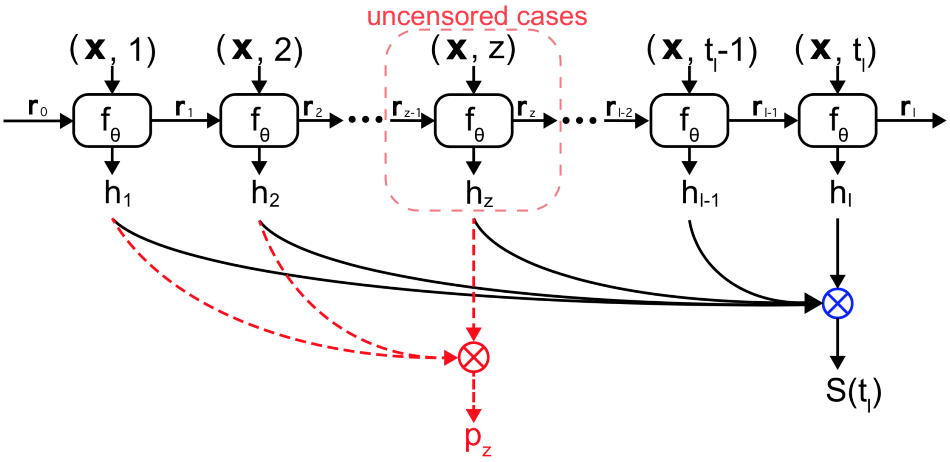


Figure 4.12: The architecture of the DRSA baseline [95].

Comparison Results Table 4.4 and Table 4.5 summarize the performance of each model on two different testing sets named as a train-censored set (V_{tc}) and a testing set (V_{te}), respectively. By predicting V_{tc}, we expect to see the superiority of our model on censored data. Also, testing on V_{te} shows that our model can effectively predict the revisit behavior of new customers. The result of the prediction on train-censored set shows that the c-index of *SurvRev* outperforms other baselines except for our prior model. On the testing set with first-time visitors, *SurvRev* outperforms other baselines on optimizing f-score and shows its effectiveness on optimizing RMSE. As a preliminary result, it is quite satisfying to observe that our model showed its effectiveness on two different settings. However, it may be necessary to further tune our model parameters to achieve the best results for every evaluation metrics.

Table 4.4: Superiority of *SurvRev* compared to baselines, evaluated on train-censored set. We highlighted in **bold** for cases when *SurvRev* shows the best performance among competitors.

(a) **C-index (180 days).**

	Store A	Store B	Store C	Store E
Majority	0.500	0.500	0.500	0.500
Poisson	0.528	0.591	0.588	0.582
Hawkes	0.530	0.593	0.588	0.580
XGBoost	0.420	0.597	0.671	0.549
NSR	0.497	0.497	0.480	0.523
DRSA	0.500	0.500	0.499	0.500
SurvRev	**0.561**	**0.672**	0.649	**0.647**

(b) C-index (240 days).

	Store A	Store B	Store E
Majority	0.500	0.500	0.500
Poisson	0.552	0.622	0.617
Hawkes	0.549	0.624	0.613
XGBoost	0.667	0.568	0.830
NSR	0.509	0.513	0.504
DRSA	0.500	0.500	0.501
SurvRev	0.606	**0.726**	0.702

(c) F-score (180 days).

	Store A	Store B	Store C	Store E
Majority	0.000	0.000	0.000	0.000
Poisson	0.177	0.140	0.187	0.198
Hawkes	0.177	0.135	0.187	0.199
XGBoost	0.083	0.476	0.415	0.414
NSR	0.291	0.244	0.190	0.173
DRSA	0.328	0.337	0.390	0.356
SurvRev	**0.328**	0.337	0.390	0.356

(d) F-score (240 days).

	Store A	Store B	Store E
Majority	0.000	0.000	0.000
Poisson	0.198	0.157	0.168
Hawkes	0.200	0.155	0.172
XGBoost	0.282	0.475	0.274
NSR	0.192	0.137	0.071
DRSA	0.268	0.274	0.224
SurvRev	0.268	0.274	0.224

(e) RMSE (180 days).

	Store A	Store B	Store C	Store E
Majority	108.846	101.433	81.651	79.550
Poisson	180.314	156.230	145.172	156.522
Hawkes	178.164	154.771	143.396	154.709
XGBoost	121.398	116.429	67.458	91.068
NSR	97.067	120.527	138.389	138.003
DRSA	184.234	156.208	123.454	126.421
SurvRev	170.969	144.604	112.603	114.263

(f) RMSE (240 days).

	Store A	Store B	Store E
Majority	92.117	93.360	72.894
Poisson	222.614	183.899	197.638
Hawkes	220.329	186.186	198.293
XGBoost	86.435	123.232	86.118
NSR	108.833	127.130	147.440
DRSA	175.912	155.168	121.681
SurvRev	163.069	144.050	110.525

Table 4.5: Superiority of *SurvRev* compared to baselines, evaluated on testing set.

(a) C-index (180 days).

	Store A	Store B	Store C	Store E
Majority	0.500	0.500	0.500	0.500
Poisson	0.505	0.506	0.497	0.499
Hawkes	0.504	0.504	0.499	0.507
Cox-ph	0.602	0.586	0.476	0.586
XGBoost	0.514	0.471	0.503	0.510
NSR	0.495	0.499	0.500	0.501
DRSA	0.498	0.499	0.501	0.496
SurvRev	0.499	0.494	0.501	0.505

(b) C-index (240 days).

	Store A	Store B	Store E
Majority	0.500	0.500	0.500
Poisson	0.506	0.499	0.501
Hawkes	0.510	0.501	0.504
Cox-ph	0.616	0.560	0.630
XGBoost	0.420	0.507	0.509
NSR	0.499	0.501	0.507
DRSA	0.499	0.502	0.494
SurvRev	0.489	0.499	0.495

(c) **F-score (180 days).**

	Store A	Store B	Store C	Store E
Majority	0.000	0.000	0.000	0.000
Poisson	0.244	0.302	0.415	0.244
Hawkes	0.242	0.304	0.412	0.241
Cox-ph	0.286	0.353	0.176	0.000
XGBoost	0.236	0.317	0.248	0.097
NSR	0.000	0.000	0.000	0.000
DRSA	0.298	0.360	0.461	0.277
SurvRev	**0.315**	**0.373**	0.458	**0.295**

(d) **F-score (240 days).**

	Store A	Store B	Store E
Majority	0.000	0.000	0.000
Poisson	0.214	0.275	0.204
Hawkes	0.212	0.276	0.209
Cox-ph	0.000	0.000	0.000
XGBoost	0.025	0.194	0.000
NSR	0.000	0.000	0.000
DRSA	0.245	0.300	0.223
SurvRev	**0.272**	**0.307**	**0.263**

(e) RMSE (180 days).

	Store A	Store B	Store C	Store E
Majority	63.901	59.692	46.276	45.847
Poisson	328.468	353.567	315.713	312.333
Hawkes	328.025	356.020	321.984	299.873
Cox-ph	113.263	121.800	108.838	131.550
XGBoost	40.321	36.401	21.508	25.260
NSR	192.233	211.235	215.902	204.495
DRSA	58.575	39.671	26.187	36.783
SurvRev	48.428	**34.424**	22.743	28.289

(f) RMSE (240 days).

	Store A	Store B	Store E
Majority	87.911	77.485	66.294
Poisson	381.737	401.879	364.398
Hawkes	376.155	415.814	367.684
Cox-ph	159.669	175.057	181.847
XGBoost	29.315	22.872	16.817
NSR	201.367	217.084	213.840
DRSA	42.322	26.703	23.202
SurvRev	32.842	24.072	17.989

Ablation Studies

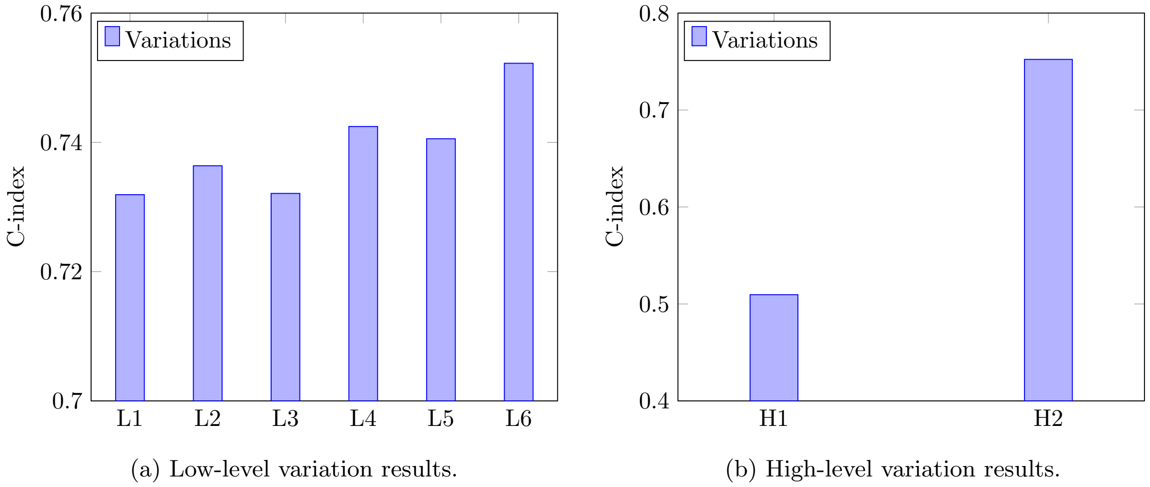
Ablation studies refer to experimental studies by removing some components of the model and seeing how that affects the model performance. Our *SurvRev* model is composed of two modules, *the low-level encoder* and *the high-level event rate predictor*. Throughout this analysis, we expect to observe the effectiveness of each module. In particular, we would like to show the performance gain by applying multiple layers in the low-level encoder.

Ablation by simplifying the low-level module First, we *remove* some components in the low-level encoder and see the contributions of each component. *Five* types of simplified low-level encoders are designed for ablation studies. For this experiment, the high-level module is maintained without any change. Variations of low-level encoders are as follows:

- L1 (*Bi-LSTM+ATT*): Use only Bi-LSTM and attention layers to represent the visit.
- L2 (*CNN+ATT*): Use only CNN and attention layers to represent the visit.
- L3 (*Bi-LSTM+CNN+AvgPool*): Substitute an attention layer to global average pooling.
- L4 (*Bi-LSTM+CNN+ATT*): Use only area sequences through three consecutive layers.
- L5 (*Bi-LSTM+CNN+ATT+UserID*): Add user embedding results to L4.
- L6 (*Bi-LSTM+CNN+ATT+UserID+FE*): Add handcrafted features to L5. This is equivalent to our original *low-level encoder* described in § 4.3.2.

Ablation by simplifying the high-level module Second, we simplify the high-level event rate predictor to see the effectiveness of our original LSTM architecture. Variation of high-level event rate predictors are as follows:

- H1 (*FC+FC*): Concatenate the low-level encoder outputs and then apply a fully connected layer instead of LSTMs. Final fully connected layers are preserved without any change.
- H2 (*LSTM+FC*): Stack (`tf.stack`) the low-level encoder outputs and then apply two-level LSTM layers. This is equivalent to our original *high-level event rate predictor* described in § 4.3.3.



(a) Low-level variation results.

(b) High-level variation results.

Figure 4.13: Ablation studies of the *SurvRev* model.

Figure 4.13 summarizes the ablation study results. These representative c-index results are evaluated on a train-censored set of Store D having 1,000 customers with 240-day training interval. We found that *low-level visit encoder* and *the high-level event generator* are *both* necessary for *SurvRev* architecture. In particular, *the LSTM structure* in the high-level module is much more effective than its replacement, a fully connected layer. Also, we can see the effectiveness of each layer in the low-level encoder. Performance gain by having Bi-LSTM, CNN, ATT are $L4 - L2$, $L4 - L1$, and $L4 - L3$, respectively. Unlike our expectation, L5 shows slightly lower performance than L4 in this experiment. Nevertheless, the performance of our final low-level encoder, L6, is superior to all other variations.

4.4.3 Discussions

These are the possible discussion topics to fully understand our *SurvRev* model.

Finding the Best Parameters Between Loss We have worked on classification and regression tasks with different statistics. In our problem setting, it was difficult to optimize diverse prediction objectives at the same time, and we observed that a single metric cannot optimize all evaluation criteria. How to deal with a problem of competing objectives?

Due to the large portion of censored data, the majority of the customer does not revisit in our dataset. Because of this limitation, the majority of the data is censored. However, when they revisit, the interval between two visits tends to be very short. Since the majority of the data is censored, the cross-entropy loss $\mathcal{L}_{uc-ce}$ tends to judge that a new customer will not return. However, the RMSE loss $\mathcal{L}_{uc-rmse}$ tends to train to judge that a new customer will return soon because the uncensored data show that if the customer visits again, the revisit interval was short. If we configure the loss to optimize only one objective (e.g., cross entropy), the RMSE results in the final evaluation will be very poor although the prediction accuracy will be satisfactory. Therefore, the model has to learn both objectives unless imputing the revisit interval of censored visits. To mitigate the difficulties of finding weights, we applied the product loss as our objective instead of using the sum loss. However, computing gradient in the sum loss can be faster than computing gradient in the product loss. Although it can be a time-consuming task, finding the best parameters to calibrate between losses would be a final challenge for model optimization.

Finding the Right Activation Functions In addition to that, we empirically found that the activation function of each layer significantly affects the prediction result. At first, we believe that applying a softmax activation at the last fully-collected layer is ideal since the softmax function makes the sum of the event rate into one, thus it was possible to keep the revisit probability within a year is less than one. However, the prediction results for each visit was almost identical owing to the bootstrap rule[8]. Finding the right activation functions would be another big challenge that we have.

Case Study We can perform several case study to show the effectiveness of our model. First, we can break down the performance of our model by controlling the maximum number of visits on our datasets. By showing this, we will be able to emphasize the strength of our model in a particular experimental setting. Second, we will put additional effort to make our deep learning model interpretable in a real-world business. Interpretability is essential for human-AI cooperation, model debugging, and detecting bias[9]. By interpreting the model, we may suggest some unknown features relates to customer revisit by understanding outputs from each layer.

[8] Revisit probabilities were 0.632.

[9] `https://github.com/Microsoft/interpret`

4.5 Summary

In this chapter, we proposed an advanced model for customer revisit prediction. The *SurvRev* model is the result of our commitment to creating a better predictive framework that works in a more realistic environment. In summary, our *SurvRev* model successfully predicts revisit rates for next time horizon by encoding each visit and managing personalized history. By applying survival analysis concepts, we smoothly handled censored visits, that caused huge data imbalance to led our previous approach to test in a downsampled prediction scheme. *SurvRev* is also free from data distribution inconsistency according to the ratio of training and testing set length. By applying deep learning, *SurvRev* becomes free from any parametric assumption and can predict customer revisit from an unknown distribution. Our flexible deep learning model with quantized event rates makes us handle those difficulties common in real-world prediction scenarios. By comparing with diverse event prediction approaches, *SurvRev* shows its effectiveness on diverse prediction objectives. The last thing to mention is that *SurvRev* is a general deep survival analysis model, which can be extended to any prediction task having partial observations and sessions with multilevel sequences.

Chapter 5. Conclusions and Future Directions

In this dissertation, we address the problem of predicting customer revisit using indoor mobility data. As well as the revisit prediction framework, we present diverse findings related to the prediction task from a broad perspective. Taken together, the work has pushed forward the frontier of predictive analytics in the offline marketing domain, with academic contribution and business implications. We give an overview of our contributions below.

5.1 Contributions

> *The future depends on some graduate student who is deeply suspicious of everything I have said.*
>
> — Geoffrey Hinton

In this dissertation, we have presented an importance of revisit prediction in off-line retail analytics with two different approaches:

- **Customer revisit prediction:** In Chapter 2, we formally define a customer revisit prediction task and we introduce our efforts on real-world data collection. This work is the first revisit prediction study on large-scale off-line customer mobility data and our frameworks can be easily used in addition to seven stores.
  - We extensively survey related works from marketing to data mining.
  - We collect customer mobility from seven flagship stores which cover up to 2.5 years with *5.7 million* visits.
  - We introduce preprocessing steps to find meaningful customer mobility from raw Wi-Fi signals.
- **Revisit prediction by feature engineering:** In Chapter 3, we develop a fast-but-powerful gradient boosting tree model powered by extensive feature engineering.
  - We prove that in-store signals captured by customer mobility can be an important clue for predicting their future behavior.
  - We measure the predictive powers of the feature groups and find that store accessibility features measured by weak Wi-Fi signals is *very* effective to predict the customer revisit.
  - We also present the effect of changing data collection period and propose an efficient way of estimating revisit predictability, by obtaining lower-bound of revisit prediction accuracy.
  - We show that our model performs well on smaller datasets.
- **Revisit prediction by deep learning:** In Chapter 4, we develop a *SurvRev* model to predict future revisit rate. The deep survival analysis model can fully use partial observations and enhance the performance of our revisit prediction framework.
  - To meet with real-world application setting, we update data splitting rule and relax data sampling strategy.
  - To train a model from partial observations, we design an event rate predictor to generate the event rates of next k days and optimize through custom loss functions.
  - To learn a hidden representation of each visit, we design an effective encoder with the combination of Bi-LSTM, CNN, attention networks with pretrained embeddings.

- To show the effectiveness of the *SurvRev* model, we implement seven baseline approaches covering customer arrival process and state-of-the-art deep survival models. We demonstrate that *SurvRev* produces desired prediction results, by improving c-index by up to 12 %.

5.2 Impact and Achievements

The thesis has a potential impact on a wide range of domains where user data is collected and solving predictive analytics are required for their success. Below, we highlight the impact of our work on academia and industry.

- **Academic recognition and media coverage:**
  - Our work on feature engineering in revisit prediction framework [52] was selected as *one of the best papers in ICDM 2018* and invited to the Knowledge and Information Systems journal [53].
  - After the *ICDM* conference, I was invited to *Korea Computer Congress (KCC)* and *Institute for Basic Science* to present our work for domestic researchers.
  - This work was featured in *Science Concert in 4th Industrial Revolution*, KBS1 in Dec 2017[1].
  - The concept of the revisit prediction with indoor mobility and its preliminary report using n-gram features with two stores was presented in *KCC 2016* [51].
  - Thanks to this work, I was selected as the *best presenter* in the bi-annual colloquium in 2017 and I received the *outstanding research award 2018* from KAIST Graduate School of Knowledge and Service Engineering.
- **Collaboration with research institutions:**
  - We received a research grant from *Microsoft Research Asia* (MSRA) for the first part of the thesis, the title of the research project is 'Prediction of customer revisit intention using indoor movements in stores'.
  - Starting with the above project, We continued our research collaboration with MSRA Social Computing group. In fall 2018, I visited his group and developed pre-trained user embedding models for user segmentation in Bing.
- **Collaboration with companies:**
  - With the help of the preliminary submission of this work using two stores, I was able to get five additional datasets from *ZOYI corporation*[2], then completed the first part of the thesis with seven flagship stores data.
  - I led the acquisition of additional data for the extension of the research and achieve another partnership with a start-up *Loplat*[3]. Currently, we are sharing the progress of our research, and receiving data that meets our requirements. Received datasets are being used by our lab members and would be a valuable resource for my future work.
- **Offspring projects:**
  - With the feature engineering technique, I participated in the 2018 WSDM Cup and formed a team online. Our team achieved *10th* over 575 teams. The task was to predict customer churn in an online streaming service.
  - Two master students wrote their Master's thesis using the dataset I have acquired.

[1] `https://www.youtube.com/watch?v=aACf5iGeE8Y`

[2] `http://zoyi.co/ko`

[3] `https://loplat.com/`

- We released a benchmark revisit prediction dataset[4] and we welcome any offspring project or collaboration opportunity using this dataset.

5.3 Vision and Future Directions

Throughout this thesis, we develop algorithms for predicting using indoor mobility data, with a focus on customer revisit prediction. We conclude by taking a step back to find opportunities to have a practical impact on *retail analytics* and *data mining.*

Mining Inter-Store Mobility for Revisit Prediction

While indoor mobility data contain interesting micro-scale dynamics during the shopping process, it is very difficult to capture the macro-scale dynamics of human beings. Using an points-of-interest (check-in) datasets obtained from Loplat, we can find larger scale patterns that determine customer revisit prediction. We will get the benefit by adopting recent techniques used in next location prediction tasks using check-in data. We reckon that the multi-layer model with user and location contexts can mimic revisit decision-making process. We anticipate that the model initiated from revisit prediction task is more effective than the model generated from the next location prediction. The natural step to extend our model is to capture diverse revisit intervals by considering different types of revisit targets. As in the indoor mobility study, we will provide diverse insights to help practitioners in this field. Ideally, we would like to visualize the characteristics of revisit-friendly-location and loyal customers by understanding revisit mechanism and the willingness-to-revisit. Since we already have the dataset and willing to collaborate, we spent some pages on Appendix C on this topic for future reference.

Deploying Our Algorithms to See Real-World Impact

In the long run, we would like to tune our algorithm to work in a stream environment and deploy it to business in real time. We would like to work more closely with a company having a whole pipeline from data acquisition to marketing service. If it happens, we can measure the direct impact of our prediction algorithm and further give a benefit to merchants who are willing to retain customers, and to customers who are willing to get more hospitality and discount. Our research can be of particular interest to managers who operate stores and manage mobile applications with a data collection platform.

Learning Directly from Wi-Fi Signals

The strength and opportunity in deep learning come from the massive amount of data. I believe deep learning have potentials to find more effective patterns out of the raw signals. As we observed, Wi-Fi signals are collected several times a second and we can directly use these signals without having any man-made preprocessing step such as signal-to-session conversion. If we are able to make use of it, the final model can benefit from a much more continuous flow of customer motion pattern. The biggest challenge is to find the best way to apply machine learning on tera-scale raw Wi-Fi signals, which is 567 GB in the case of single store O_MD. Machine learning on this stream of Wi-Fi signals can be challenging but rewarding.

[4] `https://github.com/kaist-dmlab/revisit`

Chapter A. Benchmark Data

For fertilizing this field, we also released a real-world benchmark dataset for revisit prediction, which will be the first publicly available datasets as far as I know. We believe that our benchmark datasets can be used for diverse topics such as predicting stickiness[1] of the customer, next area prediction, or funnel analysis to increase an inflow rate. Here, we describe some data samples and statistics to become familiar with our customer mobility data.

Released Data Example Tables A.1 show the first ten rows of each database in our benchmark dataset. Table A.1a shows the description of each visit, each visit has its own ID and device ID with corresponding visit date. The last column represents a sequence of corresponding Wi-Fi sessions for the visit. Table A.1b shows the visit ID with its two labels, the first label corresponds to revisit interval $RV_{int}(v)$ and the second label correspond to revisit intention $RV_{bin}(v)$. Table A.1c is a Wi-Fi session data. It contains a core information to generate a trajectory for each visit. This data can be used by connecting with `train_visits.csv`.

Exploratory Data Analysis We append some exploratory data analysis to understand some statistics of the dataset. Figure A.1 shows the histogram of customer revisit interval, which is available for revisited instances. On the training and testing set, it follows a long-tail distribution. However, the distribution is different on train-censored set. In Figure A.2, we can observe that customer revisit occurs by having an interval with a multiple of 24 hours. Figure A.3 shows the number of weekly visitors and the average revisit interval according to the visit date. We can find some unexpected patterns due to the longitudinal data set up. Last, Figure A.4 describes the effect of increasing the training length.

[1] Stickiness is a marketing term to describe the average time per month at a site.

Table A.1: Description of each database.

(a) The first ten rows of `train_visits.csv`.

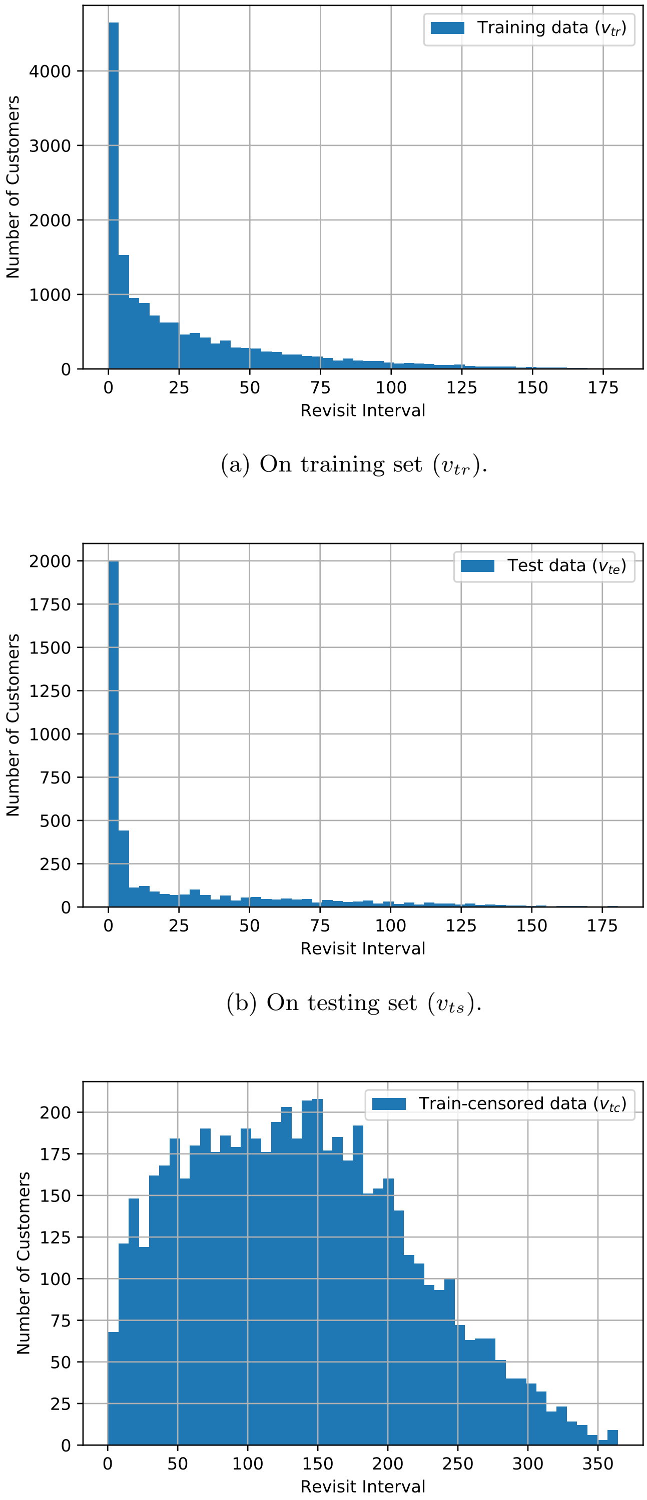
visit_id	wifi_id	date	indices
v68	104	17242	328839;328841;328846;328864
v125	170	17190	107784;107786;...;108296;108328;108341
v212	278	17338	519541;519599
v271	354	17179	59770;59772;...;59897;59904;59913
v272	354	17217	227686;227706;...;228576;228584;228585
v273	354	17224	255448;255450;255455
v335	414	17209	194049;194051;...;194061;194064;194079
v369	464	17282	418485;418487;...;418496;418498;418499
v370	464	17304	456398;456399;456400;456401

(b) The first ten rows of `train_labels.tsv`.

visit_id	revisit_interval	revisit_intention
v68	`nan`	0
v125	`nan`	0
v212	`nan`	0
v271	37.88	1
v272	6.8	1
v273	`nan`	0
v335	`nan`	0
v369	22.15	1
v370	`nan`	0

(c) The first ten rows of `wifi_sessions.csv`.

index	wifi_id	ts	area	dwell_time
105	7183	1483239765	out	2574
106	7183	1483239767	1f	1051
107	7183	1483239767	in	2423
108	7183	1483239767	1f-c	923
109	7183	1483239776	1f-d	913
158	7183	1483240006	1f-e	703
162	3881	1483240015	out	2059
184	3881	1483240174	in	1886
185	3881	1483240174	1f-d	174

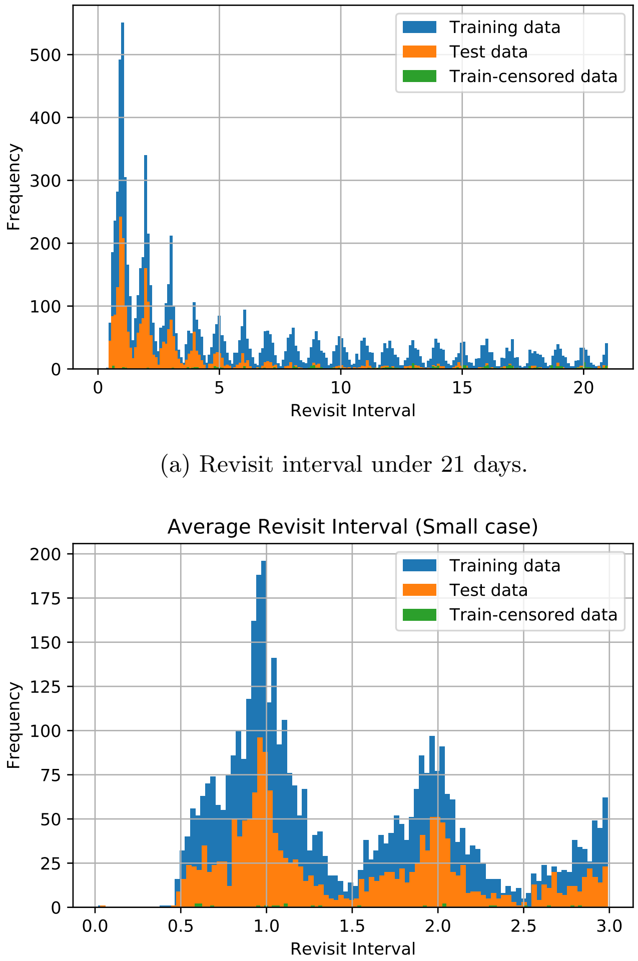


(a) On training set (v_{tr}).

(b) On testing set (v_{ts}).

(c) On train-censored set (v_{tc}).

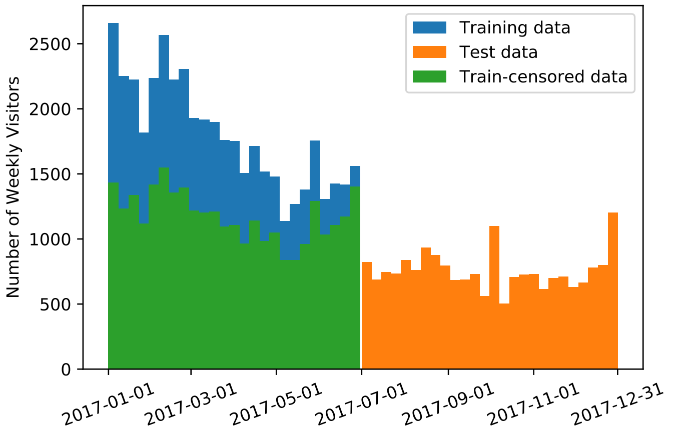
Figure A.1: Histograms of customer revisit interval (Store B, 180 days).



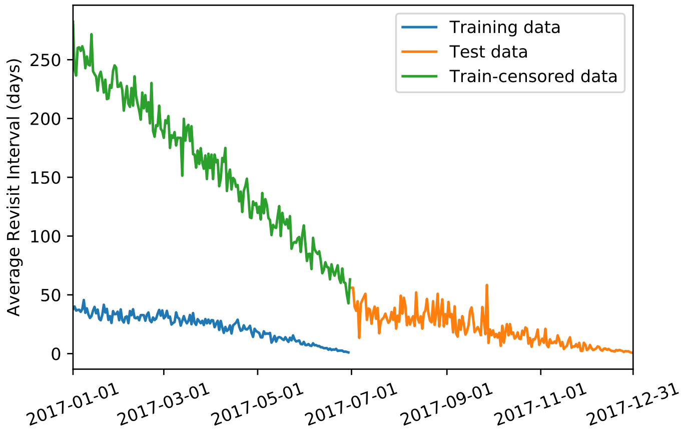
(a) Revisit interval under 21 days.

(b) Revisit interval under 3 days.

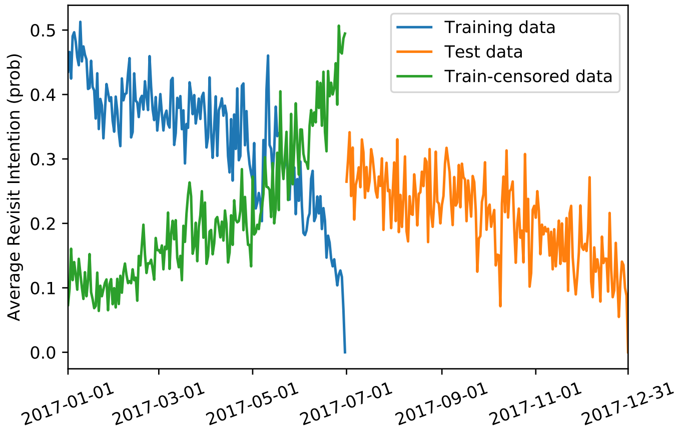
Figure A.2: Daily customer revisit cycle: most of the customers revisit the store having an interval with a multiple of 24 hours (Store B, 180 days).



(a) Number of weekly customers. Note that the number of train-censored data converges to the number of training data as time goes.

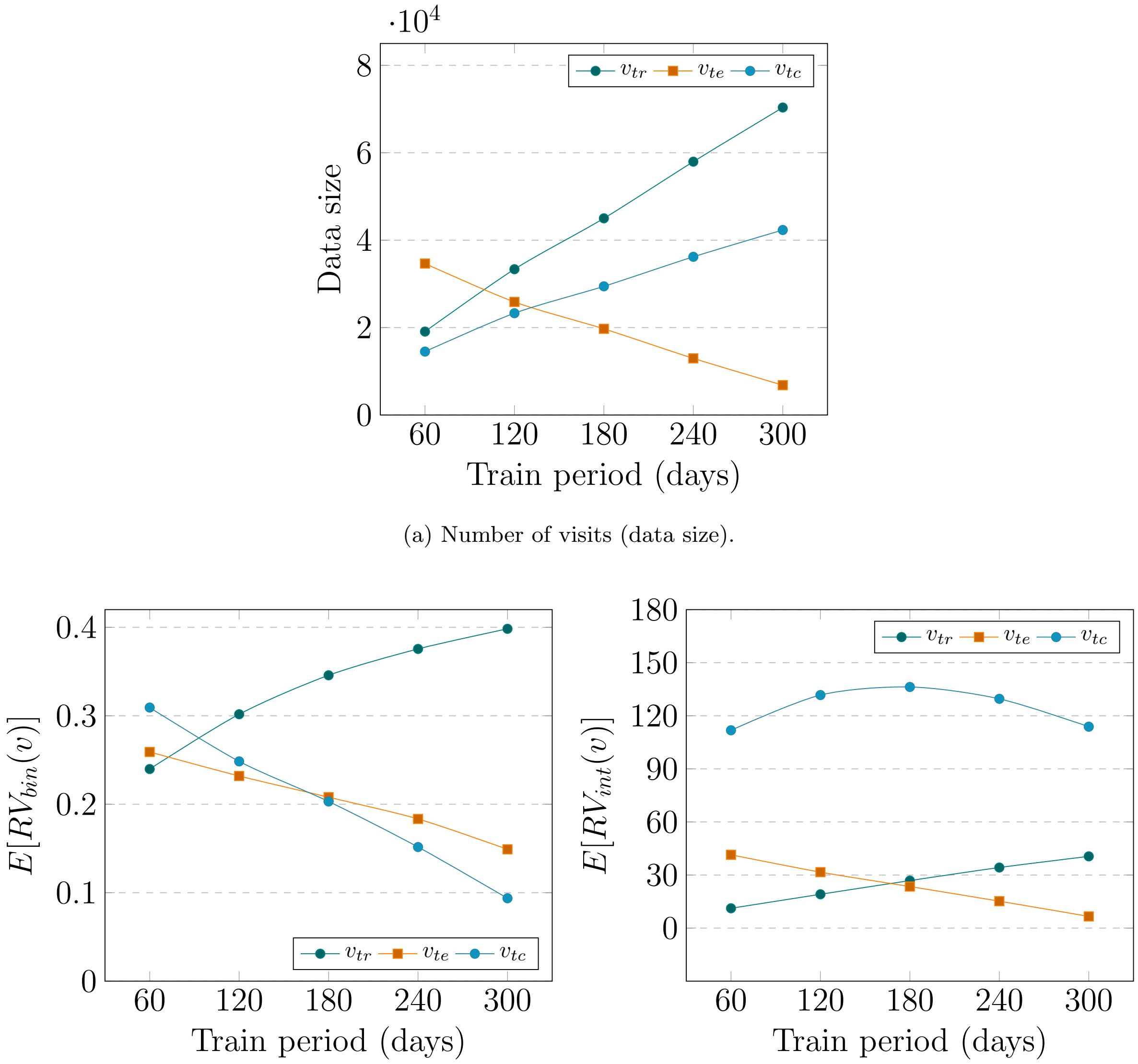


(b) Average revisit interval. It gradually decreases since we are handling longitudinal data. If a customer visits at the end of the data collection period as well as having revisitation before the time ends, his/her revisit interval should be very short.



(c) Average revisit rate. The value gradually decreases for training and testing set. However, the value increases on train-censored set since the customer who visited early but disappeared afterward, is unlikely to revisit during the test period.

Figure A.3: Statistics according to the visit date (Store B, 180 days).



(a) Number of visits (data size).

(b) Average revisit ratio.

(c) Average revisit interval.

Figure A.4: Changes in statistics according to training length (Store B).

Chapter B. Preliminary Neural Network Approaches

This appendix introduces our preliminary neural network approach proceeded with the feature engineering model. The model described in this chapter is not directly related to the final *SurvRev* model described in Chapter 4.

Feature engineering has an inherent limitation that the model performance largely depends on the feature set, and a lot of efforts has been made to design each feature manually. After having a solid feature set, it was difficult to come up with a good idea, and the performance improvement by adding additional features was marginal. Developing more features was similar to participating in endless Kaggle competition without having a public leaderboard. And we knew that feature engineering cannot be perfect since no one entirely knows the elements of a customer revisit.

We expected that model-based learning can be also possible with trajectory data, thus we applied neural network models to directly capture the characteristics of the data causing future revisit. We have tried Convolutional Neural Networks(CNN) and Recurrent Neural Networks(RNN). The CNN is to capture unknown transition patterns from images generated from trajectories. The RNN is to capture unknown sequential patterns from the semantic trajectories themselves. Notwithstanding its intuitions and fancy approaches, we did not achieve higher performance with neural network models than the feature engineering model, thus we omitted this section in our previous submissions. But it is worthwhile to report our efforts and findings on the thesis, hence we introduce our approaches in the following chapters.

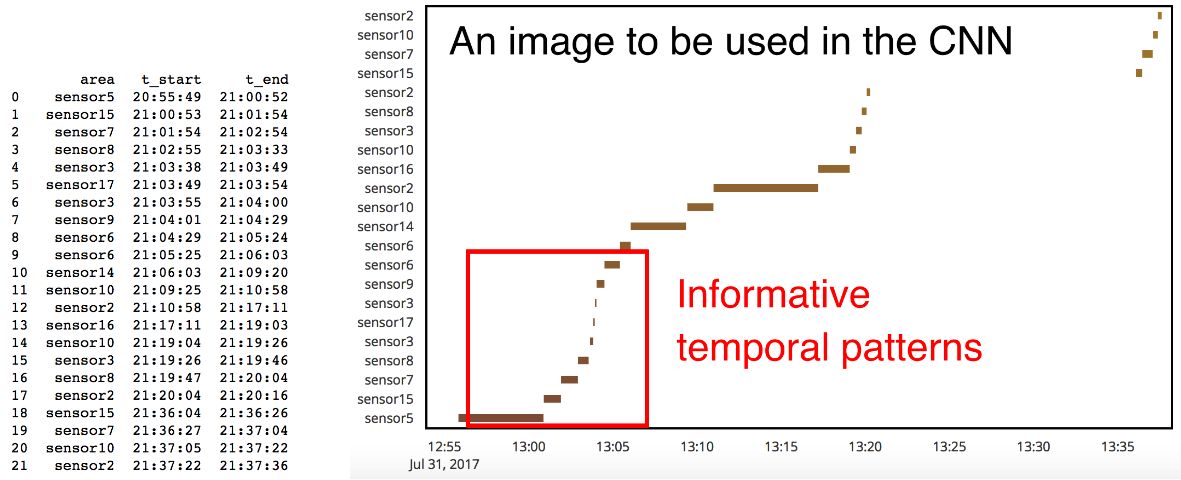
CNN Approach: Considering Trajectory as Image

A Convolutional Neural Network(CNN) is a deep feed-forward neural network which has widely been applied to computer vision. In this thesis, we skip the basics of CNN and directly delve into our application.

How to Generate an Image from Trajectory Data We encode each trajectory to image as an input to CNN. Two options are available to encode a trajectory. The first approach is to generate an image by physical coordinate [20]. The second approach is to generate a Gantt-chart like an image using time and sensor information. In our study, we used the second approach since we want to focus on hidden temporal patterns governing future revisit. For instance, if the pattern of staying in many areas is important, a diagonal of a hidden layer should be highlighted and be learned through the neural network. There is an additional reason that we select the second approach instead of the physical encoding: our data. Indoor semantic trajectories comprise at most 30-40 different areas, which are the most elaborate unit of customer location in the indoor data. Therefore, it concludes to a very small image (30-40 pixels) which does not get any benefit from convolution layers and pooling layers. One may argue that the CNN model can capture the high-level physical areas such as our multi-level semantics utilized for feature engineering. However, we conclude that a 6x6 size image is too small to apply any convolution or pooling.

Figure B.1 describes an idea to generate an image from the trajectory and how CNN works. In Figure B.1(b), the part enclosed by a black box is the image created from the trajectory given in Figure B.1(a). The convolution layer, marked with a red box, compresses the characteristics of the data.

We expect the CNN model can successfully provide informative high-level temporal patterns determines customer revisit.



```
       area   t_start     t_end
0   sensor5  20:55:49  21:00:52
1  sensor15  21:00:53  21:01:54
2   sensor7  21:01:54  21:02:54
3   sensor8  21:02:55  21:03:33
4   sensor3  21:03:38  21:03:49
5  sensor17  21:03:49  21:03:54
6   sensor3  21:03:55  21:04:00
7   sensor9  21:04:01  21:04:29
8   sensor6  21:04:29  21:05:24
9   sensor6  21:05:25  21:06:03
10 sensor14  21:06:03  21:09:20
11 sensor10  21:09:25  21:10:58
12  sensor2  21:10:58  21:17:11
13 sensor16  21:17:11  21:19:03
14 sensor10  21:19:04  21:19:26
15  sensor3  21:19:26  21:19:46
16  sensor8  21:19:47  21:20:04
17  sensor2  21:20:04  21:20:16
18 sensor15  21:36:04  21:36:26
19  sensor7  21:36:27  21:37:04
20 sensor10  21:37:05  21:37:22
21  sensor2  21:37:22  21:37:36
```

(a) An input trajectory.

(b) A Gantt-chart drawn by the trajectory.

Figure B.1: The image generation process from a trajectory.

Image Generating Options We applied different options to generate images from trajectories. The first option is a time normalization option. If the time normalization option is on, each image contains a different time-scale than others. Second, we consider an option to fix the time length of the image. If the time length is fixed to x seconds, we only consider first x seconds to generate an image. Third, we generate an option to fix the output image width. Fourth, we applied five different options to decide the order of each row. Default option is an ordering by the visit time as in Figure B.1(b). In this case, each row does not guarantee sensor uniqueness. Fifth, we add an option to embed extra information at the left or at the bottom side of the image. With this option, corresponding area names or area category of each row can be explicitly added. Last, we enable an option to fix the size of the image. Figure B.2 describes an example image with some explanations. And the value of each option is described as follows:

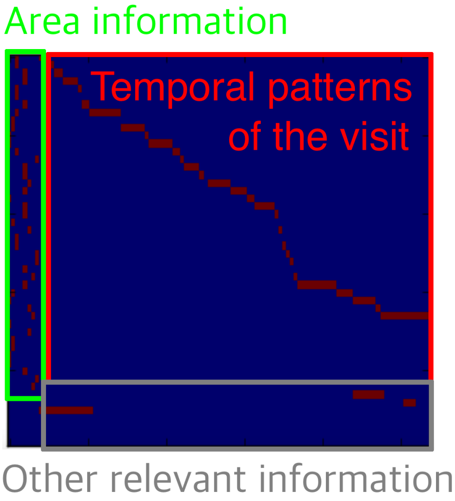


Figure B.2: Description of the generated image.

- Time normalization option: Binary
- Considered time length (second): 300, 600, 1800, 3600, 7200
- Image width: 30, 100
- Row ordering: Session start time (default), category, sensor name, occurrence frequency, dwell time
- Sensor uniqueness: Binary
- Area information (left side): Sensor category, sensor name
- Extra information (bottom side): Aggregated dwell time for each category
- Fix image size: Binary

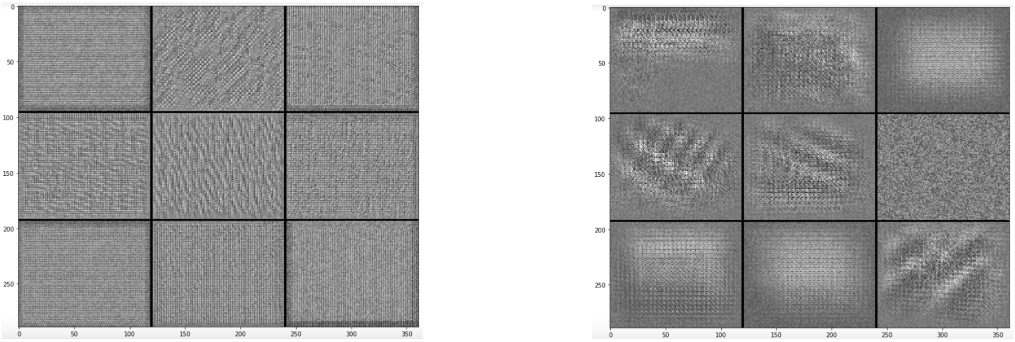
Model Architecture The CNN model consists of two cycles of image abstraction. Layers wrapped in parenthesis were used in the first version of the model, which are utilized optionally. After flattening the output of second convolutions, embeddings of auxiliary inputs (time of visit, customer ID, visit frequency, etc.) are concatenated. The architecture of our CNN model is as follows:

$$\underbrace{Conv - Conv - Drop}_{\text{First cycle of abstraction}} - \underbrace{Conv - Drop}_{\text{Second cycle of abstraction}} - \underbrace{Flatten - Concat}_{\text{Add auxiliary inputs}} - \underbrace{FC - Relu - Drop - FC - Softmax}_{\text{Multi-layer perception to predict}}$$

Model Training We optimize our model by using a binary cross entropy loss with built-in stochastic gradient descent optimizer and Adam optimizer in Keras[16]. And we report an accuracy to evaluate the performance. Hyperparameters used in our experiment are listed here:

- Number of batch size: 128
- Number of epochs: 100
- Kernel size: 9
- Pool size: 2
- Padding size: 1
- Stride size: 2
- Number of kernel for convolution layer: 10
- Dropout probability: 0.05
- Number of neuron in FC layer: 100

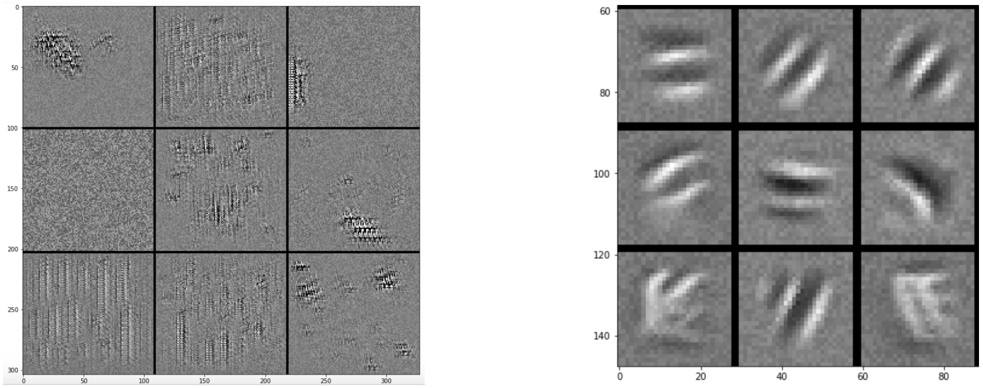
Model Performance and Reason to Stop By the time we worked on the model, we had E_SC and E_GN datasets. The highest prediction accuracy was about 0.58–0.59 depending on the parameter set. The performance was similar to the feature engineering model at that point. The features of that preliminary model were indoor-oriented without considering any store accessibility or sales information. The CNN model does not consider those aspects either. But interpret patterns obtained from the hidden layers were difficult and the obtained patterns were not clear since our model was not as predictive have as the model developed from MNIST classification. Figure B.3 illustrates the feature visualization learned from two convolution layers from our model. In Figure B.4, we note that the patterns observed from the biased sample data with higher prediction accuracy (75 %), or that observed from MNIST dataset (95 % accuracy) are much clearer than that shown in the above figure. Because of these limitations, we stopped modeling with CNN at this level.



(a) Features learned from the first convolution layer.

(b) Features learned from the third convolution layer.

Figure B.3: Features learned from our dataset with 58.4 % accuracy.



(a) Features learned from the third convolution layer of 75 % accuracy biased dataset.

(b) Features learned from the third convolution layer of 95 % accuracy MNIST dataset.

Figure B.4: Features learned from the other datasets with higher accuracy.

RNN Approach: Focusing Sequence of Areas

A recurrent neural network(RNN) is a model where sequentially connected cells represent temporal behavior from time-series input. It has been widely used to model natural language processing or speech recognition where previous data affects the current input state. Customer movement is also a type of time series, so it fits well with the RNN structure. Although the revisit behavior happens after a long period of time compared to each transition inside the store, representing customer revisit as an output of the last hidden cell is reasonable in our understandings. So we applied widely-using unidirectional RNN to feed semantic trajectory of each visit.

Model Architecture Our final model is a dynamic RNN model with LSTM cells. Dynamic RNN allows for variable length, and LSTM is a special kind of RNN capable of learning long-term dependency. To develop a structure, we utilized nn modules provided by PyTorch[91].

Input Features As an input, each cell requires data from each element of a semantic trajectory. For each element, we used following features:

- f_a: Dwell time in that area

- f_b: Area index in a trajectory
- f_c: Area ID

Below two features summarize the whole visit, and we additionally added those to the feature vector to check if there is a performance gain.

- f_d: Length of the whole trajectory
- f_e: Total dwell time of the visit

Model Training Since the computation cost of RNN model is high, we sampled 100,000 visits from O_MD dataset for this experiment. From 100,000 visits, we randomly selected 70 % as a train set and remaining 30 % as a test set. To report the meaningful accuracy, we downsampled each set to be balanced. To minimize the binary cross entropy with logit loss, optimization is done by built-in Adam optimizer with learning rate = 0.005.

- input size = 5
- hidden size = 300
- batch size = 72
- number of layers = 1

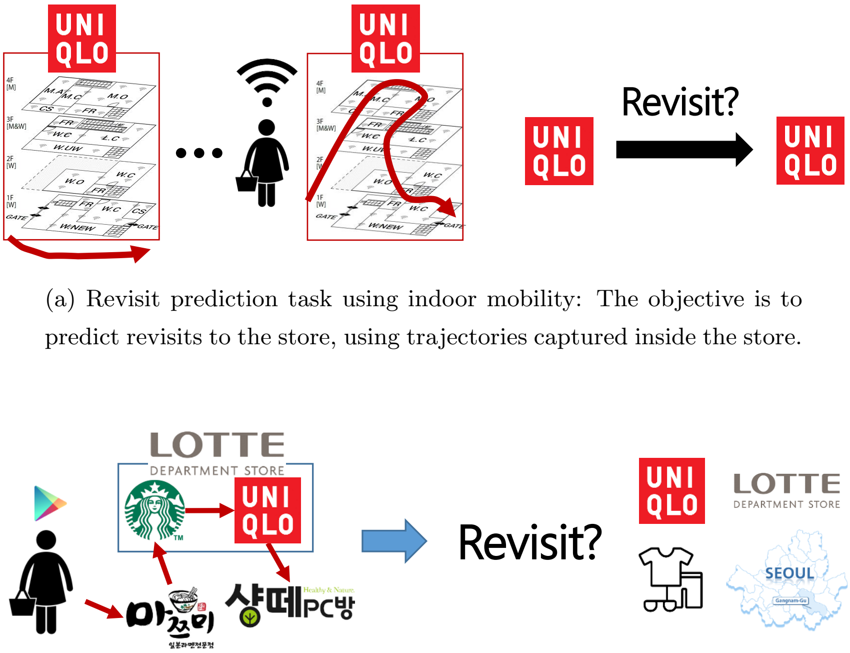
Performance Comparison We generate two RNN model with different feature sets. And compare the performance with classical machine learning models. XGBoost classifier with parameters {max_depth = 4, and learning_rate = 0.1} and RandomForest classifier with parameters {max_depth = 4} are used for comparison. Below are the description and corresponding accuracy of each model.

- LSTM models:
  - Dynamic RNN model using input features $\{f_a, f_b, f_c\} \rightarrow 0.5320$
  - Dynamic RNN model using input features $\{f_a, f_b, f_c, f_d, f_e\} \rightarrow 0.5835$
- Other classifiers using two features: $\{f_d, f_e\}$
  - XGBoost Classifier → 0.6398
  - Random forest Classifier → 0.6383

The performance of the RNN model is much lower than the accuracy of the XGBoost and the random forest. Even if the architecture and hyperparameters of the RNN model were not completely optimized, the performance of the RNN when using the same information was disappointing, so we proceeded to this extent and wrapped up.

Chapter C. Application to Points-Of-Interest Check-In Datasets

How important is the past check-in histories to predict the customer revisit to the store? In the next few pages, I would like to present a brief agenda to understand the customer revisit using the characteristics of inter-store mobility observed in the check-in dataset. Using inter-store mobility, we anticipate finding behavioral patterns which lead to revisits. Those revisit-triggering patterns could be different from store to store, and user to user. Extending the definition of revisit can be also possible. In some cases, it may also be important to predict revisit to other chain stores of the same brand, or revisit to the same shopping district, even if they are not coming back to the exact same store. To find the best model to fit the multi-objective revisit function, we aim to devise a context embedding model to capture information from high-dimensional points-of-interest (POI) data. Those approaches have been utilized in next location prediction but have not been adopted in the revisit prediction task. Figure C.1 illustrates a difference between revisit prediction using indoor mobility and revisit prediction using inter-store mobility.



(a) Revisit prediction task using indoor mobility: The objective is to predict revisits to the store, using trajectories captured inside the store.

(b) Revisit prediction task using inter-store mobility: The objective is to predict revisits to the store or related items to the store, using trajectories captured between multiple store.

Figure C.1: Using inter-store mobility for revisit prediction.

Available Check-in Datasets

Data Description

Two public location-based social network datasets can be used in this work, Foursquare [118] and Gowalla [15]. In addition to those, we used one private check-in dataset collected in the Republic of Korea. Throughout this paper, We call this dataset as Loplat-sample. The number of daily active users and daily check-in amount is roughly a million, and the number of unique POI is 300,000. Currently, we have a sample data for those who visited a multi-purpose convention center COEX at least once in 6 months, with at least 50 POIs. In addition to that, more recent Gowalla dataset[74] and Weeplace dataset are also publicly available[1], and ready to use.

Detailed statistics of the first three check-in dataset is listed on Table C.1.

Table C.1: Check-in data statistics.

Dataset	# Users	# Check-ins	# Locations
Loplat-sample	25,038	5,378,784	229,645
Foursquare	266,909	33,278,683	3,680,126
Gowalla	216,734	12,846,151	1,421,262

Exploratory Data Analysis

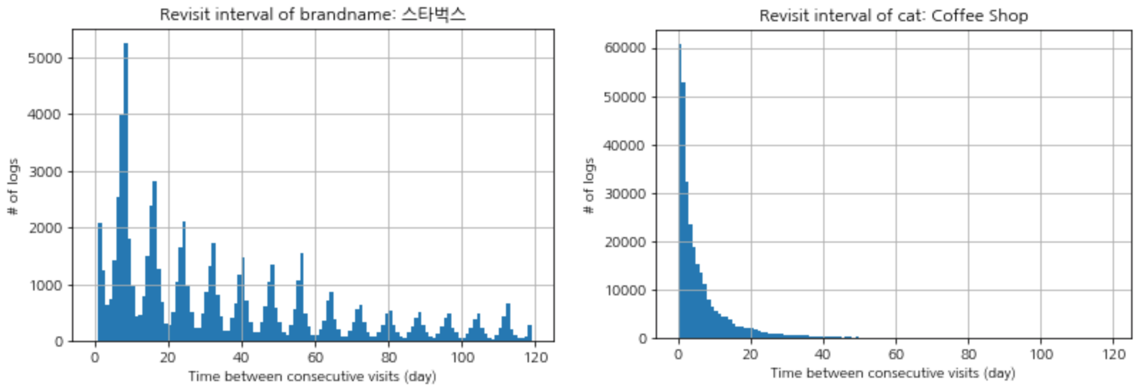
We explored Loplat-sample dataset to identify some revisit trends. As explained in Section 2.4, the revisit interval $RV_{days}(v) = r$ represents the interval between two consecutive visits to the same place, and the revisit intention $RV_{bin}(v)$ is an indicator function of revisit interval. First, we will focus on revisit interval to interpret the periodicity of check-in behavior. For each check-in logs, four types of revisit intervals are calculated. r_p, r_c, r_b, r_d denotes a time taken to revisit to the exact same place, same category, same brand, and same administrative district, respectively. Table C.2 describes an example of r_b.

Table C.2: Examples of revisit intervals: r_b for top-3 coffee franchises.

From/To	# logs	# users	# franchises	$\bar{r}_b$
Cafe S	95,961	18,329	1,184	11 days 11:51:48
Cafe T	23,334	5,979	695	16 days 13:17:56
Cafe E	18,703	5,123	1,349	17 days 07:30:35

Figure C.2 shows an example of revisit intervals with/without periodical patterns. In Figure C.2a, A revisit interval distribution of cafe S follows a power-law with a weekly periodical pattern. Meanwhile, In Figure C.2b a revisit interval distribution of entire coffee franchise follows the power-law, but no periodical pattern is observed.

[1] http://www.yongliu.org/datasets/



(a) Histogram of r_b, where b stands for the franchise cafe brand S. r_b follows a power-law distribution with weekly periodical pattern.

(b) Histogram of r_c, where c stands for the category coffee shop. r_c follows a power-law distribution, but weekly periodical pattern is not observed.

Figure C.2: The existence of periodical patterns.

Bibliography

[1] O. Aalen, "Nonparametric inference for a family of counting processes," *The Annals of Statistics*, vol. 6, no. 4, pp. 701–726, 1978.

[2] L. Adamic and B. Huberman, "The nature of markets in the world wide web," *The Quarterly Journal of Electronic Commerce*, vol. 1, pp. 5–12, 2000.

[3] M. Aitkin and D. Clayton, "The fitting of exponential, weibull and extreme valud distributions to complex censored survival data using GLIM," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 29, no. 2, pp. 156–169, 1980.

[4] A. M. Alaa and M. van der Schaar, "Deep multi-task gaussian processes for survival analysis with competing risks," in *Advances in Neural Information Processing Systems 30.* Curran Associates, Inc., 2017.

[5] J. Alstott, E. Bullmore, and D. Plenz, "Powerlaw: a python package for analysis of heavy-tailed distributions," *PLoS ONE*, vol. 9, no. 1, 2014.

[6] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations*, 2015.

[7] P. C. Besse, B. Guillouet, J.-M. Loubes, and F. Royer, "Destination prediction by trajectory distribution based model," *IEEE Transactions on Intelligent Transportation Systems*, no. 99, pp. 1–12, 2017.

[8] H. Binder and M. Schumacher, "Allowing for mandatory covariates in boosting estimation of sparse high-dimensional survival models," *BMC Bioinformatics*, vol. 9, no. 1, pp. 1–10, 2008.

[9] I. Bou-Hamad, D. Larocque, and H. Ben-Ameur, "A review of survival trees," *Statistics Surveys*, vol. 5, pp. 44–71, 2011.

[10] S. F. Brown, A. J. Branford, , and W. Moran, "On the use of artificial neural networks for the analysis of survival data," *IEEE Transactions on Neural Networks*, vol. 8, no. 5, pp. 1071–1077, 1997.

[11] J. Buckley and I. James, "Linear regression with censored data," *Biometrika*, vol. 66, no. 3, pp. 429–436, 1979.

[12] O. Chapelle, "Modeling delayed feedback in display advertising," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014.

[13] O. Chapelle, E. Manavoglu, and R. Rosales, "Simple and scalable response prediction for display advertising," *ACM Transactions on Information Systems*, vol. 5, no. 61, 2014.

[14] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2016, pp. 785–794.

[15] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: User movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2011.

[16] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[17] D. R. Cox, "Regression models and life-tables," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 34, no. 2, pp. 187–220, 1972.

[18] S. J. Cutler and F. Ederer, "Maximum utilization of the life table method in analyzing survival," *Journal of Chronic Diseases*, vol. 8, no. 6, pp. 699–712, 1958.

[19] V. de Wiele. (2017) Santander product recommendation competition 2nd place winners solution. http://bit.ly/santender-kaggle-blog.

[20] Y. Endo, H. Toda, K. Nishida, and A. Kawanobe, "Deep feature extraction from trajectories for transportation mode estimation," in *The Pacific-Asia Conference on Knowledge Discovery and Data Mining.* Springer, 2016.

[21] Z. Fang, Z. Yang, and Y. Zhang, "Collaborative embedding features and diversified ensemble for e-commerce repeat buyer prediction," Tsinghua University, Tech. Rep., 2015.

[22] M. J. Fard, P. Wang, S. Chawla, and C. K. Reddy, "A bayesian perspective on early stage event prediction in longitudinal data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 12, pp. 3126–3139, 2016.

[23] T. Fernándex, N. Rivera, and Y. W. Teh, "Gaussian processes for survival analysis," in *Advances in Neural Information Processing Systems 29.* Curran Associates, Inc., 2016.

[24] L. D. Fisher and D. Y. Lin, "Time-dependent covariates in the cox proportional-hazards regression model," *Annual Review of Public Health*, vol. 20, pp. 145–157, 1999.

[25] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[26] W. Geng and G. Yang, "Partial correlation between spatial and temporal regularities of human mobility," *Scientific Reports*, vol. 7, no. 6249, 2017.

[27] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory pattern mining," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2007, pp. 330–339.

[28] E. Giunchiglia, A. Nemchenko, and M. van der Schaar, "RNN-SURV: a deep recurrent model for survival analysis," in *International Conference on Artificial Neural Networks.* Springer, 2018, pp. 23–32.

[29] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[30] B. Gregory, "Predicting customer churn: extreme gradient boosting with temporal data," *arXiv:1705.10245*, 2018.

[31] Guggenheim. (2015) Guggenheim app adds feature to highlight artworks near users. http://bit.ly/Guggenheim_App.

[32] A. H. Hald, "Maximum likelihood estimation of the parameters of a normal distribution which is truncated at a known point," *Scandinavian Actuarial Journal*, pp. 187–220, 1949.

[33] G. D. Harrell, M. D. Hutt, and J. C. Anderson, "Path analysis of buyer behavior under conditions of crowding," *Journal of Marketing Research*, vol. 17, no. 1, pp. 45–51, 1980.

[34] A. G. Hawkes, "Spectra of some self-exciting and mutually exciting point processes," *Biometrika*, vol. 58, no. 1, pp. 83–90, 1971.

[35] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[36] T. Hothorn, P. Bühlmann, S. Dudoit, A. Molinaro, and M. Laan, "Survival ensembles," *Biostatistics*, vol. 7, no. 3, pp. 355–373, 2006.

[37] S. K. Hui, E. T. Bradlow, and P. S. Fader, "Testing behavioral hypotheses using an integrated model of grocery store shopping path and purchase behavior," *Journal of Consumer Research*, vol. 36, no. 3, pp. 478–493, 2009.

[38] I. Hwang and Y. Jang, "Process mining to discover shoppers' pathways at a fashion retail store using a wifi-base indoor positioning system," *IEEE Transactions on Automation Science and Engineering*, vol. 14, pp. 1786–1792, 2017.

[39] B. Im, H. Kim, and S. Nam, "The relation analysis among the store attributes, relationship quality and revisit intention in outdoor sport store," *The Korean Journal of Physical Education*, vol. 52, no. 6, pp. 335–345, 2013.

[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015, pp. 448–456.

[41] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, and M. S. Lauer, "Random survival forests," *The Annals of Applied Statistics*, vol. 4, no. 1, pp. 44–71, 2011.

[42] H. Ishwaran, U. B. Kogalur, X. Chen, and A. J. Minn, "Random survival forests for highdimensional data," *Statistical Analysis and Applied Statistics*, vol. 4, no. 1, pp. 115–132, 2011.

[43] H. Jing and A. J. Smola, "Neural survival recommender," in *The 10th ACM International Conference on Web Search and Data Mining.* ACM, 2017, pp. 495–503.

[44] S. Jung, C. Lim, and S. Yoon, "Study on selecting process of visitor's movements in exhibition space," *Journal of the Architectural Institute of Korea Planning & Design*, vol. 27, no. 12, pp. 53–62, 2011.

[45] Kaggle. (2015) Taxi trajectory winners' interview: 1st place, team? http://bit.ly/kaggle_taxi_interview_1st_nn.

[46] Kaggle. (2015) Taxi trajectory winners' interview: 3rd place, bluetaxi. http://bit.ly/kaggle_taxi_interview_3rd.

[47] E. L. Kaplan and P. Meier, "Nonparametric estimation from incomplete observations," *Journal of the American Statistical Association*, vol. 35, pp. 457–481, 1958.

[48] J. L. Katzman, U. Shaham, A. Cloninger, J. Bates, T. Jiang, and Y. Kluger, "DeepSurv: Personalized treatment recommender system using a cox proportional hazards deep neural network," *BMC Medical Research Methodology*, 2018.

[49] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, pp. 3146–3154.

[50] F. M. Khan and V. B. Zubek, "Support vector regression for censored data (SVRc): a novel tool for survival analysis," in *IEEE International Conference on Data Mining*. IEEE, 2008, pp. 863–868.

[51] S. Kim and J.-G. Lee, "Predicting customer's revisit intention using indoor movements in stores by Wi-Fi monitoring," in *Korea Computer Congress Winter Conference*. Korean Institute of Information Scientists and Engineers, 2016, pp. 374–376.

[52] S. Kim and J.-G. Lee, "Utilizing in-store sensors for revisit prediction," in *IEEE International Conference on Data Mining*. IEEE, 2018, pp. 217–226.

[53] S. Kim and J.-G. Lee, "A systemic framework of predicting customer revisit with in-store sensors," in *Knowledge and Information Systems (To Appear)*. Springer, 2019.

[54] T. Kim, M. Chu, O. Brdiczka, and J. Begole, "Predicting shoppers' interest from social interactions using sociometric sensors," in *CHI'09 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2009, pp. 4513–4518.

[55] Y. Kim, "Convolutional neural networks for sentence classification," in *Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1746–1751.

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[57] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine learning*, 2014.

[58] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[59] C. Lee, W. R. Zame, J. Yoon, and M. van der Schaar, "DeepHit: A deep learning approach to survival analysis with competing risks," in *The 32nd AAAI Conference on Artificial Intelligence*. AAAI Press, 2018.

[60] E. T. Lee and J. W. Wang, *Statistical methods for survival data analysis*. Wiley, 2003.

[61] J.-G. Lee, J. Han, and X. Li, "Mining discriminative patterns for classifying trajectories on road networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 5, pp. 713–726, 2011.

[62] S. Lee, S. Bae, and S. Cho, "The effect of shopping emotions in a golf apparel store on repurchase intention," *Journal of Sport and Leisure Studies*, vol. 48, no. 1, pp. 361–371, 2011.

[63] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017.

[64] X. Li, G. Cong, X.-L. Li, T.-A. N. Pham, and S. Krishnaswamy, "Rank-geofm: A ranking based geographical factorization method for point of interest recommendation," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015.

[65] Y. Li, K. S. Xu, and C. K. Reddy, "Regularized parametric regression for high-dimensional survival analysis," in *Proceedings of the 2016 SIAM International Conference on Data Mining*, 2016.

[66] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "Geomf: Joint geographical modeling and matrix factorization for point-of-interest recommendation," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014.

[67] C. Lim, H. Park, and S. Yoon, "A study of an exhibitions space analysis according to visitor's cognition," *Journal of the Architectural Institute of Korea Planning & Design*, vol. 29, no. 8, pp. 69–78, 2013.

[68] C. Lim and M. Park, "A study on the relationship between the spatial configuration and visitor's movement in museum(i, ii)," *Journal of the Architectural Institute of Korea Planning & Design*, vol. 22, no. 10, pp. 167–174, 2006.

[69] C. Lim and M. Park, "A study on visitor's flow in the exhibition area," *Journal of the Architectural Institute of Korea Planning & Design*, vol. 22, no. 2, 2006.

[70] C. Lim and S. Yoon, "Development of visual perception effects model for exhibition space," *Journal of the Architectural Institute of Korea Planning & Design*, vol. 26, no. 5, pp. 131–138, 2010.

[71] D. Lim and B. Kim, "Chinese tourist's intentions to revisit offline stores: Focusing on store attributes and mobile payment attributes," *The Journal of Internet Electronic Commerce Research*, vol. 17, no. 1, pp. 171–195, 2017.

[72] G. Liu, T. T. Nguyen, G. Zhao, W. Zha, J. Yang, J. Cao, M. Wu, P. Zhao, and W. Chen, "Repeat buyer prediction for E-Commerce," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2016, pp. 155–164.

[73] Y. Liu, T.-A. N. Pham, G. Cong, and Q. Yuan, "An experimental evaluation of point-of-interest recommendation in location-based social networks," in *Proceedings of the Very Large Data Bases Endowment*, 2017.

[74] Y. Liu, W. Wei, A. Sun, and C. Miao, "Exploiting geographical neighborhood characteristics for location recommendation," in *The 23rd ACM International Conference on Information and Knowledge Management.* ACM, 2014.

[75] Y. Liu, W. Wei, A. Sun, and C. Miao, "Exploiting geographical neighborhood characteristics for location recommendation," in *The 23rd ACM International Conference on Information and Knowledge Management*, 2014.

[76] X. Lu, E. Wetter, N. Bharti, A. J. Tatem, and L. Bengtsson, "Approaching the limit of predictability in human mobility," *Scientific Reports*, vol. 3, no. 2923, 2013.

[77] M. Luck, T. Sylvain, H. Cardinal, A. Lodi, and Y. Bengio, "Deep learning for patient-specific kidney graft survival analysis," *arXiv preprint arXiv:1705.10245*, 2017.

[78] J. Lv, Q. Li, Q. Sun, and X. Wang, "T-CONV: A convolutional neural network for multi-scale taxi trajectory prediction," in *Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing.* IEEE, 2018, pp. 82–89.

[79] D. Madigan, "Introduction to survival analysis (lecture note of biost 515, colombia university)," http://www.stat.columbia.edu/~madigan/W2025/notes/survival.pdf, 2004.

[80] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown, "A study of MAC address randomization in mobile devices and when it fails," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 4, pp. 365–383, 2017.

[81] W. Mathew, R. Raposo, and B. Martins, "Predicting future locations with hidden markov models," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing.* ACM, 2012, pp. 911–918.

[82] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, "WhereNext: A location predictor on trajectory pattern mining," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2012, pp. 637–646.

[83] W. Nelson, "Theory and applications of hazard plotting for censored failure data," *Technometrics*, vol. 14, no. 4, pp. 945—-966, 1972.

[84] OpenSignal, Inc, "Global state of mobile networks (August 2016)," OpenSignal, Inc, Tech. Rep., 2016.

[85] S. Park, S. Jung, and C. Lim, "A study on the pedestrian path choice in clothing outlets," *Korean Institute of Interior Design Journal*, vol. 28, pp. 140–148, 2001.

[86] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[87] V. Pejovic and M. Musolesi, "Anticipatory mobile computing: A survey of the state of the art and research challenges," *ACM Computing Surveys*, vol. 47, no. 3, 2015.

[88] D. Peppers and M. Rogers, *Managing customer experience and relationships.* Wiley, 2016.

[89] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features support," in *Advances in Neural Information Processing Systems 31.* Curran Associates, Inc., 2018, pp. 6639–6649.

[90] PYMNTS. (2017) Geotracking gives brick-and-mortar a leg up on ecommerce. http://bit.ly/pymnts_geofencing.

[91] PyTorch, "Pytorch recurrent layers," https://pytorch.org/docs/stable/nn.html#recurrent-layers, 2019.

[92] A. Raftery, D. Madigan, and C. T. Volinsky, “Accounting for model uncertainty in survival analysis improves predictive performance,” *Bayesian Statistics*, vol. 5, pp. 323–349, 1995.

[93] V. C. Raykar, H. Steck, B. Krishnapuram, C. Dehing-Oberije, and P. Lambin, “On ranking in survival analysis: Bounds on the concordance index,” in *Advances in Neural Information Processing Systems 20*. Curran Associates, Inc., 2007.

[94] R. Řehůřek and P. Sojka, “Software framework for topic modelling with large corpora,” in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks held in conjunction with International Conference on Language Resources and Evaluation*. ELRA, 2010, pp. 45–50.

[95] K. Ren, J. Qin, L. Zheng, Z. Yang, W. Zhang, L. Qiu, and Y. Yu, “Deep recurrent survival analysis,” in *The 33rd AAAI Conference on Artificial Intelligence*. AAAI Press, 2019.

[96] Y. Ren, M. Tomko, F. D. Salim, K. Ong, and M. Sanderson, “Analyzing web behavior in indoor retail spaces,” *Journal of the Association for Information Science and Technology*, vol. 68, no. 1, pp. 62–76, 2017.

[97] S. M. Ross, *Stochastic processes (Second edition)*. Wiley, 1996.

[98] P. Sapiezynski, A. Stopczynski, R. Gatej, and S. Lehmann, “Tracking human mobility using WiFi signals,” *PLoS ONE*, 2015.

[99] M. H. Sarshar, “Analyzing large scale wi-fi mobility data using supervised and unsupervised learning techniques,” Master’s thesis, Dalhousie University, Halifax, Canada, 2016.

[100] S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. T. Campbell, “Nextplace: A spatio-temporal prediction framework for pervasive systems,” in *Proceedings of the 9th International Conference on Pervasive Computing*. Springer, 2011, pp. 152–169.

[101] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Transactions on Signal Processing*, pp. 2673–2681, 1997.

[102] A. Sheth, S. Seshan, and D. Wetherall, “Geo-fencing: Confining Wi-Fi coverage to physical boundaries,” in *Proceedings of the 7th International Conference on Pervasive Computing*, 2009, pp. 274–290.

[103] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, “Limits of predictability in human mobility,” *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[104] R. S. Stanković and B. J. Falkowskib, “The Haar wavelet transform: its status and achievements,” *Computers & Electrical Engineering*, vol. 29, no. 1, pp. 25–44, 2003.

[105] R. E. Stevens, I. C. Sherman, and L. Curry, “The behavior of the museum visitor,” *Publications of the American Association of Museums*, vol. 1, no. 5, pp. 1–70, 1928.

[106] A. Syaekhoni, C. Lee, and Y. Kwon, “Analyzing customer behavior from shopping path data using operation edit distance,” *Applied Intelligence*, vol. 48, pp. 1912–1932, 2018.

[107] R. Tibshirani, “The lasso method for variable selection in the cox model,” *Statistics in Medicine*, vol. 16, pp. 385–395, 1997.

[108] J. Tobin, “Estimation of relationships for limited dependent variables,” *Econometrica*, vol. 26, no. 1, pp. 24–36, 1958.

[109] M. Tomko, Y. Ren, K. Ong, F. Salim, and M. Sanderson, “Large-scale indoor movement analysis: the data, context and analytical challenges,” in *Proceedings of Analysis of Movement Data, GIScience 2014 Workshop*, 2014.

[110] S. Um, K. Chon, and Y. Ro, “Antecedents of revisit intention,” *Annals of Tourism Research*, vol. 33, no. 4, pp. 1141–1158, 2006.

[111] P. Wang, Y. Li, and C. K. Reddy, “Machine learning for survival analysis: A survey,” *ACM Computing Surveys*, vol. 1, no. 1, 2018.

[112] L.-J. Wei, “The accelerated failure time model: A useful alternative to the cox regression model in survival analysis,” *Statistics in Medicine*, vol. 11, pp. 1871–1879, 1992.

[113] D. H. Wolpert, “Stacked generalization,” *Neural Networks*, vol. 5, pp. 241–259, 1992.

[114] K. Yada, “String analysis technique for shopping path in a supermarket,” *Journal of Intelligent Information Systems*, vol. 36, no. 3, pp. 385–402, 2011.

[115] S. S. Yalowitz and K. Bronnenkant, “Timing and tracking: Unlocking visitor behavior,” *Visitor Studies*, vol. 12, no. 1, pp. 47–64, 2009.

[116] X. Yan, J. Wang, and M. Chau, “Customer revisit intention to restaurants: Evidence from online reviews,” *Information Systems Frontiers*, vol. 17, pp. 645–657, 2015.

[117] Z. Yan, D. Chakraborty, C. Parent, S. Spaccapietra, and K. Aberer, “Semantic trajectories: Mobility data computation and annotation,” *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 3, pp. 1–38, 2013.

[118] D. Yang, D. Zhang, and B. Qu, “Participatory cultural mapping based on collective behavior data in location-based social networks,” *ACM Transactions on Intelligent Systems and Technology*, 2016.

[119] G. Yang, Y. Cai, and C. K. Reddy, “Recurrent spatio-temporal point process for check-in time prediction,” in *The 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 2203–2211.

[120] G. Yang, Y. Cai, and C. K. Reddy, “Spatio-temporal check-in time prediction with recurrent neural network based survival analysis,” in *The 27th International Joint Conference on Artificial Intelligence*. AAAI Press, 2018, pp. 2976–2983.

[121] J. Yoon, W. R. Zame, A. Banerjee, M. Cadeiras, A. M. Alaa, and M. van der Schaar, “Personalized survival predictions via trees of predictors: An application to cardiac transplantation,” *PLoS One*, 2018.

[122] Y. Yoshimura, A. Krebs, and C. Ratti, “Noninvasive bluetooth monitoring of visitors’ length of stay at the louvre,” *IEEE Pervasive Computing*, vol. 16, no. 2, pp. 26–34, 2017.

[123] S. Yousefi, F. Amrollahi, M. Amgad, C. Dong, J. E. Lewis, C. Song, D. A. Gutman, S. H. Halani, J. E. V. Vega, D. J. Brat, and L. A. D. Cooper, “Predicting clinical outcomes from large scale cancer genomic profiles with deep survival models,” *Scientific Reports*, vol. 7, no. 117077, 2017.

[124] Y. Yu and X. Chen, "A survey of point-of-interest recommendation in location-based social networks," in *The 29th AAAI Conference on Artificial Intelligence Workshop*, 2015.

[125] J. Zhang, S. Wang, L. Chen, G. Guo, R. Chen, and A. Vanasse, "Time-dependent survival neural network for remaining useful life prediction," in *The Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2019.

[126] T. Zhou, H. Qian, Z. Shen, C. Zhang, C. Wang, S. Liu, and W. Ou, "JUMP: A joint predictor for user click and dwell time," in *The 27th International Joint Conference on Artificial Intelligence*. AAAI Press, 2018, pp. 3704–3710.

[127] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, pp. 301–320, 2005.

[128] ZOYI, "Wi-Fi usage survey," http://bit.ly/wifi_survey_file, 2015.