A Mathematical Model of the Causal Effects of Machine Learning Algorithms on User Behavior

by

İBRAHİM DELİBALTA

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Design, Technology and Society

Koç University, Spring 2018

FORM NO: 8

KOÇ ÜNİVERSİTESİ

SOSYAL BİLİMLERİ ENSTİTÜSÜ

DOKTORA TEZ SAVUNMA TUTAĞI

Öğrencinin Adı ve Soyadı : İbrahim Delibalta

Öğrenci No. : 0036277

Anabilim Dalı : İletişim Teknolojileri (İletişim ve Tasarım Anabilim Dalı) (Tasarım Teknoloji ve Toplum Bilim Dalı)

Doktora Yeterlik Sınavına Girdiği Tarih : 29 / 03 / 2016

Doktora Tez Önerisi Kabul Tarihi : 29 / 03 / 2016

Tezin Başlığı (Lütfen İngilizce olarak yazınız) : A Mathematical Model of the Causal Effects of Machine Learning Algorithms on User Behavior

Yukarıda Adı ve Soyadı verilen öğrencinin Doktora Tez Savunması 26/02/2018 tarihinde yapılmış ve öğrencinin tezi Jüri tarafından ☒ **oybirliği** / ☐ **oyçokluğu** ile;

☒ başarılı bulunmuştur

☐ başarısız bulunmuştur

☐ düzeltmeler için ..... ek süre tanınmıştır

27/02/2018
Tarih

Prof. Dr. Oğuzhan Özcan
Program Koordinatörü

İmza

Üyeler	Ünvanı, Adı ve Soyadı	Enstitü Anabilim Dalı/Kurumu	İmza
1. Üye Tez Danışmanı	Doç. Dr. Lemi Baruh	Koç Üniversitesi	
2. Üye (Tez İzleme Komitesi)	Doç. Dr. Asım Evren Yantaç	Koç Üniversitesi	
3. Üye (Tez İzleme Komitesi)	Doç. Dr. G. Tarcan Kumkale	Kadir Has Üniversitesi	
4. Üye (Okul Dışından)	Yrd. Doç. Dr. Ali Cengiz Beğen	Özyeğin Üniversitesi	
5. Üye	Prof. Dr. Hakan Ürey	Koç Üniversitesi	
2. Tez Danışmanı (Mevcut ise)			

Yukarıda adı geçen öğrenci Tezini başarı ile savunmuş ve Doktora derecesi almaya hak kazanmıştır. İlgili Enstitü Yönetim Kurulu Ka aşağıda belirtilmiştir.

Enstitü Yönetim Kurulu Karar No:

27/02/2018
Tarih

Prof. Dr. Zeynep Aycan
Enstitü Direktörü

İmza

Thesis Committee:

Assoc. Prof. Lemi Baruh (advisor)

Assoc. Prof. Asım Evren Yantaç

Assoc. Prof. G. Tarcan Kumkale

Asst. Prof. Ali Cengiz Beğen

Prof. Hakan Ürey

ABSTRACT

Today machine learning algorithms are an integral part of many high-tech products and services. They are extensively used in the decision-making processes in virtually all the products and services of well-known information companies such as Google, Facebook and Yahoo. Typically, users' online history (e.g. web pages visited, social media messages and products viewed/purchased) is used by machine learning algorithms to infer age, gender, location, income level, and other demographics. Then, this information, along with the current context, is used by recommendation engines to predict the `rating' or `preference' that a user would give to items such as to movies, books, research articles, search queries, social tags and products in general. Targeted advertising systems use similar data to serve ads that a user is most likely to notice and take action. As the number of users of these companies increase, the effects of these algorithms on the users are getting more complex and significantly more important.

To this end, this thesis models the effects of the machine learning algorithms on user preferences. Specifically, this thesis (a) proposes a mathematical model of the potential effects of machine learning algorithms on users' preferences, and (b) utilizes the insights from this model for proposing a system design framework for altering user preferences in a desired manner. A state-space model is introduced, where the user's internal state of preferences is represented using a first order Markov chain. An additional observation model stage is added to represent observable user actions based on the latent (unobservable) internal state. The complete system is designed to estimate the latent state of the users from observations.

Estimation process of the latent internal state is extensively studied in the machine learning literature such as in the regression, classification and recommendation frameworks. However, as novel contributions of this work, the following goals are accomplished:

- We model the effects of machine learning algorithms such as recommendation engines on users through a causal feedback loop. We introduce a complete state-space formulation modeling: (1) evolution of preferences vectors, (2) observations generated by users, and (3) causal feedback effects of the actions of algorithms on the system. All these parameters are jointly optimized through an Extended Kalman Filtering framework.

- We introduce algorithms to estimate the unknown system parameters with and without feedback. In both cases, all the parameters are estimated jointly. We emphasize that we provide a complete set of equations covering all the possible scenarios.
- To tune the preferences of users towards a desired sequence, we also introduce a linear regression algorithm and introduce an optimization framework using stochastic gradient descent algorithm. Unlike all the previous work that only use the observations to predict certain desired quantities, as the first time in the literature, we specifically design outputs to "update" the internal state of the system in a desired manner.

ÖZET

Günümüzde makine öğrenmesi algoritmaları teknolojik ürün ve servislerin değismez bir parcası haline gelmiştir. Google, Facebook ve Yahoo gibi bilgi şirketlerinin ürünlerindeki karar mekanizmalarında yaygın halde kullanılmaktadırlar. Bu algoritmalar, kullanıcının online verilerini (ziyaret ettiği web sayfaları, sosyal medya mesajları, baktiği ya da satınaldığı ürünler gibi) kullanarak yaş, cinsiyet, lokasyon, gelir seviyesi ve diğer demografik bilgileri ile ilgili çıkarımlar yapar. Kullanıcıların film, kitap ve diğer ürünlere vereceği potansiyel notlar veya bu ürünlerle ilgili tercihleri tavsiye motorları tarafından bu bilgiler kullanılarak tahmin edilir. Hedefli reklam sistemleri de benzer verileri baz alarak kullanıcının ilgisini çekecek ve tıklama ihtimali yüksek reklamlari seçip gosterirler. Bu şirketlere ait ürünlerin kullanıcı sayıları arttıkça algoritmaların kullanıcılar üzerindeki etkileri daha karmaşık ve önemli hale gelmektedir.

Bu tezde makine öğrenmesi algoritmalarının kullanıcı tercihleri üzerindeki etkileri modellenmiştir. Tezde (a) kullanıcı davranış ve tercihlerinin matematiksel bir model ve (b) bu model kullanılarak kullanıcı tercihlerini istenen şekilde değiştirebilecek bir çerceve sistem önerilmektedir. Kullanıcının gözlemlenemeyen iç durumunu birinci derece Markov zinciri kullanarak modelleyen bir durum-uzay modeli geliştirilmistir. Ayrica bu iç duruma bağlı olarak ortaya cıkan gözlemlenebilir kullanıcı hareketleri icin de bir gözlem modeli eklenmiştir. Sistemin tamamı gözlemlerden iç durumu tahmin etmek üzere tasarlanmıştır.

Regresyon, sınıflandirma ve tavsiye motorlari gibi makine öğrenme literatüründe iç durumun tahminlenmesi geniş bir şekilde çalışılmıs olsa da bu tez çalışmasının özgün kazanimları şu şekildedir:

- Makina öğrenme algoritmalarının kullanıcılar üzerindeki etkileri sebep-sonuc geri bildirim döngüsü ile modellenirken durum-uzay formülasyonu şu sekilde formüle edilmistir: (1) tercih vektörlerinin evrimi, (2) kullanıcı gözlemleri, ve (3) sebep-sonuç geri bildirim döngüsü ile kullanıcı aksiyonlarının sistem üzerindeki ektisi. Bütün sistem parametreleri ortak olarak Genişletilmis Kalman Süzgeçleri ile optimize edilmiştir.
- Bilinmeyen sistem parametrelerini tahmin etmek üzere geri bildirimli ve geri bildirimsiz durumlar icin algoritmalar geliştirilmistir.
- Kullanıcı tercihlerini istenen bir sekilde ayarlamak icin doğrusal regresyon ve stokastik gradyan inişi çerçevesinde algoritmalar geliştirilmistir. Çıktıların sistemin iç durumunu istenen şekilde ayarlaması literatürde ilk kez yapılmıştır.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor Dr. Lemi Baruh for his support and guidance. His vision has helped shape this thesis. I would like to thank my thesis committee members Dr. Evren Yantaç and Dr. Tarcan Kumkale for their insightful comments and encouragement. I would also like to thank Dr. Hakan Ürey and Dr. Ali Beğen for committing their precious time to be in my thesis committee.

My wife, Betül, and children, Kerem and Sena, have been very supportive and understanding during all the times I had to spend away from them for studying and attending classes. I would also like to thank my father-in-law, Alaattin Büyükkaya, for his great words of wisdom and encouragement.

CONTENTS

1 INTRODUCTION 8
1.1 DATASETS 12
2 FROM RAW DATA TO INSIGHT 15
2.1 DATA PROCESSING 19
2.1.1 Pre-processing 19
2.1.2 Encoding and Representation 20
2.1.3 Processing and Applications 24
2.2 PUBLICATIONS 27
3 ANOMALY DETECTION 28
3.1 NATURE OF DETECTION PROBLEMS AND CHALLENGES 29
3.2 ALGORITHMS AND APPROACHES 30
3.3 APPLICATIONS 32
3.4 PUBLICATIONS 36
4 ACTIONS TO STEER PREFERENCES 37
4.1 MOTIVATION 38
4.2 BACKGROUND 41
4.3 PUBLICATIONS 43
5 CONCLUSIONS AND FUTURE WORK 44
6 REFERENCES 53
APPENDIX 57

LIST OF FIGURES

FIGURE 1 THE DIGITAL FEEDBACK LOOP 10

FIGURE 2 THE STATE-SPACE MODEL WITH FEEDBACK 12

FIGURE 3 AN EXAMPLE HIDDEN MARKOV MODEL 42

1 INTRODUCTION

Recent innovations in communication technologies, coupled with the increased use of interactive media and smartphones, greatly enhanced the ability of companies and governments to gather and process an enormous amount of information on individual users. Today, information can be collected from many sources, including, but not limited to, content users share on social networks and blogs, behavioral data collected from online services, intelligent device activities, and security camera recordings. Efficient and effective processing of this “big data" have the potential to significantly improve the quality of many real life applications or products since this enormous amount of data can be used to accurately profile and, then, target particular users. This unprecedented quantity of information has already attracted the attention of knowledge conglomerates such as Google, Yahoo and Microsoft. These companies use this information not only to steadily increase quality of their services by designing highly

sophisticated machine learning algorithms, but also to influence the behavior of consumers and users due to their extensive access into our daily lives (Stoicescu 2015). To this end, we seek to mathematically model the effects of machine learning algorithms on users, particularly preferences of the users, and then use this mathematical model to tune the overall system in order to change the behaviors/preferences of the users (if possible) in a desired manner.

Unlike conventional applications of machine learning algorithms whereby available data is used to make inferences about users and predict, for example, the most suitable movie for a particular user, new generation of machine learning systems employed by these enormously large companies have the capability, ability and potential to change the underlying problem framework, i.e., the user itself, by design (Brodersen et. al. 2015; Zarsky 2004). As an example, consider “Yemek Sepeti", which provides services such as food delivery, user ratings, restaurant search and recommendation. Based on the user history, Yemek Sepeti tries to provide the most appropriate content and the well-tuned offers to its users. In a large scale, this comparably powerful medium can be used not only to provide targeted content, but can also be used to change user behavior, inclinations or preferences, e.g., unhealthy eating habits of a consumer can potentially be changed by suggesting more healthier foods with high ratings similar to usual preferred diet of the user. Hence, this abundance of new sources of information and previously unimaginable ways of access to consumers' data have the potential to substantially change the underlying applications and classical machine learning approaches by incorporating the effects of the algorithm on the intended user in the design.

We argue and mathematically model that the actions of the machine learning algorithms, such as recommending targeted healthier food to users, can affect the preferences and behaviors of users by changing their internal state, however, there are certain attributes of the internal state that cannot be altered. In the next chapters and as summarized in the following, we show that we can represent both effects using a statistical approach, which depends on both immutable characteristics of the user (incorporated as a side information and a state equation) as well as on inputs (or actions) from the environment (or the machine learning algorithm) incorporated as a deriving term.

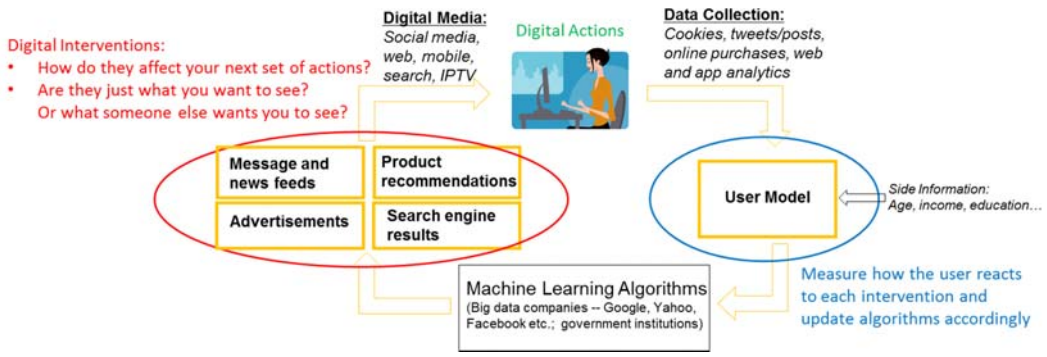


Figure 1 The Digital Feedback Loop

The machine learning algorithms produce digital interventions or triggers to users in the form of online advertisements, social media and news feeds, product recommendations and search engine results. The user acts on these interventions by reading a recommended article, clicking on a search result or an online advertisement. These

actions leave digital footprints for the big data companies to collect, process and feed their machine learning algorithms. And this completes the digital feedback loop as shown in Figure 1. As the loop is repeated, the algorithms get smarter and more effective in knowing how the user behaves and reacts to interventions. Thus, the algorithms not only produce more effective interventions but also learn to steer the user behavior in an intended manner.

We model this phenomenon using a state-space model as shown in Figure 2. Next, we provide a high level description of our model and the structure of the thesis. In the figure, the current preferences of the user are represented by $\boldsymbol{p}_t$ and the next state of the preferences by $\boldsymbol{p}_{t+1}$. These preferences are latent and can only be indirectly observed by the user actions, modeled as $\boldsymbol{x}_t$. The user actions are a collection of Facebook shares, comments, status updates, spending patterns and all other digital footprint. The model assumes that the user actions are a function of his/her preferences $\boldsymbol{p}_t$ and the side information $\boldsymbol{s}_t$. The side information consists of age, gender, residency and other demographics. Machine learning algorithms choose digital interventions, $\boldsymbol{a}_t$, based on the observations, $\boldsymbol{x}_t$. The next state of the preferences $\boldsymbol{p}_{t+1}$ are modeled as a function current preferences, the side information and the digital interventions. With such a model, the interventions can be chosen in a way to update the user preferences. The model also takes into account random noise effects, indicated by $\boldsymbol{v}_t$ in the figure.

In Chapter 2, we study how the data input of the model, $\boldsymbol{x}_t$ and $\boldsymbol{s}_t$ are produced and processed. This data input can be extracted from the abundant digital footprint of the user (about 13GB per month per capita, for a total of 96 exabytes (96 billion gigabytes) in 2016 (Cisco 2017)). We take a look at the steps involved in processing (preprocessing, encoding and processing) this enormous amount of raw data efficiently

and effectively to extract meaning and insight. In Chapter 3, we focus on the anomalies in the data since irregularities can convey valuable information as well as the mainstream data. We study the overall anomaly detection landscape and then introduce a novel anomaly detection algorithm. In Chapter 4, we dive deep into the state-space model to mathematically prove that user preferences, $\boldsymbol{p}_t$, can be changed by the machine learning algorithms in a desired manner.

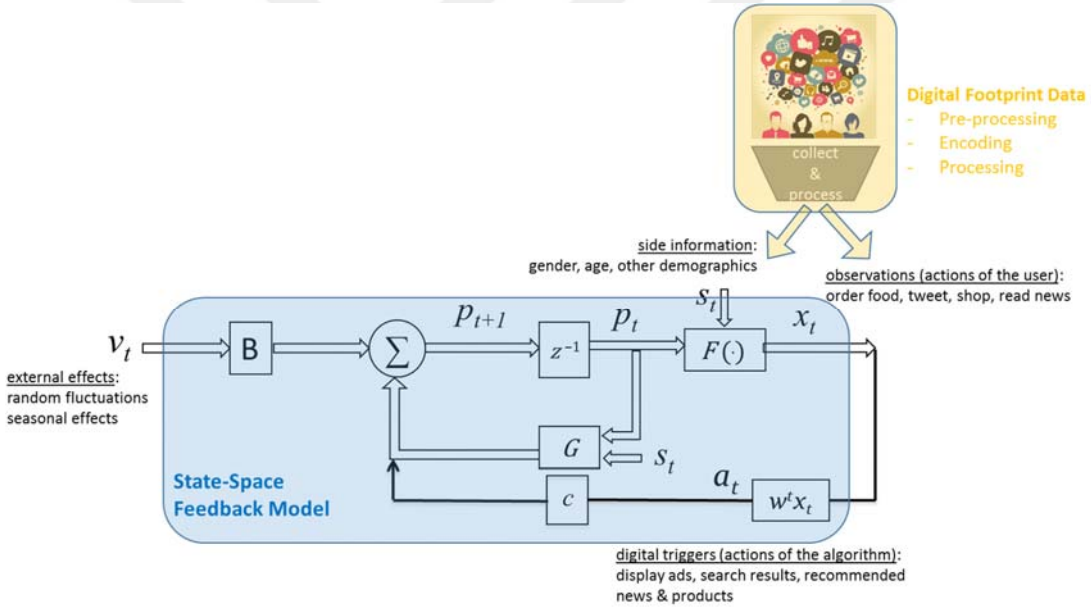


Figure 2 The State-Space Model with Feedback

1.1 Datasets

The data set we use in Chapter 2 is a collection of tweets gathered through a case-study which includes 1440 tweets written in Turkish. The tweets are collected from 168

different users between April 10th, 2013 and May 28th, 2013. There are at most 10 tweets from a single user. The tweets, whose contents can be related to anything, are freely worded and unstructured. There are 3 classes, i.e., a tweet falls into one of three categories, which are “No Statement(0)”, “Specific(1)” and “General(2)”. These categories reflect the level of statement about other people in a tweet. Tweets are manually labeled by human coders. Three coders were trained for six months. A subset of their coding results were cross-checked by an expert in order to make sure their coding results were correct and consistent with each other.

In Chapter 3, we use the Istanbul Stock Exchange (ISE) dataset for real data benchmark purposes. Daily price data of nine stocks are downloaded from ‘http://finance.yahoo.com’ and ‘http://imkb.gov.tr’ from January 5, 2009 to February 22, 2011 and the prices are converted to returns over 536 samples. We randomly add 64 (nearly 10 percent) anomalous samples to this dataset. The anomalous data are generated from a multivariate Gaussian process whose mean is the negative of the batch mean of the nominal data (with the same estimated covariance).

We made an effort to use existing datasets to validate and verify our algorithms and models. However, there are inherent difficulties with using true big data to be able to do validation at a large scale. We touch on such difficulties in the remainder of this section.

Data is abundant and ever increasing due to an increasing number of new digital platforms and usage methods (Bradlow et al. 2017). The impression of data being abundant is typically fueled by data conglomerates such as Facebook, Google and Amazon (Wedel and Kannan 2016). Another giant, Walmart, collects 2.5 petabytes of data from 1 million customers every hour about transactions, customer behavior and a number of other things (McAfee et al. 2012). There is an obvious theme here, big data

implies big user base. There are other sources of data such as surveys and experiments. However, these not only lack the “big” qualifier but also suffer from some inherent problems. The accuracy of many surveys and experiments are subject to misrepresentation and self-consciousness. Many times these surveys and interviews involve counterfactual questions which may lead to major inaccuracies. Since experiment setup may influence the results of the experiment, especially in retail, companies prefer field experiments rather than lab experiments (Sudhir 2016).

Algorithms and models developed for predicting user preferences, psychological traits and states require big data. These algorithms and models evolve to make better use of data size and variety. Insights that might otherwise be completely obscure come into light with the use of advanced models and big data. The data needs to be not only big but timely as well (Bradlow et al. 2017). Micro-targeting and behavioral targeting requires real-time or near real-time data. Context, including location and psychological states, is transient. The relevance of insights gained from the data might have a short period of effectiveness. Therefore both data collection and processing need to be very efficient to allow real-time results.

2 From Raw Data to Insight

In this chapter, we study the data aspect of our state-space model. The main data input to the model is about user actions, the observable reflections of latent user preferences. A supplemental set of data input to the model is the user side information which consists of age, gender, residency and other demographics. We take a close look at the steps involved in going from raw data to a form of data that can be fed into the model. We outline the state of the art for these steps of data processing. While this chapter focuses on methods which use the mainstream data points to extract knowledge for the model, the data points that are not part of the mainstream data can be precious. In the next chapter, we look into how we can make use of irregular or outlier data points to get further information for the model.

Data is deemed very valuable. It is the new "oil". It is also the new bacon and the new currency… Data is valuable, only if one knows how to use it and what to do with it (Sondergaard 2015):

> *Big data is the oil of the 21st century. But for all of its value, data is inherently dumb. It doesn't actually* do *anything unless you know how to use it.*

While the mathematical tools and statistical fundamentals for processing raw data have been around for a long time, there was neither sufficient useable data nor the amount of data that could be processed was enough to yield any revelations up until 2010s. The wide spread use of social media contributed significantly to available data. The processing power of computers increased enormously over the last three decades. As of June 2017, the world's most powerful system, developed by China's National Research Center of Parallel Computer Engineering & Technology, has a performance of 93 petaflops (https://www.top500.org/lists/2017/06/). That is about 5 million times more capable than a Cray-2 supercomputer was in 1985. With the advances in big data, huge amounts of data can be processed to extract information and insight that would have otherwise been impossible. Computers started doing the tasks of categorization, encoding and classification not only much faster but also more accurately than humans. In the 2015 ImageNet Classification Challenge, Google's deep learning algorithm was able to beat a human annotator's error rate (Russakovsky et. al. 2015). In a study with 86,200 Facebook users who took personality tests, computers predicted personality more accurately than the Facebook friends of the participants, based on Facebook likes (Youyoua et al. 2014).

In fact, the abundant raw data of today's life style is sufficient to Both the main and supplemental data inputs to the model can be We take a look at the steps involved in processing (preprocessing, encoding and processing) this

Today, the problem is not the lack of data. Data is abundant. Willingly or unwillingly, people share data more than ever. Free and ubiquitous access to social media platforms, combined with people's self-expression needs, create a huge amount of data in the public domain. Social media posts include opinions, feelings and facts. The key task is to process this enormous amount of raw data efficiently and effectively to extract meaning and insight. Precise and precious information can be extracted from these otherwise seemingly useless data (Popescu and Baruh 2013; Huang et. al. 2015; LeCun et. al. 2015).

If people search for a car dealership using map applications on their cell phones, they are very likely to see car advertisements next time they use the internet. Soon after someone emails his/her friend about sports shoes, he/she is more than likely to see sports shoes advertisements. Individuals are tracked online to better understand their interests, behavior and preferences. In its privacy notice, Google clearly states that your emails are processed for such purposes (Google Privacy Notice 2017). However, most people do not mind this and allow companies and organizations to use their data in exchange for free services such as email (Datatilsynet 2016). Companies like Google and Facebook provide free services in order to collect this enormous amount of data.

Recently a new source of data has emerged through self-tracking. Being self-conscience along with having the capability to conveniently collect one's own data has fueled the concept of "The Quantified Self" (Swan 2013). People collect and track self-data on physical activities, diet, psychological and mental states, environment, any social or

contextual situation (Augemberg 2012). Virtually all smart phones have applications to collect physical activity statistics. Data from biosensors add up to huge amounts. 232 million wearable fitness tracking devices were sold in 2015 (IMS 2015). Technology companies such as Validic has access to patient data from over 400 sources reaching 223M individuals in 52 countries (https://validic.com).

As abundant as the data is, not all of it is the same in conveying useful information. There is a conceptual usefulness hierarchy in data.

An ounce of information is worth a pound of data.

An ounce of knowledge is worth a pound of information.

An ounce of understanding is worth a pound of knowledge (Ackoff).

In the process of extracting something beneficial and useful from data, feeling the need for more formalization, initial proposals about a conceptual hierarchy were made in the 1980s (Zeleny 1987; Ackoff 1989). As significance of data was understood more and a need emerged for going from raw data to more refined forms of data, other proposals came about to finally converge on the widely used concept of Data-Information-Knowledge-Wisdom (DIKW) (Rowley 2007; Awad and Ghaziri 2004).

As one moves higher in the DIKW hierarchy, more insight and enlightenment can be obtained. There is more meaning and value in the higher layers. Data, at the bottom, does not provide much insight as is. Raw data is a collection of records: manual or online survey results, phone call records (commonly referred to as call data records or CDR), search and web browsing history, social media posts including voice, video and text. It needs to be processed to take our insights to the next level and turn it into information and then to knowledge and wisdom. Processing the data at the lower layers

requires compute power and storage capacity but conceptually it is not complex. Since the abstraction increases at the higher levels, computer processing and programmability decreases (Rowley 2007). More sophisticated forms of data processing is needed. In the next section, we look into the problem of data processing more closely.

2.1 Data Processing

Data in question can be in the form of text, image, video or sound. In this thesis, we limit our studies to text data due to time and resource constraints. The text data can come from social media, surveys, interviews, TV program transcripts, books, papers or any kind of source that produces text. Today's computer systems are capable of processing enormous amounts of data. However, this sheer processing power may not be sufficient for extracting information and meaning hidden in the higher layers of the insight hierarchy. This is where advanced computer algorithms such as deep learning (LeCun et. al. 2015) come into the picture to extract meanings that may not be obvious to humans.

The processing of textual data by computers can be conceptually considered in three sequential steps: pre-processing, encoding, and processing.

2.1.1 Pre-processing

Computer programming requires pre-processing of text before any kind of encoding or classification can be done. Especially free form text can be challenging to interpret by computers and therefore requires some form of pre-processing to alleviate this challenge. Social media data is one of the main sources of free form text. Use of informal and incorrect language with incorrect grammar and spelling as well as use of

emoticons and non-standard abbreviations make social media text processing challenging (Farzindar and Inkpen 2015). To deal with this problem, some of the most common text pre-processing steps are:

- **Corpus Cleaning**: Removal of duplicate messages such as retweets, removal of URLs and links.
- **Spelling Correction**: Correction of intentional or unintentional spelling errors: “coooool” to “cool”, “u comin?” to “are you coming?”. Techniques using Recurrent Neural Networks (RNN) and Long-Short Term Memory (LSTM) can correct spelling errors at success rates well above 90%.
- **Stop Word Removal**: Some words are common words in a language that do not convey any extra information or meaning. “and”, “are”, “the” are some of the stop words that need to be removed in pre-processing.
- **Stemming and Lemmatization:** The goal of stemming is to reach a common root or base of a word and it may not always be an actual word. For example, the stem for the word “typed” may be processed as “typ” since there are words like “typing”. Lemmatization would, however, try to reach at a real root of this word as “type”.
- **Tokenization**: Chopping a stream of text into words or elements that can be processed more meaningfully. It typically involves finding word boundaries and removal of punctuation and special characters.

2.1.2 Encoding and Representation

Encoding is one of the techniques used in the process of extracting insight from raw data in applied sciences (Desrosieres 1998). Data encoding puts potentially very large

and free form text in more concise forms which facilitates meaning extraction and thus helps move up in the insight hierarchy. Encoding involves assigning attributes and concepts from data to categories in a systematic way (Lockyer 2004). These attributes and concepts are codes which are typically words, numbers or short phrases to capture the true essence of the data. Codes can be qualitative such as encoding the concept in the following text as "*security*" (Saldana 2009):

> *I notice that the grand majority of homes have chain link fences in front of them. There are many dogs (mostly German shepherds) with signs on fences that say "Beware of the Dog."*

Codes can also be numerical such as the number of times a particular word is used in a script.

Taxonomy, the science of encoding and classification, requires categories. According to Desrosieres, there are two approaches to forming categories. The first one is the nominalist approach where categories are pre-defined by public norms, conventions or commonly accepted procedures. The second approach infers categories based on data. As more data analysis takes place, categories may get redefined. Thus, categories are fluid and can be determined based on a method as opposed to a pre-defined system of the first approach (Desrosieres 1998). The traditional machine learning tasks of classification and clustering have remarkable similarities to these two approaches, respectively. In machine learning, classification is the task of assigning a data point to one of the pre-determined categories. This process resembles the nominalist approach since the categories are determined in advance and the task at hand is to make assignments to those categories. In contrast, clustering in machine learning uses a

method to determine the categories in a similar manner to the second approach as described by Desrosieres.

Data encoding not only helps human interpretation and analysis but also makes computer analysis more straight forward (Bourque 2004). Computer analysis of textual data requires representation. Computers can interpret digital data that consist of 1's and 0's. There are potentially countless ways of representing text as 1's and 0's. The key is to find a way to model it such that these models can be manipulated algebraically in an efficient way and also carry contextual information to enable complex analysis such as semantic analysis. A representation that can capture contextual information enables smart algorithms to conduct complex analysis without the need for explicit and manual encoding. The sentiment of a sentence can be determined to be "sad", "happy" or even "sarcastic" without using explicit encoding.

The most common way of text representation is as vectors. This geometric representation of words as vectors is called "word embedding". Word embedding is the process of mapping strings, more specifically words, into vectors. One of the simplest word embedding choices is to encode them as *one-hot* vectors, i.e. represent the existence of a word by 1 and the opposite by 0. The English language, having about 13 million tokens, would have vectors of size 13 million each as:

aardvark = [1 0 0 … 0], a = [0 1 0 … 0], at = [0 0 1 … 0], …, zebra = [0 0 … 0 1]

While this encoding is quite intuitive, it is not very useful since it does not convey any contextual or semantic information. Also, being very sparse makes it inefficient for computer processing. Thus, the main goals of word embeddings are to minimize vector dimensions for efficient computations as well as to find a geometric representation to

capture semantic relationship between words. It has been shown that words that have a semantic relationship tend to appear in similar context (Miller and Charles 1991). Based on this principle, two categories of models for deriving word embeddings have emerged (Baroni et. al. 2014):

- **Count Based Models**: The main idea is to count the number of times a particular word is seen within a certain proximity (window) of each word in the available data sets, i.e. corpora. Words like “school”, “student”, “book”, “teacher” are likely to appear together while “school”, “sushi”, “romance”, and “surfing” are not. Forming co-occurrence based counts requires still very large dimensional matrices. Dimensionality reduction techniques are applied to make the dimensions manageable while retaining contextual information. GloVe (Global Vectors for Word Representation) (Pennington et. al. 2014) is one of the most widely used co-occurrence methods. Count based models can be formed in an unsupervised manner, that is, they can be formed without human supervision.
- **Predictive Models**: The goal is to predict words based on their context. Given context words, the model tries to predict the center word. In the sentence “A car ***accelerates*** uniformly from rest”, the Continuous Bag of Words (CBOW) Model treats {“A”, “car”, “uniformly”, “from”, “rest”} as context and tries to predict the center word “accelerates”. Another well-known predictive algorithm, skip-gram, does just the opposite: given the center word, it tries to predict the context words. Predictive models use supervised algorithms. However, they do not incur the manual annotation costs associated with supervised algorithms since the context windows used for training can be automatically extracted from an unannotated corpus (Baroni et. al. 2014).

Once textual data are represented as geometric word vectors. Similarity is simply a measure of distance between these vectors. Some of the most common distance measures used by both classification and clustering tasks are cosine, Jaccard and Euclidean. If two words are represented as d-dimensional vectors $\boldsymbol{u} \in \mathbb{R}^d$ and $\boldsymbol{v} \in \mathbb{R}^d$, the Euclidean distance between these two vectors would simply be

$$\|\boldsymbol{u} - \boldsymbol{v}\| = \sqrt{\sum_{i=1}^{d} (u_i - v_i)^2}.$$

The Jaccard similarity between two sets of words is the intersection of these sets divided by the union of the sets. Cosine similarity is especially used in measuring semantic similarity (Lintean and Rus 2012). It is the cosine angle between two vectors which indicates how close the two vectors are:

$$\cos\theta = \frac{\boldsymbol{u}.\boldsymbol{v}}{\|\boldsymbol{u}\|\|\boldsymbol{v}\|}$$

2.1.3 Processing and Applications

Classification and clustering are common information extraction tasks used in the processing phase. In classification, the machine learning algorithm to assign data points to categories learns based on training data points whose category information are already labeled, typically, by humans. This kind of learning is called supervised learning since a training set of correctly assigned data points are available. The algorithm learns from this training set and makes assignments for new data points whose category information is not available. In contrast, the technique of clustering is used when training data is not available. Clustering involves grouping together data

points that are similar to each other. Each cluster makes a category which is formed in a fluid manner as data is analyzed. A similarity measure is needed to make a decision on which data points can form a cluster. Clustering is a type of unsupervised learning since there is no need for training data and thus labeling by humans.

As one moves higher in the Insight Hierarchy, information becomes more subtle. Meaning extraction based on context, something humans do naturally, is extra challenging for computers. Scientists and engineers look for algorithms and methods to process and analyze text semantically without human intervention,

Recurrent neural networks (RNN) are one of the most powerful and widely used topologies in text processing problems since they are well-suited for sequential patterns (Liu et. al. 2016). In the process of predicting the next word in a sentence, the actual sequence of words that came before that particular word matters and hence such problems fit nicely into the solution space of RNNs. While deep neural networks typically consist of many layers of neurons each with a different set of parameters, there is no feedback and thus, deep neural networks are not sensitive to the order in the input sequence. In contrast, RNNs use a feedback mechanism to account for the impact of the previous input in computing the state of the neuron for the current input. This mechanism effectively acts as a memory and make RNNs work well with inputs where sequences matter. The closer in position the inputs are, the more impact they have on the state of the neuron. This impact can sometimes vanish quickly over a few positions in input data and longer term dependencies in input sequences are effectively ignored those cases. To overcome this problem, long short-term memory networks (LSTM), a particular flavor of RNNs with longer term dependencies, are more commonly used.

In many text processing applications, the inputs are sentences that are a sequence of words, expressed as word embeddings (vector representation as explained in section 2.1.2 Encoding and Representation). The task might be to predict the probability of each word in a sentence. This, in turn, helps determine the correctness of a sentence by looking at how probable the sentence might be. This is a typical application of LTSSMs in machine translation. With the ability of predicting the next word in a sentence, once can generate new sentences by using high probability sequence of words. This opens up the possibility of very interesting use cases. Some of the most popular application areas for text processing are:

- **Question Answering:** Answering human questions by automated systems in a way humans would.
- **Machine Translation:** Auto translation of text entered by humans on the fly or pre-existing text.
- **Semantic Search:** The goal is to predict and understand the intent of the user in context rather than merely matching search terms in documents.
- **Named Entity Recognition (NER):** Finding entities in text such as names of people, companies, and locations.
- **Sentiment Analysis:** Involves determining the sentiment of author, typically as positive, negative and neutral.
- **Summarization:** Creating a summary of a document.
- **Disambiguation:** Finds the actual meaning of a word in the context it is used when a word has multiple potential meanings.
- **Part-of-speech Tagging (POS):** Marks word as nouns, verbs, adjectives etc.

- **Recommendation:** Predicting the preferences of a user in order to recommend items such as consumer goods, news articles and music.

2.2 Publications

Our publication in the field of processing raw data to extract more meaning is provided in Appendix A.

3 ANOMALY DETECTION

In the last chapter, we looked into ways of extracting information from raw data to feed into our state-space model. In this chapter, we continue with the data processing aspect of our model, however, we focus on outliers as opposed to the mainstream data points which were the focus of the last chapter. We study anomaly detection, the main technique to deal with outliers, in terms of its challenges, most common applications and algorithms. Using anomaly detection does not only provide additional information which the state-space model can benefit from but also fits into the model due to the nature of our main data source, specifically social media text data. As we will see in this chapter, such data is incomplete and contains incorrect information which need to be imputed and cleaned using anomaly detection. We will also look into another interesting use case of anomaly detection, identifying high impact content and people, i.e. influencers, and how that can be used by our model. In the next chapter, we look into

the facts that lead to our model before we provide a deep dive of how we build our model and then provide a mathematical proof of the effectiveness of the model.

The common way of extracting insight from raw data is to use the data points that convey mainstream information. Knowledge is built on these examples. Outliers and exceptions are not considered. If data is so valuable, why would we ignore and waste these irregular data points? These irregularities might be pointing to peculiar and unusual events that have just happened or are likely to happen soon. And that is information leading to insight. Detecting a change in spending habits of a person can point to a fraud in progress. An unusual traffic pattern on the network could signal a cyber-attack. A change in the heart rate of a person can be the early sign of a health condition. These problems of the modern world lend themselves to techniques used in anomaly detection. Anomaly detection has been an area of research since the end of 1800s (Edgeworth 1887) and continued at a modest pace until early 2000s (Chandola et. al 2009). With the rise of machine learning and deep learning algorithms, and the wide variety of use cases, anomaly detection has been an area of research focus recently.

3.1 Nature of Detection Problems and Challenges

Anomaly is deviation from what is expected. Observations that deviate from other observations indicate a potentially different underlying generation mechanism (Hawkins 1980). Anomaly detection focuses on the problem of detecting unusual data points or observations and predicting any future occurrences of such data points along with underlying events. One of the main challenges is to define what is usual and expected so that deviations from the expected can be identified and detected. Many times, the boundary between what is expected and what is unusual is fuzzy. The blurry boundary

between these two cases can cause positive data points to be detected as negative and vice versa. These incorrectly identified cases are called false positives and false negatives. Especially, ill-intentioned attempts, such as network intrusion, try to imitate regular events. When an intrusion detection system signals an intrusion incorrectly, a number of actions might be taken that are unnecessary. These false positives can not only impact regular system operation but may also be a cause of extra cost. In a medical application, false negatives can be fatal while false positives may lead to unnecessary cost and psychological stress. The nature of expected data can be time-dependent. An increase in spending habits over the holidays has a higher likelihood of being a normal event compared to an increase during other times. For a data point to be classified as unexpected, there has to be a certain amount of deviation from the expected and this amount is context-dependent. Subtle changes in human cell characteristics can be the cause of a serious condition (Lyons et. al 2016) while significant changes in spending habits may be quite normal. Another challenge is related to the nature of the problem. Even though data is abundant, enormous and widely available, anomalies are rare and irregular by definition. This makes the learning and inference processes extra difficult since the more data the better the learning. Many anomaly applications, such as network intrusion, terrorism detection, cancer detection, require quick action, therefore quick detection. Many times data is noisy and requires noise to be eliminated to detect anomalies since it can obscure anomaly detection.

3.2 Algorithms and Approaches

When a data point can be identified as irregular or unusual in comparison to the rest of the points in the dataset, this is a case of point anomaly detection (Chandola et. al 2009).

Collective (Chandola et. al 2009) or group anomaly (Yu et. al 2010) occurs when a group of data points are considered to be irregular or unusual in comparison to the rest of the points in the dataset. Detecting a disease outbreak is an example of group anomaly.

In applications where labeled data is available, supervised anomaly detection techniques can be applied. However, just as in any supervised learning case, labeling data can be cumbersome and costly. In addition, the scarcity of anomalous cases provides an extra challenge to traditional supervised learning algorithms. Classification-based algorithms suffer from these problems. They learn what is normal and not based on a dataset that has data points labeled as expected and unexpected. They compensate for the lack of sufficient unexpected data points (anomalies) by using oversampling techniques. Unsupervised anomaly detection algorithms do not require labeled training data. The algorithm tries to identify anomalous cases based on the nature and properties of the data it has seen so far. This allows for more powerful and generalized algorithms since there is no requirement to anticipate the nature of the data in advance. In contrast, the success rate of supervised learning might be limited by the extent of available anomalous cases in the training data.

One of the widely used approaches in many anomaly detection algorithms is to build a case for expected behavior and then try to detect data points that do not fit in. Statistical approaches using a parametric model for the data distribution build on certain expectations of the data. The model defines a confidence interval based on the expected number of anomalies and tries to detect the cases outside of this interval. While parametric models converge quickly, they can be overly erroneous if the actual data distribution does not match the distribution assumption. Estimating the underlying

distribution can be especially challenging for high dimensional data. Non-parametric statistical approaches build their models without pre-defined expectations. The model is built based on the data.

When data is represent the data in a vector form, various distance-based approaches can be used. Such algorithms choose a distance metric. For instance, kNN, a nearest neighbor algorithm, measures the distance between each data pair and then defines its distance to the k^{th} neighbor as a threshold beyond which data points are considered to be outliers. Another measure of distance can be density, defined as the number of data points within a radius.

Online anomaly detection algorithms are used in cases where data arrives in a sequential manner and on the fly decision has to be made for immediate feedback. The model classifies the incoming data as anomalous or not and then updates itself.

3.3 Applications

The relatively few unusual data points in these routine data sets may contain very useful information. This inherent feature of anomaly detection enables it to be applied over a wide range of applications. By using anomaly detection techniques, one can extract very valuable and actionable information from data sets that are mostly routine and boring. The outbreak of a contagious disease can be detected from social media data (Xie et al. 2013). The routine financial transactions of a person can contain a few unusual transactions which may be an indicator of fraud. In a more subtle use case, some unusual, non-fraudulent nonetheless, spending patterns of people can help identify them as significant targets in advertising campaigns. Similarly, anomaly detection can be

used to identify opinion leaders in social media based on some rare but critical attributes.

Anomaly detection applications can be grouped and classified based on a wide range of attributes. Online vs offline, image vs text or nature of the data such as social media, medical, financial or astrological. Many applications include multiple such attributes and as such, it is possible to group them in a variety of ways. Online systems are typically used for early warning systems that require timely detection (near real-time) so that quick action can be taken. Predictive actions may also be necessary as a consequence of early detection (Goldstein and Uchida 2016). A medical diagnostic anomaly detection application is typically done offline while it is also an image detection application.

We will list a number of common as well as niche applications without grouping them with respect to their attributes.

- **Data imputation and cleaning:** Many times data is noisy and needs to be cleaned before meaningful information and insight can be extracted. These are cases when unusual or irregular data points are not considered to contain extra information and in fact are considered to pollute the mainstream data. Intentional or unintentional injection of incorrect information in social media data can make it hard to find useful information and need to be cleaned. This use case is particularly important for our state-space model since social media data is one of the major sources of easily attainable digital footprint of users. Similarly, surveys may also contain incorrect information and need to be detected and cleaned. Noise in sensor data is an anomaly and makes it harder to read the actual signal values from the sensor. The source of noise can be due to

inaccuracy of the sensor as well as the quality of the network the data is on (the data points may be dropped or pick up noise as they are being transported over this network). Anomaly detection techniques can be used to detect these cases to clean and normalize that data.

- **Identifying Influencers and Niche Customers:** Social media influencers can have a significant positive impact on sales. Influencer are not limited to celebrities and famous experts. Anyone might be an influencer and identifying these influencers is one of the first main tasks of marketer working in this field (Ranga and Sharma 2014). One might approach the problem of identifying influencers as an anomaly detection application since they are different from the norm. Similarly, while niche marketing can result in more profitability and better market share, there has not been much formal research in the area of identifying niche customers (Toften and Hammervoll 2013). Niche customers have rare traits that make them suitable for anomaly detection applications. Also, high impact content such as online advertisements or news feeds relatively rare and are potential applications of anomaly detection. Being able to identify high impact content and influencers can facilitate steering users in making certain decisions and taking certain actions. We will look futher into this interesting consequence Chapter 4.
- **Fraud Detection:** Frauds detection aims to identify unauthorized or unlawful transactions by criminals in various financial fields such as banking, insurance and securities trading. These transactions are expected to have a pattern that is different from usual ones and therefore are a good fit as an anomaly detection application. While timely detection is critical in order to minimize the

economical negative impact it might have, precision is also very important since companies do not want to bother customers with false fraud cases.

- **Early Detection Systems:** Detecting an anomaly early (days or weeks in advance) or in near real-time can be crucial in a wide variety of applications. Analysis of biosensor data can catch early warnings of some **medical conditions** such as cardiac events, skin diseases or diabetes (Swan 2013). Other applications of early detection systems are typically cases of collective anomaly detection. **Disease outbreaks**, **natural disasters** and **terror plots and events** can be detected as anomalies in social media data.
- **Network Intrusion and Cyber-attack Detection:** These types of events are especially challenging to detect since cyber-attackers try to conceal their attacks by mimicking the behavior of regular network events.
- **Data Center Monitoring:** Large data centers must do continuous load-balancing between machines to optimize performance and profit. The particular load conditions to take an action might be defined as an anomaly and detected accordingly.
- **Physical Security:** Image and other sensor data are processed to detect anomalies such as intruders.
- **Astrological Events:** Anomaly detection techniques are used on astronomical data for various research activities such as detecting unique and interesting objects.
- **Manufacturing**: Defect and fault detection are essentially about detecting anomalies to prevent bad parts and products from reaching customers.

3.4 Publications

Our publication in the field of anomaly detection is provided in Appendix B.

4 ACTIONS TO STEER PREFERENCES

In Chapters 2 and 3, we studied how the data inputs to our state-space model are produced and processed. In this chapter, we focus on the model itself which constitutes the core of this thesis. We start by making observations that support the foundational motivations of our model, that is, user preferences can be steered by digital interventions, which are in most cases the actions or outcomes of machine learning algorithms. Then we build a state-space model from ground up and then provide a mathematical proof that user preferences can be steered in a desired manner by the actions of machine learning algorithms.

4.1 Motivation

It has been shown by large scale experiments that emotional states of people can be transferred to others via social networks (Kramera et. al. 2014). The news in one's Facebook feed and people's response to these news may impact one's emotions and preferences. In many cases, the people being affected may not even notice the transfer of emotions.

One of the most well-known cases of steering user preferences has been happening in the United States recently. There is significant amount of suspicion regarding the influence of Russians in the 2016 United States presidential elections. It is claimed that Russians meddled with the U.S. elections by using social media ads and unpaid social media messages. The U.S. Senate held a hearing with representatives of the social media platforms, Facebook, Twitter and YouTube on this topic in November 2017 (https://www.intelligence.senate.gov/hearings/open-hearing-social-media-influence-2016-us-elections).
The ads and other social media messages from Russian-linked accounts allegedly aimed to create racial and political division among the U.S. public. They portrayed Hillary Clinton, the democratic candidate, as evil and also tried to flare sensitive topics such as police brutality on blacks. A sample set of these messages can be found in https://www.nytimes.com/2017/11/01/us/politics/russia-2016-election-facebook.html. Most of the polls and predictions before the elections were pointing to Hillary Clinton as the winner of the election. https://www.270towin.com/2016-election-forecast-predictions/ summarizes predictions from a wide range sources including far right and far left sources. Virtually all predictions point to Hillary Clinton as a clear winner over Donald Trump. This makes a strong point for the claim that social media ads and messages, allegedly by Russians, made significant impact on the election results. What makes these events even

more noteworthy is what it takes to achieve the desired results. While the actual amount of money spent on these advertisements and messages is not determined yet, it is estimated to be on the order of a few hundred thousand dollars since 2015 (https://www.forbes.com/sites/kathleenchaykowski/2017/10/18/facebook-investigates-how-messenger-app-was-used-in-russian-meddling/#71f717a53af5). If we assume for a moment that these messages really helped swing the votes of a number of U.S. citizens to change the election results, a huge historical impact was achieved with such a low budget.

According to a Wall Street Journal investigation in 2012 (https://www.wsj.com/articles/SB10001424052970203347104578099122530080836), in the days leading to the 2012 U.S. Presidential elections, Google produced customized search results when users searched for Barack Obama, the democratic candidate in the election. However, no customized results were returned for the searches that involved the republican candidate Mitt Romney. At this point it is not possible to tell how much positive impact Google's search engine behavior had on the election results but the fact remains that Barack Obama won the 2012 elections. A research study in the United States and India showed that biased search rankings can sway undecided voters by at least 20% (Epstein and Robertson 2015).

It is striking to note that Zarsky pointed out the potential impact of online content providers on the decision making process of people, well before any of Facebook, YouTube and Twitter were founded (Zarsky 2004). The digital footprint of an internet user has increased by about 100 times since then. In 2005, one year after Zarsky's analysis was published, the amount of global internet traffic was under 3 exabytes

(http://www.hbtf.org/files/cisco_IPforecast.pdf), about 1/100th of what it was in 2016 (Cisco 2017).

> *[…] the mixture of several novel elements produces a result that might prove harmful to our personal autonomy. These elements are (1) information providers' enhanced ability to garner personal data about their users, especially regarding their interests, preferences, and possible vulnerabilities; (2) their ability to analyze such data in an automatic and efficient manner through the use of data mining applications; and (3) their capability to reach out and provide every user with a personalized package of content, especially by making use of the Internet's unique infrastructure. Content providers will not only tailor their content to the specific individual upon delivery, but can constantly assess the effectiveness of their marketing schemes and persuasion attempts through the new and updated personal data streaming in from relevant users. Thus, they can create a feedback loop for every user — with the ability to constantly change the information they provide, until they achieve an optimal outcome. These abilities lead to the enhanced opportunity to unfairly persuade and manipulate — a power vested in the hands of a few — and raise concerns both in the context of commercial advertising and agenda-setting by mass media editors.*

4.2 Background

The conceptual foundation of our model is easy to understand since virtually any internet user can relate to it. However, the mathematical notation and formation may require some background information to follow. In this section, we explain why we choose to use a state-space model. Then, we provide a brief background on state-space models and Extended Kalman Filters, a framework we use to formulate our mathematical proof.

Statistical models can be discriminative or generative. Discriminative models, e.g. support vector machines (SVM) and logistic regression, focus on the differences in data. On the other hand, generative models, e.g. Linear Gaussian systems, Hidden Markov models (HMM) and state-space model, mimic the source and define the type of expected data. Thus, they are better at handling missing and incorrect data. As explained in Chapter 3, social media data, one of the major data sources to our model, can be incomplete and incorrect, making state-space model a well fit for this case. Another feature of HMMs is that the states of the model are hidden and can only be observed indirectly via the observed data. User preferences are very similar to Hidden Markov states since they are also latent and indirectly observed via digital actions and footprints of the user. In our model, an additional observation stage is added to represent user actions based on the latent (unobservable) internal state. The complete system is designed to estimate the latent state of the users from observations.

HMMs have hidden states with certain probabilities of transitions between states. State-space models are very similar to HMMs, except that they are continuous while HMM states are discrete. Figure 3 shows a simplistic example HMM with hidden states and observable actions.

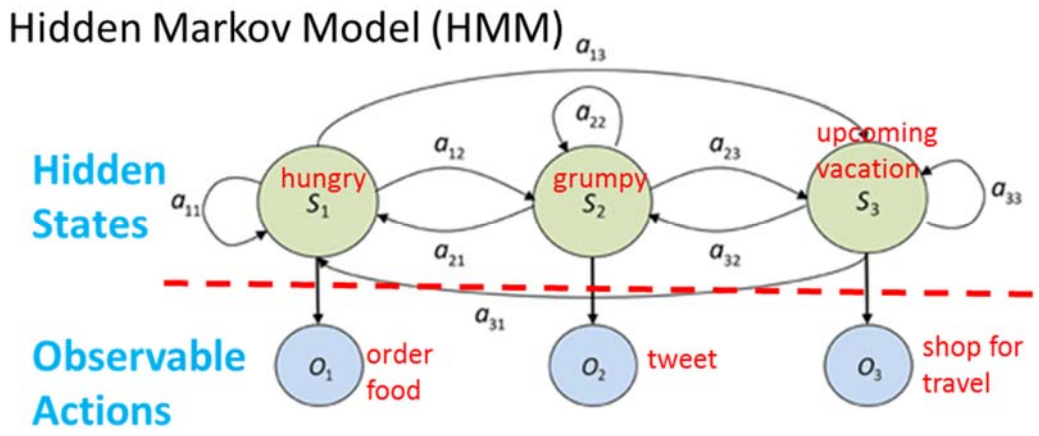


Figure 3 An Example Hidden Markov Model

The Kalman filter, originally invented by Rudolph E. Kalman (Kalman 1960), is a recursive set of equations to *estimate* the state of a model or process by minimizing the measurement error (Welch and Bishop 2001). And the Extended Kalman Filter (EKF) is an approximation to deal with nonlinearities in the system. An excellent EKF tutorial exists in https://home.wlu.edu/~levys/kalman_tutorial/.

Using a state-space topology, we build a model to represent the current and next state of user preferences with the state update equations capturing the impact of digital interventions (actions or outputs of machine learning algorithms) and other factors such as noise. We formulate the parameters of our model such that they can be estimated via the EKF framework. Next, we derive recursive equations for the system parameters such that the sequence of user preferences are tuned towards a *desired* sequence of preferences, e.g., one can desire to sway the preferences of a user to a certain product.

Finally, we run experimental simulations to show that estimated parameters of the system converge to the real values of the system, proving that a system can be designed with the right parameters to allow a sequence of actions or interventions to tune the preferences of a user in a desired manner. The true parameters of the system are known to us since we are running our experiments in the form of simulations. Specifically, the preferences of the user, which are not directly observable in real life, are known in case of simulations. We run simulations for the EKF formulations we derive to show that our estimation of the preferences converge to the real preference values.

4.3 Publications

Our publication, provided in Appendix C, walks through the details of how build our model, make the mathematical proof of our claims and then the simulation results.

5 CONCLUSIONS AND FUTURE WORK

In this thesis, we model the effects of the machine learning algorithms such as recommendation engines on users through a causal feedback loop. To this end, we introduce a complete state space formulation modeling: (1) evolution of preference vectors, (2) observations generated by users, and (3) the causal feedback effects of the actions of machine learning algorithms on the system. All these parameters are jointly optimized through an Extended Kalman Filtering framework. We introduce algorithms to estimate the unknown system parameters with and without feedback. In both cases, all the parameters are estimated jointly. We emphasize that we provide a complete set of equations covering all the possible scenarios. To tune the preferences of users towards a desired sequence, we also introduce a linear feedback mechanism and introduce an

optimization framework using stochastic gradient descent algorithm. Unlike previous work that only use the observations to predict certain desired quantities, we specifically design outputs to “update” the internal state of the system in a desired manner. Through a set of experiments, we demonstrate the convergence behavior of our proposed algorithms in different scenarios.

Our work achieves a significant first step in designing a system which allows a sequence of actions or interventions to tune the preferences of a user in a desired manner. We establish a pathway to design such a system. We provide a model of the system and support its theoretical foundation with mathematical proof and simulations that such a system can be built with the right parameters. We also provide some of the building blocks of the system which include ways to process input data to feed the system.

In Chapter 2, we study the data aspect of our model. The main data input to the model is about user actions, the observable reflections of latent user preferences. A supplemental set of data input to the model is the user side information which consists of age, gender, residency and other demographics. We take a close look at the steps involved in going from raw data to a form of data that can be fed into the model. This data input can be extracted from the abundant digital footprint of the user. We outline the state of the art for these steps of data processing. We present our work on classification and dimensionality reduction techniques such as random projections and also data pre-processing techniques such as corpus cleaning. This work constitutes one of the building blocks of our system.

In Chapter 3, we continue with the data processing aspect of our model, however, we focus on outliers as opposed to the mainstream data points which were the focus of

Chapter 2. Anomalies in the outliers can convey valuable information as well as the mainstream data. We study the overall anomaly detection landscape and then introduce a novel anomaly detection algorithm to be used in our system. Our algorithm processes data in a sequential manner. It observes new data, produces its decision and then adaptively updates all its parameters to enhance its performance. The algorithm mainly works in an unsupervised manner since in most real life applications labeling the data is impractical and costly. Using anomaly detection does not only provide additional information which the model can benefit from but also fits into the model due to the nature of our main data source, specifically social media text data. As we show in Chapter 3, such data is incomplete and contains incorrect information which need to be imputed and cleaned using anomaly detection. We also look into another interesting use case of anomaly detection, identifying high impact content and people, i.e. influencers, and how that can be used by our model.

In Chapter 4, we focus on the model itself which constitutes the core of this thesis. We start by making observations that support the foundational motivations of our model, that is, user preferences can be steered by digital interventions, which are in most cases the actions or outcomes of machine learning algorithms. Then we build a state-space model from ground up and then provide a mathematical proof that user preferences can be steered in a desired manner by the actions of machine learning algorithms. We support our proof with simulations.

While we complete the necessary groundwork to build a system in this thesis, a next step in future studies can be to make the system more stable and also to make the design process easy and practical for system designers. Further analysis on the convergence of the system along with more simulations, experiments and numerical analysis are needed

to take our results to the next level. A direct comparison to previous studies is not possible for this step of our study since, to the best of our knowledge, this is the first time a task of this nature is being undertaken. One of the main success criteria in our work is the fact that estimated parameters converge to the real parameter values. However, as our framework evolves, one can track its relative performance to our work.

It is worth pointing out that the validation of our framework using real life application data has significant difficulties. In order to measure the impact of machine learning algorithms on users at an individual level, interviews and/or surveys might be required. Hypothetical what-if questions would be asked in such interviews and surveys to understand what the user would prefer to do had he/she not encounter a particular set of digital interventions. These are counterfactual questions by definition. The accuracy of the answers to such questions inherently suffer from misrepresentation and self-consciousness. Therefore, it is not straight forward to set up these measurement techniques and get reasonably accurate results. Large scale A-B testing is a commonly used validation method to overcome these problems at population level (Wedel and Kannan 2016). In order to use A-B testing, one needs to have access to a large user base and a robust application. Mainly big data conglomerates and government institutions have such capabilities. A collaborative effort would be necessary to conduct such experiments. Data companies such as telecoms and banks have significant reservations in sharing any such data or collaborating at a meaningful level, mainly due to customer privacy and company secret security concerns. Lack of employees with the necessary skill sets to overcome these problems discourages companies from such collaborations with academia or other 3rd parties.

The psychological traits and states of people have successfully been predicted from various forms of data such as text, speech, video and other online data sources (Matz and Netzer 2017; Cowie et al 2000; Teixeira et al 2012; Alhanai et al 2017; LiKamWa et al 2013) in the academic world. As these predictions become more robust, accurate and consistent, we see modeling of human preferences make its way to the industry as a growing trend. One of the most successful and recent examples in the industry is Cambridge Analytica (CA) which ran the 2016 U.S. Presidential campaign for Donald Trump, helping him win against all odds. Only 22 out of 237 national polls in the U.S. estimated a Trump win (complete list in https://www.270towin.com/2016-polls-clinton-trump/national/). CA describes itself as a *behavioral science* and *micro-targeting* company. They collect a tremendous amount of data, close to 5000 data points on every adult American from healthcare to car ownership. They then correlate this data with the voting decisions to accurately predict voting behavior and more importantly to steer voting behavior.

Accurate analysis and prediction capabilities using big data has shown its impact in a number of other industry segments as well. An interesting example is the evolution of credit scoring. Credit decisions are shifting from traditional decision tools to using digital footprints of the users (Hurley and Adebayo 2016).

Next, we consider a specific real-life application of our system and discuss how one might approach designing such a system. Consider a navigation application with traffic information as a sample system to be used in a city, such as Istanbul, with a number of bridges. While the main goal of this application may be to help each user find an optimal route to their destination based on the congestion status of each bridge, a collective goal can be to optimize the average drive time of all users by balancing the

traffic across all bridges. In the course of achieving this goal, the system may need to suggest some users with suboptimal routes. The design of such a system would involve steering user preferences in a way that the machine learning algorithm desires. Specifically, the actions taken by the system would have to convince the users to take the routes suggested by the system. These actions have algorithmic as well as user experience aspects which support each other. The optimal sequence of actions that the algorithms select must be presented to the user in a way that will get the best perception and acceptance by the user. The drivers may want to see how much money and time they will save by taking the suggested route or in case they do not take the suggested route, show how much money and time they could have saved. Maybe it would help to convince the user to show the state of the bridge when they are projected to be on the bridge instead of the current state of the bridge. All of these are potential actions computed by the algorithms that require the right user experience to work as expected by the system.

Conversely, the overall user experience design of the system needs some key data to be collected and processed by the machine learning algorithms. This data can be collected within or outside the system. The data collected within the system can help the user experience. How often does the user select the suggested route? Is there a correlation between the suggested route selection rate and the route duration or distance, time of day, weather conditions, day of the week and other conditions? When the algorithms know the answers to these questions, the right information can be presented at the right time, providing better user experience and increasing the chance of success.

In the design of our sample system, another set of inputs may come from the data collected outside the system such as side information and user actions (observations).

This needs to be considered both at collective and individual basis. The system may collect and process tweets regarding traffic congestion during rush hours to learn tendencies of potential users collectively. On an individual basis, whenever the system can identify a user of the application on different platforms, it continues to increase its learning about the user. The identification of the user can be by the user opting in voluntarily. It is also possible for the system to approximately match the user based on various features such as behavior, style and location.

Another area of focus for future studies is the optimal selection of action sequences. The system has to decide on the best action strategy, based on the collected information on the user such as past spending patterns and eating habits. The obvious choice of action may not always be the best one. For instance, in the scope of a restaurant recommendation application, users might claim that they choose food based on their taste. The algorithm may learn in time to suggest restaurants, i.e., the product, to users predominantly based on what their friends prefer by giving more weight to this component in the space of context information. In a typical application, the system would have to decide among K available actions which can represent different products or services to be provided to the user. By keeping a loss (or conversely a utility) function based on the difference between the system action and the response of the user, the algorithm can enhance itself for subsequent actions. One of the algorithms that fit this setting is the multi-armed bandit problem (Katehakis and Veinott 1987) where each bandit arm corresponds to a different action, e.g., product or service to be recommended to the user. However this can be particularly challenging since user preferences can change over time due to the abundance of new products and services. Algorithms to optimally select actions may require online learning and decision making in real time to

accommodate these changes. This is a recent focus area of research (Gokcesu and Kozat 2017).

In our thesis, we do not focus on the ethical implications of the digital feedback loop. We merely point out and mathematically prove that user preferences can be steered in a desired manner. It has significant implications on human autonomy regardless of whether the underlying motives of the entities, e.g. data and content companies and governments, are positive or negative. According to the Internet Encyclopedia of Philosophy, "Autonomy is an individual's capacity for self-determination or self-governance." One of the practical implications of achieving this self-determination, one needs choice (Bernal 2014). Clearly, when the system in question is trying to steer the preferences of the user such that he/she makes the choice the system desires, an attempt at manipulating the users behavior is being made. And manipulation interferes with autonomy (Raz 1986). Even if the system makes a disclaimer about its intent, that can hardly alleviate the impact of its manipulation attempt because an extremely small percentage of users read end user agreements (Bakos 2014). Would people truly understand the implications when they read it? And even if they do, the practical choice they have is arguable. People opt in for free services especially from conglomerates like Google even when they know they are being tracked (Datatilsynet 2016). A counter argument to diminished autonomy might be giving the user an ability to make informed and good decisions. The system can examine and learn the tendencies of a user based on data and might be able to provide the user with the best available choices. The system can eliminate the penultimate choices and allow the user to make a better decision.

People with early signs of significant psychological problems can be identified and be guided to get help (which helps the society and the individual). On the flip side,

individuals who are prone to compulsive or addictive behavior could be targeted with ads for an online casino (Matz and Netzer 2017). While healthy eating habits can be promoted using models and algorithms described and referenced in this thesis, they can as well be used to market unhealthy food (Montgomery et al 2017), going as far as micro-targeting people with such inclinations.

6 REFERENCES

1. Ackoff, R. L. 1989. From data to wisdom. In *Journal of Applied Systems Analysis*.
2. AlHanai T, Ghassemi MM. 2017. Predicting latent narrative mood using audio and physiologic data. *AAAI 2017*.
3. Augemberg, K. 2012. http://measuredme.com/2012/10/building-that-perfect-quantified-self-app-notes-to-developers-and-qs-community-html/.
4. Awad, E. M. and Ghaziri, H. M. 2004. *Knowledge Management*. Pearson Education International, Upper Saddle River, NJ.
5. Bakos, Y., Marotta-Wurgler, F., and Trossen, D. R. 2014. Does Anyone Read the Fine Print? Consumer Attention to Standard Form Contracts. *New York University Law and Economics Working Papers*. Paper 195. http://lsr.nellco.org/nyu_lewp/195

6. Baroni, M.; Dinu, G. and Kruszewski G. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 238–247.
7. Bernal, P. 2014. Internet Privacy Rights: Rights to Protect Autonomy. Cambridge University Press.
8. Bernstein, J. H. 2009. The data-information-knowledge-wisdom hierarchy and its antithesis. In Jacob, E. K. and Kwasnik, B. (Eds.). (2009). *Proceedings North American Symposium on Knowledge Organization* Vol. 2, Syracuse, NY, pp. 68-75.
9. Bourque, L. B. 2004. "Coding." In *The Sage Encyclopedia of Social Science Research Methods*, Edited by Michael S. Lewis-Beck, Alan Bryman, and Timothy Futing Liao, v. 1, 132-136. Thousand Oaks, Calif.: Sage, 2004.
10. Bradlow, E. T., Gangwar, M., Kopalle, P., Sudhir Voleti, S. 2017. The Role of Big Data and Predictive Analytics in Retailing. In Journal of Retailing, Elsevier, (1, 2017) 79–95.
11. Brodersen, K. H., Gallusser, F., Koehler, J., Remy, N., and Scott, S. L. 2015. Inferring causal impact using Bayesian structural time-series models. In *The Annals of Applied Statistics*, vol. 9, no. 1, p. 247-274, 2015.
12. Chandola V., Banerjee, A. and Kumar, V. 2009. Anomaly Detection: A Survey. In *ACM Computing Surveys, September 2009*.
13. Cisco. 2017. The Zettabyte Era: Trends and Analysis. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html.

14. Cowie R, Douglas-Cowie E, Savvidou S, McMahon E, Sawey M, Schroder M. 2000. FEELTRACE: An instrument for recording perceived emotion in real time. *ISCA Tutorial and esearch Workshop (ITRW) on Speech and Emotion*.
15. Datatilsynet. 2016. Personal data in exchange for free services: an unhappy partnership? https://www.datatilsynet.no/globalassets/global/english/privacy-trends-2016.pdf.
16. Desrosieres, A. 1998. The politics of large numbers: a history of statistical reasoning. Harvard University Press.
17. Edgeworth, F. Y. 1887. On discordant observations. *Philosophical Magazine*.
18. Epstein, R. and Robertson, R. E. 2015. The search engine manipulation effect (SEME) and its possible impact on the outcomes of elections. In *Proceedings of the National Academy of Sciences* (*PNAS)*, published online before print August 4, 2015, doi: 10.1073/pnas.1419828112 PNAS August 18, 2015 vol. 112 no. 33 E4512-E4521.
19. Farzindar A. and Inkpen D. 2015. "Natural Language Processing for Social Media". In *Synthesis Lectures on Human Language Technologies*. Edit by Graeme Hirst of the University of Toronto.
20. Furht, B. 2010. Handbook of Social Network Technologies and Applications. Springer.
21. Gokcesu, K. and Kozat, S. 2017. A Rate Optimal Switching Bandit Algorithm. In *Proceedings of 25th European Signal Processing Conference (EUSIPCO)*.
22. Goldstein, M., Uchida, S. 2016. A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data. In *PLOS ONE*. https://doi.org/10.1371/journal.pone.0152173.
23. Google Privacy Notice. 2017. https://support.google.com/mail/answer/6603?hl=en.

24. Hawkins, D. M. 1980. *Identification of outliers*. Chapman and Hall, 1980
25. Huang, G., Song, S., and You, K. 2015. In *Neural Networks*. Trends in extreme learning machines: a review.
26. Hui S. K., Inman J, Huang Y., and Suher J. 2013. "The Effect of In-Store Travel Distance on Unplanned Spending: Applications to Mobile Promotion trategies," *Journal of Marketing*, 77 (March), 1–16.
27. Hurley, M., Adebayo, J. 2016. "Credit Scoring in the Era of Big Data". In *Yale Journal of Law and Technology*, volume 18, issue 1, article 5.
28. IMS Institute for Healthcare Informatics. 2015. "Patient Adoption of mHealth: Use, Evidence and Remaining Barriers to Mainstream Acceptance," Sept. 2015, http://www.imshealth.com/files/web/IMSH%20Institute/Reports/Patient%20Adoption%20of%20mHealth/IIHI_Patient_Adoption_of_mHealth.pdf
29. Kalman, R. E. 1960. A New Approach to Linear Filtering and Prediction Problems. In *Transaction of the ASME—Journal of Basic Engineering*, pp. 35-45. March 1960.
30. Katehakis, M. N. and Veinott, A. F. 1987. "The Multi-Armed Bandit Problem: Decomposition and Computation". *Mathematics of Operations Research. **12** (2): 262–268. doi:10.1287/moor.12.2.262*
31. Kramera, A., Guilloryb, J., Hancock, J. 2014. Experimental evidence of massive-scale emotional contagion through social networks. In *Proceedings of the National Academy of Sciences* (*PNAS),* published online before print June 2, 2014, doi:10.1073/pnas.1320040111 PNAS June 17, 2014 vol. 111 no. 248788-8790.
32. LeCun, Y., Bengio, Y., and Hinton, G. 2015. Deep Learning. In *Nature*, volume 521, 2015.

33. LiKamWa R, Liu Y, Lane ND, Zhong L. 2013. Moodscope: Building a mood sensor from smartphone usage patterns. In Proceeding of the 11th annual international conference on Mobile systems, applications, and services; ACM: 2013:389-402.
34. Lintean, M., and Rus, V. 2012. Measuring Semantic Similarity in Short Texts through Greedy Pairing and Word Semantics. In *Proceedings of the Twenty-Fifth International Florida Artificial Intelligence Research Society Conference*. 2012.
35. Lockyer, S. 2004. "Coding Qualitative Data." In *The Sage Encyclopedia of Social Science Research Methods*, Edited by Michael S. Lewis-Beck, Alan Bryman, and Timothy Futing Liao, v. 1, 137-138. Thousand Oaks, Calif.: Sage, 2004.
36. Luo, Xueming, Michelle Andrews, Zheng Fang and Chee Wei Phang. 2014. "Mobile Targeting". In *Management Science*, 60 (7), 1738–56.
37. Lyons, S. M., Alizadeh, E., Mannheimer, J., Schuamberg, K., Castle, J., Schroder, B., Turk, P., Thamm, D., Prasad, A. 2016. Changes in cell shape are correlated with metastatic potential in murine and human osteosarcomas. In *Biology Open*.
38. Matz, S. C., and Oded Netzer, O. 2017. "Using Big Data as a window into consumers' psychology". In *Current Opinion In Behavioral Sciences*, Elsevier, 18: 7-12.
39. McAfee A., Brynjolfsson E., Thomas H. Davenport T. H., D.J. Patil D. J. and Barton D. 2012. "Big Data: The Management Revolution," *Harvard Business Review*, 90 (10), 61–7.

40. Miller, G. and Charles, W. 1991. Contextual correlates of semantic similarity. In *Language and Cognitive Processes*.
41. Montgomery K, Chester J, Nixon L, Levy L, and Dorfman L. 2017. Big Data and the transformation of food and beverage marketing: undermining efforts to reduce obesity? Critical Public Health, DOI: 10.1080/09581596.2017.1392483.
42. Pennington, J., Socher, R. and D. Manning, C. D. 2014. GloVe: Global Vectors for Word Representation. https://nlp.stanford.edu/pubs/glove.pdf.
43. Popescu, M. and Baruh, L. 2013. Digital Literacy and Privacy Self-Governance: A Value-Based Approach to Privacy in Big Data Ecosystems. Paper presented at the annual meeting of *the International Association for Media and Communication Research Conference*, Dublin, Ireland.
44. Qiu, X., Liu, P., and Huang, X. 2016. Recurrent Neural Network for Text Classification with Multi-Task Learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence* (IJCAI-16).
45. Ranga, M., Sharma, D. 2014. Influencer Marketing – A Marketing Tool in the Age of Social Media. In *Abhinav International Monthly Refereed Journal of Research in Management & Technology*. Volume 3, issue 8, August 2014.
46. Raz, J. 1986. The Morality of Freedom. Clarendon Press, June 12, 1986.
47. Rowley, J. 2007. The wisdom hierarchy: representations of the DIKW hierarchy. *Journal of Information Science*.
48. Russakovsky, O., Deng, J., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg A. C. and Fei-Fei, L. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015.

49. Saldana, J. 2009. The Coding Manual for Qualitative Researchers. Sage Publishing.
50. Sondergaard, P. 2015. Big Data Fades to the Algorithm Economy. Forbes. https://www.forbes.com/sites/gartnergroup/2015/08/14/big-data-fades-to-the-algorithm-economy/#20dc535251a3
51. Stoicescu, C. 2015. Big Data, the perfect instrument to study today's consumer behavior. In *Database Systems Journal* vol. VI, no. 3/2015.
52. Sudhir, K. 2016. "Editorial—The Exploration-Exploitation Tradeoff and Efficiency in Knowledge Production," Marketing Science, 35 (1), 1–9.
53. Swan, M. 2013. The Quantified Self. In *Mary Ann Liebert, Inc.* vol. 1 no. 2.
54. Teixeira T, Wedel M, Pieters R. 2012. Emotion-induced engagement in internet video advertisements. J. Mark. Res. 2012, 49:144-159.
55. Toften, K., Hammervoll, T. 2013. Niche marketing research: status and challenges. In *Marketing Intelligence & Planning*, Vol. 31 Iss 3 pp. 272 – 285.
56. Wedel M., and Kannan P. K. 2016. Marketing Analytics for Data-Rich Environments. *Journal of Marketing*: AMA/MSI Special Issue vol. 80 (November 2016), 97–121.
57. Welch, G. and Bishop, G. 2001. "An Introduction to the Kalman Filter", University of North Carolina at Chapel Hill Department of Computer Science, 2001. http://www.cs.unc.edu/~tracker/media/pdf/SIGGRAPH2001_CoursePack_08.pdf
58. X. Wu, X. Zhu, G.-Q. Wu, and W. Ding. 2014. "Data mining with big data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, pp. 97-107, Jan 2014.
59. Xie, Y., Chen, Z., Cheng, Y., Zhang, K., Agrawal, A., Liao, W., Choudhar, A. 2013. Detecting and Tracking Disease Outbreaks by Mining Social Media Data.

In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*.

60. Youyoua, W., Kosinskib M., and Stillwell D. 2014. Computer-based personality judgments are more accurate than those made by humans. In *Proceedings of the National Academy of Sciences of the United States of America*.
61. Yu, R., Qiu, H., Wen, Z., Lin, C., and Liu, Y. 2016. A Survey on Social Media Anomaly Detection. In *SIGKDD Explorations* Volume 18, Issue 1.

62. Yu, R., He, X., Liu, Y. 2010. GLAD: Group Anomaly Detection in Social Media Analysis. In *ACM Transactions on Knowledge Discovery from Data*, Vol. 9, No. 4, Article 3.
63. Zarsky, T. Z. 2004. Thinking outside the box: Considering transparency, anonymity, and pseudonymity as overall solutions to the problems of information privacy in the internet society," *University of Miami Law Review*, vol. 58, pp. 1301-1354, 2004.
64. Zeleny, M. 1987. Management support systems: towards integrated knowledge management. *Human Systems Management*.

APPENDIX A

Our publication in the field of processing raw data to extract more meaning is included in the section. In this study we work on classification and dimensionality reduction techniques such as random projections and also data pre-processing techniques such as corpus cleaning. Below, we provide an English version of our paper published in the Proceedings of The Signal Processing and Communication Application Conference (SIU), 2016 24th. http://ieeexplore.ieee.org/document/7496063/.

Computationally Highly Efficient Online Text Classification and Regression for Real Life Tweet Analysis

Ersin Yar
Bilkent University
Universiteler Mahallesi,
Bilkent, 06800 Ankara Turkey
eyar@ee.bilkent.edu.tr

Ibrahim Delibalta
Koc University
Rumeli Feneri Yolu,
Sariyer, 34450 Istanbul Turkey
ibrahim.delibalta@turktelekom.com.tr

Lemi Baruh
Koc University
Rumeli Feneri Yolu,
Sariyer, 34450 Istanbul Turkey
lbaruh@ku.edu.tr

Suleyman S. Kozat
Bilkent University
Universiteler Mahallesi,
Bilkent, 06800 Ankara Turkey
kozat@ee.bilkent.edu.tr

Abstract

In this paper, we study multi-class classification of tweets, where we introduce highly efficient dimensionality reduction techniques suitable for online processing of high dimensional feature vectors generated from freely-worded text. As for the real life case study, we work on tweets in the Turkish language, however, our methods are generic and can be used for other languages as clearly explained in the paper. Since we work on a real life application and the tweets are freely worded, we introduce text correction, normalization and root finding algorithms. Although text processing and classification are highly important due to many applications such as emotion recognition, advertisement selection, etc., online classification and regression algorithms over text are limited due to need for high dimensional vectors to represent natural text inputs. We overcome such limitations by showing that randomized projections and piecewise linear models can be efficiently leveraged to significantly reduce the computational cost for feature vector extraction from the tweets. Hence, we can perform multi-class tweet classification and regression in real time. We demonstrate our results over tweets collected from a real life case study where the tweets are freely-worded, e.g., with emoticons, shortened words, special characters, etc., and are unstructured. We implement several well-known machine learning algorithms as well as novel regression methods and demonstrate that we can significantly reduce the computational complexity with insignificant change in the classification and regression performance.

Introduction

Due to recent developments in Internet technologies, the amount of accessible text information has significantly increased with the contribution of forums, columns, blogs, and social media. Clearly, processing of this big data, extracting information, performing classification and regression can significantly contribute to commercial products or to social sciences. However, text-based analysis proves to be very challenging due the variability and irregularity of media for text shares, the rapid variation of user sharing habits, and the large volume of data to be processed. Although text processing and classification are highly important due to many applications such as emotion recognition, advertisement selection, etc., online classification and regression algorithms over text are limited due to need for high dimensional vectors to represent natural text inputs. Especially, the state of the art representations such as the N-grams that are widely used as feature vectors require millions of components for accurate results, which deem them impractical for real time processing for text data such as real time emotion classification or sentiment analysis. This problem is especially exacerbated for agglutinative morphological structured languages such as Turkish, Finnish and Hungarian. These special languages derive words using extensive suffixes usually from a single word. Hence, the dimension of the word space exponentially increase unlike the Anglo-Saxon vocabularies. Because of the fundamental differences of agglutinative languages i.e. extreme usage of suffixes, making NLP research based on those languages is much more difficult.

To this end, in this paper, we introduce highly novel and computationally efficient feature extraction methods that can be even used for agglutinative languages. We emphasize that our methods directly apply to English, however, we choose the Turkish language as the real life case study to demonstrate the versality of our approach. We construct online and offline algorithms for multi-class classification of tweets, where we introduce highly efficient dimensionality reduction techniques suitable for online processing of high dimensional feature vectors generated from freely-worded text. Since we work on a real life application and the tweets are freely worded, we introduce a preprocessing pipeline with text correction, normalization and root finding components. Note that these components are also essential for other languages. We then introduce methods to derive feature vectors corresponding to tweets, which can be efficiently processed by the subsequent machine learning algorithms. We accomplish this by showing that randomized projections and piecewise linear models can be efficiently leveraged to significantly reduce the computational cost for feature vector extraction from the tweets. Hence, we can perform multi-class tweet classification and regression in real time. We demonstrate that our methods increase the speed of text classification 10^2 times over the state-of-art methods such as PCA (Sebastiani 2002).

The organization of the paper is as follows. In Problem Definition Section we provide the problem. In Regression

and Classification on Freely Worded Tweets Section we introduce highly efficient dimensionality reduction techniques and piecewise linear models for online classification of high dimensional feature vectors obtained using our vector space model that is constructed after data preprocessing steps on our collected data. We illustrate the performance of the introduced algorithms in Simulations Section. In Conclusion Section we provide certain remarks.

Problem Definition

In this paper, we study multi class classification of tweets. For tweet analysis we introduce preprocessing techniques due to unstructured and freely worded tweets. We then obtain feature vectors by representing them in our vector space model. We introduce highly efficient dimensionality reduction techniques suitable for online processing of high dimensional feature vectors generated from tweets.

Regression and Classification on Freely Worded Tweets

In this section, we first present our case study and data collection procedure. We then introduce our data preprocessing steps since the tweets are freely worded. After preprocessing, we construct feature vectors using these tweets and then introduce our classification and regression methods in a real life case scenario.

Data Collection

The tweets in our database are gathered through a case study where 1440 tweets written in Turkish are collected from 168 different users between April 10th, 2013 and May 28th, 2013. There are at most 10 tweets from a single user. The tweets, whose contents can be related to anything, are freely worded and unstructured. There are 3 classes, i.e., a tweet fall into one of three categories, which are "No Statement(0)", "Specific(1)" and "General(2)". These categories reflect the level of statement about other people in a tweet. Tweets are manually labeled by human experts.

Data Preprocessing

Agglutinative morphological structure of languages such as Turkish, Finnish and Hungarian enables one to derive numerous words using derivational suffixes even from a single root (Oflazer 1994). Thus, dimension of the word space constructed as a collection of distinct words can be considerably large. Moreover, we observe that tweets are freely worded, unstructured and they are not typed correctly all the time. Same word can emerge in significantly different forms due to aforementioned issues. Therefore, we apply a number of data preprocessing techniques to interpret the tweets properly. Our methods are generic such that they can be applied to any languages.

To this end, we removed urls, links and location information as well as mentions in tweets. We also discarded retweets. There are some words encountered frequently in most of the sentences that do not carry importance in terms of providing thematic content. Hence, we used a list of common words to eliminate them. We also eliminate numbers and the words having sizes smaller than 3. We then apply text correction to correct first the unwanted characters and then to correct the words that are misspelled. As we mention earlier, in agglutinatively morphological languages words having similar meanings can have the same roots. To be able to represent these words in one form, we apply stemming to obtain the roots. After these operations, final form of the tweets are obtained. The process pipeline is explained in Figure 1.

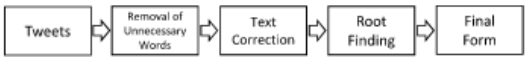


Figure 1: Processing pipeline of tweets.

Vector Space Model

We use a vector space model to represent tweets in our corpus. In tweet classification we define our vocabulary as the union of all distinct words used in the whole dataset and equate the dimension of our vector space to the size of the vocabulary. We represent the tweets in terms of N-grams (Jurafsky and Martin 2009), which is a representation technique consisting of N consecutive words. In this study, we use unigrams and bigrams to represent the tweets. An example of N-grams representation is given in Figure 2.

Figure 2: Unigram and bigram representation of a tweet.

In vector space model, we express each tweet as a vector where each component is related to a distinct word and assign a weight to that component. We use "TF-IDF" measure to calculate this weight (Rajaraman and Ullman 2011). "TF" means term frequency and we take it as the relative frequency of a word in a tweet. "IDF" means inverse document frequency and emphasizes how uncommon of a word is between other tweets .If a word does not appear in many tweets we increase its emphasis according to "IDF" measure. "TF" and "IDF" measures are found by

$$TF(f,t) = \frac{f}{t}, \tag{1}$$

$$IDF(f,d_t) = 1 + \frac{\log(|d_t|)}{|t|}, \tag{2}$$

where f is the current word, t is the corresponding tweet and d_t denotes tweet corpus. In our vector space model we use the multiplication of both term as the weight for a word in a tweet

$$TF - IDF_{(f,t,d_t)} = TF(f,t) * IDF(f,d_t). \tag{3}$$

At the end of these operations for each tweet t_t in tweet space $T = \{t_1, t_2, ..., t_n\}$ we derive a d dimensional feature

vector $t_t = [w_1, w_2, ..., w_d]^T$. Since text inputs are represented in high dimensional vectors we introduce two methods to represent them in low dimensional vectors to process efficiently, namely random projection and principal component analysis.

To this end, we present random projection as a simple and computationally efficient way to reduce the dimensionality of the data (Bingham and Mannila 2001). We project the original d dimensional vector to k-dimensional space by multiplying it with a random $k \times d$ dimensional matrix R. We construct this random matrix R chosing its entries randomly from the set $\{-1, 1\}$ or as samples from standard normal distribution.

Principal component analysis is another dimensionality reduction technique we employ. We map the high dimensional feature vectors to a lower dimensional space by multiplying them with a $k \times d$ transformation matrix whose rows are the eigenvectors corresponding to the k largest eigenvalues of the covariance matrix of data (Bingham and Mannila 2001).

We verify the validity of the transformations of feature spaces from high dimension to low dimension using following lemma.

Johnson Lindenstrauss lemma: *For any $0 < \epsilon < 1$ and any integer n, let k be a positive integer such that*

$$k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \ln(n)$$

Then for any set V of points in $\mathbb{R}^d$, there is a map $f : \mathbb{R}^d \rightarrow \mathbb{R}^k$ such that $\forall u, v$

$$(1 - \epsilon)\|u - v\|_2 \leq \|u - v\|_2 \leq (1 + \epsilon)\|u - v\|_2 . \quad (4)$$

Using the result of Johnson Lindenstrauss lemma (Dasgupta and Gupta 1999) we show that we can transform points from a high-dimensional space to a lower dimensional space in such a way that the distances between the points remain approximately same (Johnson and Lindenstrauss 1984).

Classification

We define automatic tweet classification as the process of identifying the class which a tweet belongs to. There is a space containing tweets $T = \{t_1, t_2, ..., t_n\}$, where each tweet t_t is represented by a d dimensional vector $t_t = [w_1, w_2, ..., w_d]^T$, where each w_k is the weight of term k in tweet t_t and there is a fixed set of classes $C = \{c_1, c_2, ..., c_C\}$. Our goal is to build a classification function matching tweets to their classes.

We carry out classification in two parts. In the first part we perform offline classification where we use all the data available. In the second part we introduce online classification of tweets by using them sequentially.

Offline Classification There are many types of algorithms used in text categorization (Sebastiani 2002). In this study, we use the following classifiers

- Support Vector Machines
- K-Nearest Neighbors
- Decision Trees
- Logistic Regression

In this part, we employ classification algorithms given above along with two different dimensionality reduction techniques, namely random projection and principal component analysis. We give the results in simulations section.

Online Classification In this part, we use a piecewise linear model (Vanli and Kozat 2014) to represent the relationship between features vectors and class labels. We construct this piecewise linear model combining seperate linear models trained in disjoint regions that are generated by partitioning d dimensional feature space using seperator functions. Our approach is adaptive in the sense that at each instance both model parameters and seperator function parameters are updated. In other words, we adaptively train model parameters and seperator function parameters to minimize the final regression error. We point out that as we sequentially classify tweets both model and seperator function parameters are adjusted such that space partitioning characterizes the structure of the data better and piecewise linear model predicts the corresponding class more accurately. In order to obtain satisfactory results parameter tuning should be done carefully. In Figure 3 we indicate a sample partitioning of two dimensional feature space into 4 disjoint regions.

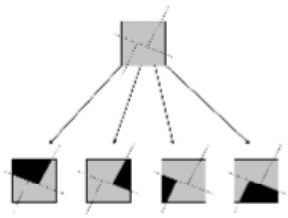


Figure 3: Sample partitioning of a two dimensional feature space into 4 disjoint regions.

Simulations

In this section, we demonstrate the performances of the algorithms. The dimensions of the feature vectors for unigrams and bigrams are 2511 and 6139, respectively. We reduce each of these dimensions to 125 and 250 applying different dimensionality reduction techniques. We obtain the accuracy values for classification algorithms by optimizing their parameters over grid search using 10-fold cross validation.

We point out that the accuracy values obtained using low dimensional feature vectors are comparably smaller than the accuracy values obtained without applying dimensionality reduction. This small loss comes with gain in computational complexity. For instance, logistic regression classifier utilizing random projection executes at least 100 times faster than the standard logistic regression classifier.

The results are given in Table 1 and in Table 2 for unigrams and bigrams, respectively. In Table 3 computational complexities of classification algorithms are given (Bottou and Lin 2007; Arya et al. 1994; Bingham and Mannila 2001).

Dim. Reduction \ Classifier	SVM	KNN	DT	Log. Reg.
No Reduction	0.582	0.594	0.454	0.638
PCA_{125}	0.573	0.565	0.525	0.598
PCA_{250}	0.580	0.585	0.502	0.599
$RP_{-1,1_{125}}$	0.572	0.554	0.489	0.468
$RP_{-1,1_{250}}$	0.571	0.566	0.498	0.438
$RP_{Gaussian_{125}}$	0.569	0.567	0.484	0.488
$RP_{Gaussian_{250}}$	0.574	0.573	0.501	0.471

Table 1: Accuracy values of classification algorithms for different dimensionality reduction techniques for unigrams.

Dim. Reduction \ Classifier	SVM	KNN	DT	Log. Reg.
No Reduction	0.575	0.564	0.510	0.573
PCA_{125}	0.227	0.523	0.393	0.562
PCA_{250}	0.191	0.541	0.411	0.571
$RP_{-1,1_{125}}$	0.572	0.528	0.494	0.472
$RP_{-1,1_{250}}$	0.570	0.563	0.498	0.436
$RP_{Gaussian_{125}}$	0.568	0.532	0.499	0.505
$RP_{Gaussian_{250}}$	0.570	0.562	0.485	0.430

Table 2: Accuracy values of classification algorithms for different dimensionality reduction techniques for bigrams.

Algorithm	Computational Complexity
SVM	$O(n^3)$
SVM with PCA	$O(n^3)$
SVM with RP	$O(n^3)$
KNN	$O(nd)$
KNN with PCA	$O(nd)$
KNN with RP	$O(nk)$
DT	$O(dn^2 \log(n))$
DT with PCA	$O(kn^2 \log(n))$
DT with RP	$O(dn^2 \log(n))$
Log. Reg.	$O(nd^2)$
Log. Reg. with PCA	$O(nk^2) + O(nd)$
Log. Reg. with RP	$O(nk^2)$

Table 3: Comparison of the computational complexities of classification algorithms. In the table, n represents the number of training instances, d represents regular dimension, k represents reduced dimension.

For online classification, we illustrate the performance of our algorithm having 1, 2 and 4 disjoint regions with respect to the truncated Volterra filter (Schetzen 1980). In Figure 4 we provide the time accumulated regression errors for each of them averaged over 10 trials. We emphasize that as the number of regions increase error value decreases and the performance of algorithm with 4 regions is comparable to the performance of Volterra filter.

Conclusion

We study multi class classification of tweets where we present preprocessing techniques since the tweets are freely worded. We construct feature vectors from tweets using our vector space model. Since text inputs are represented with high dimensional vectors we introduce highly efficient dimensionality reduction techniques. We show that we can significantly reduce the computational complexity with insignificant change in classification performance. We also present piecewise linear models suitable for online processing of tweets. We demonstrate their performance over minimization of time accumulated regression error.

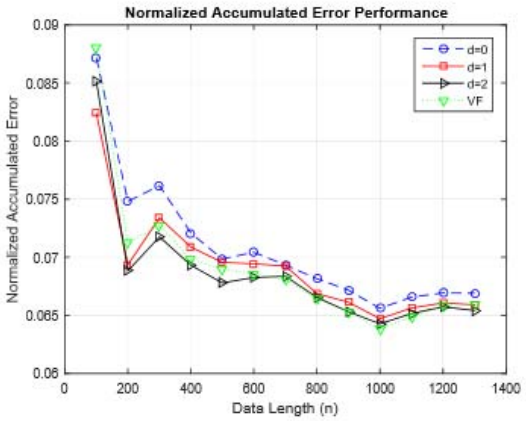


Figure 4: Normalized Accumulated Error Performance.

References

Arya, S.; Mount, D. M.; Netanyahu, N. S.; Silverman, R.; and Wu, A. Y. 1994. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. In *ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS*, 573–582.

Bingham, E., and Mannila, H. 2001. Random projection in dimensionality reduction: Applications to image and text data. In *in Knowledge Discovery and Data Mining*, 245–250. ACM Press.

Bottou, L., and Lin, C.-J. 2007. Support vector machine solvers. In Bottou, L.; Chapelle, O.; DeCoste, D.; and Weston, J., eds., *Large Scale Kernel Machines*. Cambridge, MA.: MIT Press. 301–320.

Dasgupta, S., and Gupta, A. 1999. An elementary proof of the johnson-lindenstrauss lemma. Technical report.

Johnson, W. B., and Lindenstrauss, J. 1984. Extensions of lipschitz mappings into a hilbert space. In *Contemporary Mathematics*, volume 26, 189206. Providence, RI: American Mathematical Society.

Jurafsky, D., and Martin, J. H. 2009. *SPEECH and LANGUAGE PROCESSING An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition Second Edition*. Prentice Hall.

Oflazer, K. 1994. Two-level description of turkish morphology. *Literary and Linguistic Computing* 9(2):137–148.

Rajaraman, A., and Ullman, J. D. 2011. *"Data Mining" Mining of Massive Datasets*. Cambridge University Press.

Schetzen, M. 1980. *The Volterra and Wiener Theories of Nonlinear Systems*. NJ: John Wiley & Sons.

Sebastiani, F. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.* 34(1):1–47.

Vanli, N., and Kozat, S. 2014. A comprehensive approach to universal piecewise nonlinear regression based on trees. *Signal Processing, IEEE Transactions on* 62(20):5471–5486.

APPENDIX B

Online anomaly detection is key in many of the applications covered in this chapter. As part of our thesis study, we introduce an online anomaly detection algorithm, which processes data in a sequential manner. The algorithm observes new data, produces its decision and then adaptively updates all its parameters to enhance its performance. The algorithm mainly works in an unsupervised manner since in most real life applications labeling the data is impractical and costly. Below, we provide a copy of our paper

published in IEEE Signal Processing Letters (Volume: 23, Issue: 12, Dec. 2016). http://ieeexplore.ieee.org/document/7727946/.

Online Anomaly Detection Using Nested Decision Trees

Ibrahim Delibalta, Kaan Gokcesu, Mustafa Simsek, Lemi Baruh, Suleyman S. Kozat, *Senior Member, IEEE*

***Abstract*—We introduce an online anomaly detection algorithm, which process data in a sequential manner. At each time, the algorithm observes a new data, produces its decision and then adaptively updates all its parameters to enhance its performance. The algorithm mainly works in an unsupervised manner since in most real life applications labeling the data is impractical. However, if feedback is available, the algorithm uses it for better adaptation. The algorithm has two stages. In the first stage, it constructs a probability distribution function (pdf) or a score function to model the underlying nominal distribution (if it is present) or to fit to the observed data. In the second stage, this pdf is used to score the newly observed data to provide the final decision. The decision is given after the well-known thresholding. We construct the pdf using a highly versatile and completely adaptive nested decision trees. Soft nested decision trees are used to partition the observation space in an hierarchical manner. We adaptively optimize every component of the tree including decision regions, probabilistic models at each node as well as the overall structure based on the sequential performance. This extensive in-time adaptation provides strong modeling capabilities, however, may cause overfitting. We mitigate overfitting issues by using the nodes of the tree to produce several subtrees from coarser models to the full extend, which are then adaptively combined to avoid overfitting. Due to such combination and adaptation, our algorithm significantly outperforms the state-of-the-art methods in the benchmark real life data sets.**

I. Introduction

We introduce an online anomaly detection algorithm that works on sequentially observed data [1], [2]. At each time, the algorithm decides whether the newly observed data is anomalous or not, and then updates all its internal parameters. We mainly work in an unsupervised manner since in most real life applications labeling the data is usually impractical [2]. However, if such labeling is present, then we use this information to improve adaptation. The algorithm has two stages. In the first stage we assign a sequential probability (or score) to the newly observed data based on our previous observations. Based on this assigned probability, we decide whether the newly observed data is anomalous or not. This decision is formed by comparing the assigned probability with a threshold [1], [2]. To assign sequential probabilities, we use highly versatile soft decision trees [3]–[5], where we adaptively learn every component of the tree including decision regions, probabilistic models at each node as well as the overall structure based on the sequential performance [3].

This work is supported in part by Turkish Academy of Sciences Outstanding Researcher Programme. S. S. Kozat, K. Gokcesu and M. Simsek are with the Electrical and Electronics Engineering Department, Bilkent University, Ankara, Turkey, email: {kozat,kaan.gokcesu,mustafa.simsek}@ee.bilkent.edu.tr, tel: +90 312 290 2336, fax: +90 312 290 2333. I. Delibalta and L. Baruh are with Design, Technology and Society Program, Koc University, Istanbul, Turkey, email: {idelibalta,lbaruh}@ku.edu.tr.

Two-stage anomaly detection methods especially in unsupervised and/or adversarial settings are extensively studied in the literature [2], [6], [7]. Although there exist several nonparametric approaches to model the nominal distribution, especially in adversarial settings [1], [8], parametric models offer significant advantages such as quick convergence and high accuracy [8]. However, the parametric models suffer enormously if the assumed model does not match to the underlying true model (if such a true model exists) or it is not rich enough to accurately capture the salient nature of the data [2]. Even if the assumed model correctly fits to certain parts, we may still face underfitting or overfitting issues since these algorithms usually work in highly nonstationary real life environments.

To this end, we first introduce a highly adaptive and efficient decision tree, which softly partitions the observation space. To boost modeling capabilities, we assign to each terminal leaf-node a pdf from an exponential-family of distributions, where parameters of these pdfs are sequentially learned. The boundaries of the regions assigned to each leaf are soft such that they are also updated based on the performance. In this form, the tree structure is similar to Self Organizing Maps (SOM)s or Gaussian Mixture Models (GMM)s [9], [10], where learning the partitions (or boundaries) corresponds to learning the apriori weights of the Gaussian pdfs in the GMMs (or SOMs). It is well-known that the mixture models provide high modeling power [1], [2], [11], however, may overfit due to excessive number of leaves, i.e., Gaussians in the mixture. Hence, to avoid overfitting or committing to a fixed decision tree, we go one step further and use all the nodes of the tree in addition to the leaf nodes such that each node is assigned to a particular region with its own pdf. This structure effectively constructs several subtrees with different depths on the original tree, which are then adaptively combined to maximize the overall performance. Since we adaptively merge both coarser and richer models, our algorithm avoids overfitting issues while preserving the modeling power [4].

II. Anomaly Detection Framework

A. Two-stage Processing

Here[1], we sequentially receive $\{\boldsymbol{x}_t\}_{t\geq 1}$, where $\boldsymbol{x}_t \in \mathbb{R}^m$, and seek to find whether the received data is anomalous or not at each time t. To produce the decision, we sequentially construct a pdf $p_t(\cdot)$ (or a scoring function to be rigorous) using $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}\}$ to model the underlying nominal distribution

[1]We represent vectors (matrices) by bold lower (upper) case letters. For a matrix $\boldsymbol{A}$ (or a vector $\boldsymbol{a}$), $\boldsymbol{A}^T$ is the transpose. $\|\boldsymbol{a}\|$ is the Euclidean norm. For notational simplicity we work with real data and all vectors are column vectors.

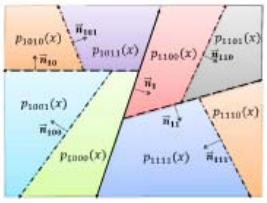


Fig. 1: Hard Decision Boundaries for a Depth-3 Decision Tree

(or to fit to the observed data if no such nominal distribution exists). Then, at each time t, based on the constructed distribution $p_t(\cdot)$, we score $\boldsymbol{x}_t$ as $p_t(\boldsymbol{x}_t)$ and produce our decision $\hat{d}_t$. We produce the final decision using thresholding [1] (where such an approach is optimal minimizing the Type-1 error in certain settings [12]), i.e., if

$$p_t(\boldsymbol{x}_t) \geq \tau_t, \tag{1}$$

then $\hat{d}_t = 0$ (not anamolous), otherwise $\hat{d}_t = 1$ (anamolous) for some time varying threshold τ_t. Then, this decision is compared with the correct result d_t if available, otherwise we work in an unsupervised manner [6].

To construct the sequential distribution, we use decision trees. A decision tree is a hierarchical structure composed of both internal nodes and terminal nodes, i.e., the leaf nodes. Unlike [3], [4], [13], we do not require the tree to be complete. After we observe $\boldsymbol{x}_t$, each node η produces its probability or score as

$$f_\eta(\boldsymbol{x}_t) = \begin{cases} p_\eta(\boldsymbol{x}_t), & \text{if } \eta \text{ is a leaf,} \\ f_{\eta l}(\boldsymbol{x}_t), & \sigma_\eta(\boldsymbol{x}_t) \geq 0 \text{ go to right child,} \\ f_{\eta r}(\boldsymbol{x}_t) & \sigma_\eta(\boldsymbol{x}_t) < 0 \text{ go to left child,} \end{cases} \tag{2}$$

where $\sigma_\eta(\cdot)$ is the hard decision boundary of the node η as shown in Fig. 1. In this letter, we use linear separating hyper planes for decision boundaries such that $\sigma_\eta(\cdot)$ is given as $\sigma_\eta(\boldsymbol{x}_t) = \boldsymbol{n}_\eta^T[\boldsymbol{x}_t; 1]$, where $\boldsymbol{n}_\eta$ is the normal vector of the separating hyper plane and we extend $\boldsymbol{x}_t$ as $[\boldsymbol{x}_t; 1]$ to include the bias term for a compact notation. Our approach is generic such that one can also use nonlinear separation boundaries, however, we use linear boundaries to avoid overfitting. Here, $f_{\eta l}(\boldsymbol{x}_t)$ (or $f_{\eta r}(\boldsymbol{x}_t)$) is the score of the left hand (or the right hand) child node. Each leaf node η is assigned a pdf from an exponential family of distributions as

$$p_\eta(\boldsymbol{x}_t) = \exp(\boldsymbol{\theta}_\eta^T \boldsymbol{x}_t - G(\boldsymbol{\theta}_\eta)) P_o(\boldsymbol{x}_t),$$

where $\boldsymbol{\theta}_\eta$ is from some convex set, $G(\boldsymbol{\theta}_\eta)$ is sufficient statistics and $P_o(\boldsymbol{x}_t)$ is for normalization [9]. For each $\boldsymbol{x}_t$, the final probability is given by

$$p(\boldsymbol{x}_t) = f_r(\boldsymbol{x}_t),$$

which is the score of the root node and the recursion starts from the root node until we reach to one of the leaves.

As the first extension to the basic decision tree, we use soft partitioning [3] similar to the SOM models and set

$$\sigma_\eta(\boldsymbol{x}_t) = 1/\left[1 + \exp\left(-\boldsymbol{n}_\eta^T \boldsymbol{x}_t\right)\right], \tag{3}$$

where we use $\boldsymbol{x}_t$ instead of $[\boldsymbol{x}_t; 1]$ with an abuse of notation. Then, for each node, we set

$$f_\eta(\boldsymbol{x}_t) = \begin{cases} p_\eta(\boldsymbol{x}_t), & \text{if } \eta \text{ is a leaf,} \\ \sigma_\eta(\boldsymbol{x}_t) f_{\eta l}(\boldsymbol{x}_t) + (1 - \sigma_\eta(\boldsymbol{x}_t)) f_{\eta r}(\boldsymbol{x}_t) & \text{otherwise.} \end{cases}$$

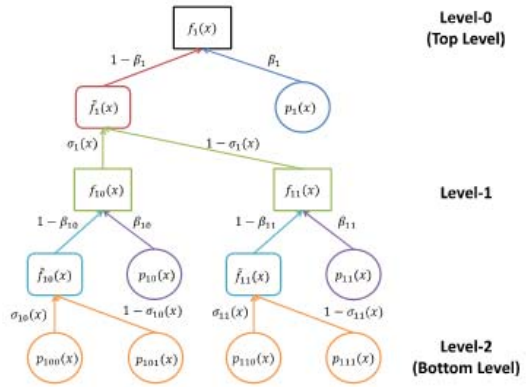


Fig. 2: Nested Combination Structure for a Depth-2 Decision Tree

For the soft decision tree, the calculation starts from the leaf nodes such that all the leaf nodes contribute to the final pdf, unlike a hard decision tree (2). We start from bottom of the tree and proceed to the top node, i.e., to the root node, as in Fig. 2 to get the final score.

As the second and final extension, we assign pdfs from exponential-family distributions to all nodes of tree, including both the terminal and internal nodes. In the previous cases, either hard or soft, only the leaves of the tree, i.e., the finest and the most detailed structure of the tree, were used to partition the space of observations or assign scores. Here, by assigning pdf to the internal nodes, we also represent much coarser models or partitions of the observation space. After this assignment of pdfs, we define for each node

$$f_\eta(\boldsymbol{x}_t) = \begin{cases} p_\eta(\boldsymbol{x}_t), & \text{if } \eta \text{ is a leaf,} \\ \beta_\eta p_\eta(\boldsymbol{x}_t) + (1 - \beta_\eta) \times & \text{else} \\ [\sigma_\eta(\boldsymbol{x}_t) f_{\eta l}(\boldsymbol{x}_t) + (1 - \sigma_\eta(\boldsymbol{x}_t)) f_{\eta r}(\boldsymbol{x}_t)] & \end{cases} \tag{4}$$

where $0 \leq \beta_\eta \leq 1$ for all η. Since, we use the stochastic gradient descent for optimization [14], we satisfy this constraint by reparametrizing the mixture weight as

$$\beta_\eta = 1/\left[1 + \exp\left(-\alpha_\eta\right)\right], \tag{5}$$

where $\alpha_\eta \in \mathbb{R}$. The final probability is given by $p(\boldsymbol{x}_t) = f_r(\boldsymbol{x}_t)$. Here, after we observe $\boldsymbol{x}_t$, the calculation of the final probability starts from the bottom of the tree, where not only each leaf but also all the internal nodes contribute to the final probability.

In this form, for a decision tree of depth d, we have $\eta = 2^d - 1$ nodes including both internal and terminal nodes. For each internal node, we have a tupple $\{\beta_\eta, \sigma_\eta, \boldsymbol{\theta}_\eta\}$ (or equivalently $\{\alpha_\eta, \boldsymbol{n}_\eta, \boldsymbol{\theta}_\eta\}$), the mixture coefficient that merges score of the node with its children's scores, the soft partition parameter that is similar to the apriori weights assigned to each child and finally pdf parameters assigned to the node. For terminal nodes, we only have one parameter set $\{\boldsymbol{\theta}_\eta\}$.

Remark: In [5] and [4], the combination weights are fixed in time and equal to $\beta_\eta = 1/2$ for all nodes η. In [13], these weights are again fixed in time, however, set to a desired

Algorithm 1 Online Anomaly Detection Algorithm

Initialize tree and all parameters $\alpha_{\eta,1}$, $\boldsymbol{n}_{\eta,1}$, τ_1
for $t = 1$ **to** ... **do**
  Receive observation, $\boldsymbol{x}_t$
  for all nodes **do**
    Calculate $\sigma_{\eta,t}$, $\beta_{\eta,t}$, $f_\eta(\boldsymbol{x}_t)$ according to (3), (5), (4)
  end for
  $p_t(\boldsymbol{x}_t) = f_1(\boldsymbol{x}_t)$
  $\hat{d}_t = \max\{0, \operatorname{sgn}(p_t(\boldsymbol{x}_t) - \tau_t)\}$
  if ($d_t = 0$) **or** (d_t is **not** avaliable **and** $\hat{d}_t = 0$) **then**
    for all nodes **do**
      Calculate $\delta_{\eta,t}$ according to (10), (11), (12)
      Calculate $\nabla_{\boldsymbol{\theta}_\eta} l_t(\boldsymbol{x}_t)$, $\partial l_t(\boldsymbol{x}_t)/\partial\alpha_\eta$, $\nabla_{\boldsymbol{n}_\eta} l_t(\boldsymbol{x}_t)$ according to (7), (8), (9)
      Update parameters $\boldsymbol{\theta}_{\eta,t+1}$, $\alpha_{\eta,t+1}$, $\boldsymbol{n}_{\eta,t+1}$ according to (13), (14), (15)
    end for
  end if
  if d_t is available **then**
    $\tilde{d}_t = (1 + \exp(-(\tau_t - p_t(\boldsymbol{x}_t))))^{-1}$
    $\tau_{t+1} = \tau_t - \mu_t(d_t - \tilde{d}_t)\tilde{d}_t(1 - \tilde{d}_t)$
  end if
end for

apriori values based on the user input. In [3], in a regression framework, these weights are unconstraint, i.e., $\beta_\eta \in \mathbb{R}$, can even take non-negative values and adapted in time to minimize the final regression error. Here, inspired from [10] and since we work with probabilities, we constraint these weights to the unit simplex, i.e., $\beta_\eta \in [0, 1]$.

Remark: Although the soft decision tree, which only uses the leaf nodes, i.e., the finest models, has the highest modeling power, there is no guarantee that it would provide the best performance in applications involving online or sequential data. The modeling power comes with increase in the number of parameters that must be sequentially learned. Hence, when there are limited data or the data is highly nonstationary, coarser models, i.e., subtrees of the full tree, may perform better. By adaptively combining both the coarser and finer models we retain the breadth of the finest model while alleviating the overfitting problems associated with too powerful modeling.

III. The Online Algorithm

In this section, we train our algorithm in an online manner. We have two cases. In the first case, the true label is not present at time t. Then, we decide the label of the data based on (1). If $\hat{d}_t = 0$, then the new observation $\boldsymbol{x}_t$ can be used to update $p_t(\cdot)$. If $\hat{d}_t = 1$, then we discard it. In the second case, we have the correct label d_t. If $d_t = 0$, then we naturally update $p_t(\cdot)$. If $d_t = 1$, no update is necessary on $p_t(\cdot)$. When we have d_t, we also update the threshold.

When $\hat{d}_t = 0$ or $d_t = 0$ (if it is available), we update the $p_t(\cdot)$. In this case, we measure the performance of our sequential probability assignment using the most obvious loss measure [9], which is the negative log probability

$$l_t(\boldsymbol{x}_t) = -\ln p_t(\boldsymbol{x}_t). \tag{6}$$

To optimize and learn the system parameters, we use the Stochastic Gradient Descent (SGD) algorithm [14]. The SGD recursion enjoys deterministic bounds in sequential convex optimization problems [16]. The pdf estimation problem is convex under the loss (6) when we have only one exponential distribution. However, due to the sigmoid nonlinearities in (3) and (5), the underlying problem is not convex.

To use the SGD, we need to calculate the gradient of the final loss with respect to all parameters. We observe that the soft decision structure shown in Fig. 2 is similar to a neural network architecture where the bottom of the tree corresponds to the input layer with $p_\eta(\boldsymbol{x}_t)$'s as inputs, i.e., the input layer has 2^d neurons, and the output layer corresponds to the root of the tree, where the final output of the system is given by $p_t(\boldsymbol{x}_t) = f_\eta(\boldsymbol{x}_t)$. In this sense, β_η's correspond to gating functions and σ_η's correspond to combination weights at each layer [9]. Hence, to calculate the gradients at each level, we can use the well-known back-propagation algorithm [9], which is basically the chain rule. The back-propagation algorithm proceeds as follows. When $\boldsymbol{x}_t$ arrives, we start from the leaf nodes, i.e., from the input layer, and calculate all the terms, $\sigma_{\eta,t}$, $\beta_{\eta,t}$, $p_{\eta,t}(\boldsymbol{x}_t)$ and $f_{\eta,t}(\boldsymbol{x}_t)$. This is the "forward-propagation" [9]. We are now ready for the backward propagation. In the back-propagation step, we start from the top, i.e., from the root node, and calculate step by step the gradient until we reach to the bottom nodes, i.e., to the leaves. For any internal η including the root node, using the chain rule, we have

$$\nabla_{\boldsymbol{\theta}_\eta} l_t(\boldsymbol{x}_t) = \delta_{\eta,t}\beta_{\eta,t}p_\eta(\boldsymbol{x}_t)\left(\boldsymbol{x}_t - \nabla_{\boldsymbol{\theta}_\eta} G(\boldsymbol{\theta}_{\eta,t})\right), \tag{7}$$

$$\partial l_t(\boldsymbol{x}_t)/\partial\alpha_\eta = \delta_{\eta,t}(1-\beta_{\eta,t})\beta_{\eta,t}\Big(p_\eta(\boldsymbol{x}_t) - [\sigma_\eta(\boldsymbol{x}_t)f_{\eta l}(\boldsymbol{x}_t) + (1-\sigma_\eta(\boldsymbol{x}_t))f_{\eta r}(\boldsymbol{x}_t)]\Big), \tag{8}$$

$$\nabla_{\boldsymbol{n}_\eta} l_t(\boldsymbol{x}_t) = \delta_{\eta,t}(1-\beta_{\eta,t})\times (1-\sigma_{\eta,t}(\boldsymbol{x}_t))\sigma_{\eta,t}(\boldsymbol{x}_t)\left(f_{\eta l,t}(\boldsymbol{x}_t) - f_{\eta r,t}(\boldsymbol{x}_t)\right)\boldsymbol{x}_t, \tag{9}$$

where $\delta_{\eta,t} \triangleq \partial l_t(\boldsymbol{x}_t)/\partial f_\eta(\boldsymbol{x}_t)$. We calculate $\delta_{\eta,t}$ using the back-propagation. For the root node, by using (6), we get

$$\delta_{\eta,t} = -1/p_t(\boldsymbol{x}_t). \tag{10}$$

Since we start from the top node, we back propagate to the lower nodes. For any internal node, we distinguish left and right children. For a node η, which is the left child of some $\tilde{\eta}$, we have

$$\delta_{\eta,t} = \partial l_t(\boldsymbol{x}_t)/\partial f_{\tilde{\eta}}(\boldsymbol{x}_t)\,(1-\beta_{\tilde{\eta},t})\sigma_{\tilde{\eta},t}(\boldsymbol{x}_t). \tag{11}$$

Similarly for a node η, which is the right child of $\tilde{\eta}$, we have

$$\delta_{\eta,t} = \partial l_t(\boldsymbol{x}_t)/\partial f_{\tilde{\eta}}(\boldsymbol{x}_t)\,(1-\beta_{\tilde{\eta},t})(1-\sigma_{\tilde{\eta},t}(\boldsymbol{x}_t)). \tag{12}$$

The recursion stops at the terminal leaf nodes. Then, we update the corresponding parameters using the SGD as

$$\boldsymbol{\theta}_{\eta,t+1} = \boldsymbol{\theta}_{\eta,t+1} - \mu_t\nabla_{\boldsymbol{\theta}_\eta} l_t(\boldsymbol{x}_t) \tag{13}$$

$$\alpha_{\eta,t+1} = \alpha_{\eta,t} - \mu_t\partial l_t(\boldsymbol{x}_t)/\partial\alpha_\eta \tag{14}$$

$$\boldsymbol{n}_{\eta,t+1} = \boldsymbol{n}_{\eta,t+1} - \mu_t\nabla_{\boldsymbol{n}_\eta} l_t(\boldsymbol{x}_t) \tag{15}$$

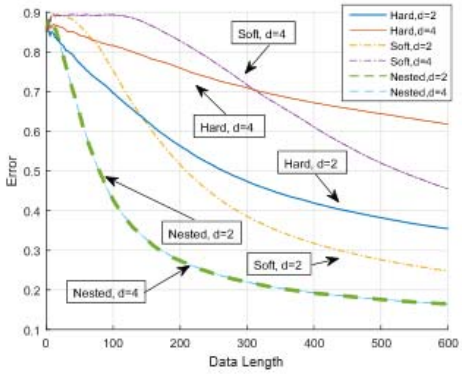
for some learning rate μ_t.

When we have the feedback, we train the threshold again using the SGD approach. As the error we use the square error, $(d_t - \hat{d}_t)^2$, and get

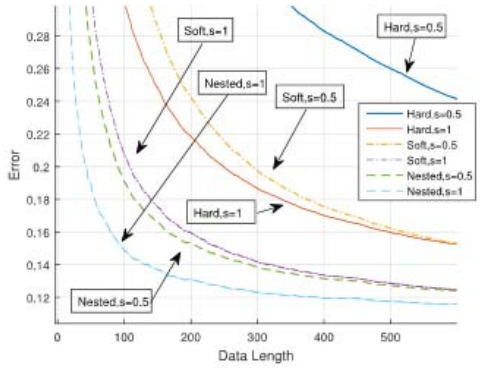
$$\tau_{t+1} = \tau_t - \mu_t(d_t - \hat{d}_t)\partial\hat{d}_t/\partial\tau. \tag{16}$$

Since $\hat{d}_t$ given in (1) is not differentiable, as widely done in the signal processing literature [14], we use

$$\tilde{d}_t = 1/\left[1 + \exp\left(-(\tau_t - p_t(\boldsymbol{x}_t))\right)\right].$$



(a) Time averaged anomaly detection error of hard, soft and nested depth $d = 2$ and $d = 4$ decision trees for ISE dataset [15] in a $s = 0.2$ feedback probability environment averaged over 100 trials.



(b) Time averaged anomaly detection error of hard, soft and nested depth $d = 3$ decision trees for ISE dataset [15] in a $s = 0.5$ and $s = 1$ (full feedback) feedback probability environment averaged over 100 trials.

Hence, (16) yields

$$\tau_{t+1} = \tau_t - \mu_t(d_t - \tilde{d}_t)\tilde{d}_t(1 - \tilde{d}_t). \tag{17}$$

This completes the full set of equations.

The complete algorithm is given in Algorithm 1.

IV. Experiments and Final Remarks

We use the Istanbul Stock Exchange (ISE) [15] dataset for real data benchmark purposes. Daily price data of nine stocks are downloaded from 'http://finance.yahoo.com' and 'http://imkb.gov.tr' between dates January 5, 2009 to February 22, 2011 and prices are converted to returns [15], yielding $\boldsymbol{x}_t \in \mathbb{R}^9$, $m = 9$, over 536 samples. We randomly add 64 (nearly 10 percent) anomalous samples to this dataset. The anomalous data are generated from a multivariate Gaussian process whose mean is the negative of the batch mean of the nominal data (with the same estimated covariance).

We run online anomaly detection algorithms using a hard decision tree, soft decision tree and nested decision tree. Our algorithm, i.e., the nested decision tree, combines the beliefs of all internal nodes as well as the terminal nodes. Soft decision tree is the first extension where the self combination weights of internal nodes are zero, i.e., $\beta_{\eta,t} = 0$ for all t and η. However, a soft decision tree still updates its boundaries since $\sigma_{\eta,t}(\boldsymbol{x}_t)$ is the sigmoid function. For the hard decision tree, we again have $\beta_{\eta,t} = 0$, however, $\sigma_{\eta,t}(\boldsymbol{x}_t)$ is the unit step function, i.e., the boundaries do not change. We use a depth 3 to train all algorithms, where the depth is arbitrarily set. We set $\mu = 1/\sqrt{t}$ [6] for all t to get a decreasing learning rate to compensate the nonstationarity of the data. We normalize each dimension of the feature vectors to $[-1, 1]$. We run a multivariate Gaussian density estimator in each node. Initially the self combination weight of each node $\beta_{\eta,1}$ is set to $1/2$ and $\tau_1 = 1$ for all algorithms. The boundaries are selected such that, the split at the first layer of the tree splits according to the first feature, i.e., whether it is greater or less than zero; the second layer splits according to the second feature, etc. All Gaussian density estimators are started from zero mean and identity covariance matrices. In Fig. 3a, we illustrate the average online anomaly detection performance of all algorithms using depth $d = 2$ and $d = 4$ trees in a $s = 0.2$ feedback probability environment, i.e., with s probability we provide the true label d_t at each time t. In Fig. 3b, we illustrate the average online anomaly detection performance of all algorithms using depth $d = 3$ trees in a $s = 0.5$ and $s = 1$ (full feedback) feedback probability environment. The plotted results are averaged over 100 independent trials. Since hard decision trees have less parameters to learn, they show better performance than soft trees at the beginning, but are quickly outperformed after a while. Nevertheless, both hard decision trees and soft decision trees are continuously outperformed by the nested decision trees. As shown in Fig. 3a, both hard decision and soft decision trees perform better with depth 2 trees, since they have less parameters to learn. However, the depth selection is not an issue for nested decision trees, since they also use the internal nodes for comparable performance. In Fig. 3b, we illustrate that higher feedback provides higher performance as expected. However, the algorithms dependence on feedback changes with the combination structure. Soft trees with $s = 0.5$ feedback show comparable performance to hard trees with $s = 1$ feedback. Similarly, nested trees with $s = 0.5$ feedback outperform soft trees with $s = 1$ feedback. Using nested trees mitigates overfitting and undertraining issues and thus outperforms other combination structures.

We introduced a highly versatile and effective online anomaly detection algorithm based on nested trees. Based on the sequential performance, we learn every component of the tree including decision regions, probabilistic models at each node as well as the overall structure. We mitigate overfitting issues by using all nodes of the tree to produce several subtrees from coarser models to the full extend, which are then adaptively combined to avoid overfitting.

References

[1] H. Ozkan, F. Ozkan, and S. S. Kozat, "Online anomaly detection under markov statistics with controllable type-i error," *IEEE Transactions on Signal Processing*, vol. 64, no. 6, pp. 1435–1445, 2016.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection : A survey," *ACM Computing Surveys*, vol. 11, pp. 1–72, 2009.

Appendix

[3] N. D. Vanli and S. S. Kozat, "A comprehensive approach to universal piecewise nonlinear regression based on trees," *IEEE Transactions on Signal Processing*, vol. 62, no. 20, pp. 5471–5486, Oct. 2014.

[4] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: basic properties," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 653–664, 1995.

[5] S. S. Kozat, A. C. Singer, and G. C. Zeitler, "Universal piecewise linear prediction via context trees," *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3730–3745, 2007.

[6] M. Raginsky, C. H. R. M. Willett, J. Silva, and R. F. Marcia, "Sequential anomaly detection in the presence of noise and limited feedback," *IEEE Transactions on Information Theory*, vol. 58, no. 8, pp. 5544–5562, 2012.

[7] C. Horn and R. M. Willett, "Online anomaly detection with expert system feedback in social networks," in *ICASSP*, 2011, pp. 1936–1939.

[8] H. Ozkan, F. Ozkan, I. Delibalta, and S. S. Kozat, "Efficient NP tests for anomaly detection over birth-death type DTMCs," *Journal ofSignal Processing Systems*, vol. 1, no. 1, pp. 1–10, June 2016.

[9] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. NJ: John Wiley & Sons, 2000.

[10] J. Arenas-Garcia, A. R. Figueiras-Vidal, and A. H. Sayed, "Mean-square performance of a convex combination of two adaptive filters," *IEEE Transactions on Signal Processing*, vol. 54, no. 3, pp. 1078–1090, 2006.

[11] O. J. J. Michel, A. O. Hero, and A.-E. Badel, "Tree-structured nonlinear signal modeling and prediction," *IEEE Transactions on Signal Processing*, vol. 47, no. 11, pp. 3027–3041, 1999.

[12] H. V. Poor, *An Introduction to Signal Detection and Estimation*. NJ: Springer, 1994.

[13] J. Suzuki, "A CTW scheme for some FSM models," in *IEEE International Symposium on Information Theory, Canada*, 1995, p. 389.

[14] A. H. Sayed, *Fundamentals of Adaptive Filtering*. NJ: John Wiley & Sons, 2003.

[15] O. Akbilgic, H. Bozdogan, and M. E. Balaban, "A novel hybrid rbf neural networks model as a forecaster," *Statistics and Computing*, vol. 24, no. 3, pp. 365–375, 2014. [Online]. Available: http://dx.doi.org/10.1007/s11222-013-9375-7

[16] K. Z. Azoury and M. K. Warmuth, "Relative loss bounds for on-line density estimation with the exponential family of distributions," *Machine Learning*, vol. 43, pp. 211–246, 2001.

APPENDIX C

A copy of our paper, "An Online Causal Inference Framework for Modeling and Designing Systems Involving User Preferences: A State-Space Approach," published in the *Hindawi Journal of Electrical and Computer Engineering*, vol. 2017, Article ID 1048385, 11 pages, 2017. doi:10.1155/2017/1048385, is provided in this section. https://www.hindawi.com/journals/jece/2017/1048385/.

An Online Causal Inference Framework for Modeling and Designing Systems Involving User Preferences: A State Space Approach

Ibrahim Delibalta, Koc University, Istanbul, idelibalta13@ku.edu.tr
Lemi Baruh, Koc University, Istanbul, lbaruh@ku.edu.tr
Suleyman Serdar Kozat, Bilkent University, Ankara, kozat@ee.bilkent.edu.tr

Abstract

In this paper, we provide a causal inference framework to model the effects of machine learning algorithms on user preferences. We then use this mathematical model to prove that the overall system can be tuned to alter those preferences in a desired manner. A user can be an online shopper or a social media user, exposed to digital interventions produced by machine learning algorithms. A user preference can be anything from inclination towards a product to a political party affiliation. Our framework uses a state space model to represent user preferences as latent system parameters which can only be observed indirectly via online user actions such as a purchase activity or social media status updates, shares, blogs or tweets. Based on these observations, machine learning algorithms produce digital interventions such as targeted advertisements or tweets. We model the effects of these interventions through a causal feedback loop, which alters the corresponding preferences of the user. We then introduce algorithms in order to estimate and later tune the user preferences to a particular desired form. We demonstrate the effectiveness of our algorithms through experiments in different scenarios.

Introduction

Recent innovations in communication technologies, coupled with the increased use of Internet and smartphones greatly enhanced institutions' ability to gather and process an enormous amount of information on individual users on Social Networks or consumers in different platforms (Gupta et al. 2014; Ruta 2014; Wang and Djuric 2015; Buttou and Le Cun 2005). Today, many sources of information from shares on Social Networks to blogs, from intelligent device activities to security camera recordings are easily collectable. Efficient and effective processing of this "big data" can significantly improve the quality of many real life applications or products, since this data can be used to accurately profile and then target particular users (Buttou and Bousquet 2007; Yan et al. 2009; Peng et al. 2013). In this sense, abundance of new sources of information and previously unimaginable ways of access to consumer data have the potential to substantially change the classical machine learning approaches that are tailored to extract information with rather limited access to data using relatively complex algorithms (Bottou et al. 2013; Subakana et al. 2014; Achbany et al. 2008; Jahrer et al. 2010).

Furthermore, unlike applications where the machine learning algorithms are used as mere tools to process and infer using the available data such as predicting the best movie for a particular user (Toscher et al. 2008), the new generation of machine learning systems employed by enormously large and powerful data companies and institutions have the potential to change the underlying problem framework, i.e., the user itself, by design (Bottou et al. 2013; Chan et al. 2010). Consider the Google search engine platform and its effects on user preferences. The Google search platform not only provides the most relevant search results, but also gathers information on users and provides well-tuned and targeted content (from carefully selected advertisements to specifically selected news) that may be used to change user behavior, inclinations or preferences (Epstein and Robertson 2015).

Online users are exposed to persuasive technologies, and are continually immersed in digital content and interventions in various forms such as advertisements, news feeds, and recommendations (Salah et al. 2011). User decisions and preferences are affected by these interventions (Zarsky 2004). We define a feedback framework in which these interventions can be selected in a systematic way to steer users in a desired manner. In Figure 1, we introduce "The Digital Feedback Loop" on which we base our model.

Figure 1: The Digital Feedback Loop

To this end, in this paper, we are particularly interested in the causal effects of machine learning algorithms on users (Wang et al. 2015a; Wang et al. 2015b). Specifically, we introduce causal feedback loops to accurately describe effects of machine learning algorithms on users in order to design more functional and effective machine learning systems (Wang et al. 2015b; Brodersen et al. 2015). We model the latent preferences and/or inclinations of a user, as an unknown state in a real life causal system and build novel algorithms to estimate and, then, alter this underlying unobservable state in an intentional and preferred manner. In particular, we model the underlying evolution of this state using a state space model, where the latent state is only observed through the behavior of the user such as his/her tweets, Facebook status shares. The internal state is causally affected by the outputs of the algorithm (or the actions of the company), which can be derived from the past observations on the user or outputs of the system. The purpose of the machine learning algorithm can be, for example, (i) to drive the internal system state towards a desired final state, e.g., try to change the opinion of the population towards a newly introduced product; (ii) to maximize some utility function associated with the system, e.g., entice the users to a new and more profitable product, or (iii) to minimize some regret associated with the disclosed information, e.g., minimize the effects of unknown system parameters. Alternatively, the machine learning system may try to achieve a combination of these objectives.

This problem framework readily models a wide range of real life applications and scenarios (Brodersen et al. 2015; Wang et al. 2015b). As an example, an advertiser may aim to direct the preferences of his/her target audience towards a desired product, by designing advertisement using data collected by consumer behavior surveys (Wang et al. 2015b). This framework is substantially different from the classical problem of targeted advertisement based on user profiling. In the case of targeted advertising, the goal is to match the best advertisement to the current user, based on the user's profile. Another part of the classical problem is to measure the true impact of an ad (a 'treatment' or an 'intervention' in the general case) and thus find its effectiveness to help the ad selection for the next time or the next user as well as for billing purposes. Here, we assume that the underlying state, i.e., the preferences of the consumers, are not only used to recommend a particular product, but also are intentionally altered by our algorithm. As in some of the earlier work (Sun et al. 2015; Wang et al. 2015a; Toscher et al. 2008), we use a causal framework to do our modeling. We then take it a step further to mathematically prove that the impact of a treatment can be predesigned and the user can, in theory, be swayed in accordance with the designer's intent. To the best of our knowledge, this is unique to our work. We can further articulate the difference between our work and some of the earlier work using an example in the context of news recommendation. The classical approach tries to show the user news articles he/she might be interested in reading, based on their profile and possibly some other contextual data. A separate process collects information on whether the user clicked on a particular news item and what that item's context is. This collected data is then used to augment the user's profile so that the recommendation part of the process makes a better decision the next time or for the next user. The connection between separate decisions is mainly the enhanced user profile. In reality, the recommended news articles have impacted the user's news preferences to some degree. This is a classical counterfactual problem (Bottou et al. 2013). While the user preferences themselves are latent and cannot be directly measured, the impact manifests itself in a number of ways that are observable. For instance, the user might tweet about that news with a particular sentiment or buy a book online which is related to the topic in the news item. What we prove with our framework is that using the observable data and our model, one can produce a sequence of actions which will influence and steer the user's preferences in a pattern that is intended by the recommender system. These actions can be in the form of content served to the user such as news articles, social media feeds and search results.

In different applications the preferences can be the state and the advertisements (content, the medium of the advertisement, the frequency etc.) are the actions or output of the machine learning algorithm. In a different context, the opinions of the Social Network users on Facebook of a particular event or a new product can be represented as a state. Our model is comprehensive such that the relevant information on the user such as his/her age, gender, de-

mographics and residency is collectively represented by a side information vector since the advertiser collects data on the consumer such as the spending patterns, demographics, age, gender and polls.

A summary of our work in this paper is as follows, with the last bullet being our key contribution:

- We model the effects of machine learning algorithms such as recommendation engines on users through a causal feedback loop. We introduce a complete state space formulation modeling: (1) evolution of preferences vectors, (2) observations generated by users, and (3) causal feedback effects of the actions of algorithms on the system. All these parameters are jointly optimized through an Extended Kalman Filtering framework.
- We introduce algorithms to estimate the unknown system parameters with and without feedback. In both cases, all the parameters are estimated jointly. We emphasize that we provide a complete set of equations covering all the possible scenarios.
- To tune the preferences of users towards a desired sequence, we also introduce a linear regression algorithm and introduce an optimization framework using stochastic gradient descent algorithm. Unlike all the previous work that only use the observations to predict certain desired quantities, as the first time in the literature, we specifically design outputs to "update" the internal state of the system in a desired manner.

The rest of the paper is organized as follows. In the next section, we present a comprehensive state space model that includes the evolution of the latent state vector, underlying observation model and side information. In the same section, we also introduce the causal feedback loop and possible variations to model different real life applications. We then introduce the Extended Kalman Filtering framework to estimate the unknown system parameters. We investigate different real life scenarios including the system with and without the feedback. We present all update and estimation equations. In the following section, we introduce an online learning algorithm to tune the underlying state vector, i.e., preferences vector, towards a desired vector sequence through a linear regression and causal feedback loop. We then demonstrate the validity of our introduced algorithms under different scenarios via simulations. We include our simulation results to show that we are able to converge on unknown parameters in designing a system which can steer user preferences. The final section includes conclusions and scope of future work.

A Mathematical Model for User Preferences with Causal Feedback Effects

In this paper, all vectors are column vectors and denoted by lower case letters. Matrices are represented by uppercase letters. For a vector $\boldsymbol{u}$,

$$\|\boldsymbol{u}\| = \sqrt{\boldsymbol{u}^T \boldsymbol{u}}$$

is the l^2-norm, where $\boldsymbol{u}^{\mathrm{T}}$ is the ordinary transpose. For vectors $\boldsymbol{a} \in \mathbb{R}^m$ and $\boldsymbol{b} \in \mathbb{R}^n$, $\boldsymbol{a}^T$ is the transpose, $[\boldsymbol{a};\boldsymbol{b}] \in \mathbb{R}^{m+n}$ is the concatenated vector. Here, $\boldsymbol{I}$ represents an identity matrix, $\boldsymbol{0}$ represents a vector or a matrix of all zeros and $\boldsymbol{1}$ represents a vector or a matrix of all ones, where the size is determined from the context. The time index is given in the subscript, i.e., $\boldsymbol{x}_t$ is the sample at time $\boldsymbol{t}$. δ_t is the Kronecker delta functions.

We represent preferences of a user as a state vector $\boldsymbol{p}_t$, where this state vector is latent, i.e., its entries are unknown by the system designer. The state vector can represent affinity or opinions of the underlying Social Network user for different products or for controversial issues like privacy. The actual length and values of the preferences depend on the application and context. As an example for the mood of a person in a context of 6 feelings (happy, excited, angry, scared, tender, sad), the preference vector might be $[0, 1, 0, 0, 0, 0]^{\mathrm{T}}$.

The relevant information on the user such as his/her age, gender, demographics and residency is collectively represented by a side information vector $\boldsymbol{s}_t$. The side information on users on the Social Networks can be collected based on their profiles or their friendship networks. We assume that the side-information is known to the designer and, naturally, change slowly so that $\boldsymbol{s}_t = \boldsymbol{s}$ is constant in time.

The machine learning system collects data on the user, say $\boldsymbol{x}_t$, such as Facebook shares, comments, status updates and spending patterns, which is a function of his/her preferences $\boldsymbol{p}_t$ and the side information $\boldsymbol{s}$, given by

$$\boldsymbol{x}_t = F_t\left(\boldsymbol{p}_t, \boldsymbol{s}\right), \tag{1}$$

where the functional relationship $F(.)$ will be clear in the following. Since the information collection process may be prone to errors or misinformation, e.g., untruthful answers in surveys, we extend (1) to include these effects as

$$\boldsymbol{x}_t = F_t(\boldsymbol{p}_t, \boldsymbol{s}) + \boldsymbol{n}_t, \tag{2}$$

where $\boldsymbol{n}_t$ is a noise process independent of $\boldsymbol{p}_t$ and $\boldsymbol{s}$. We can use other approaches instead of an additive noise model, however, the additive noise model is found to accurately model unwanted observation noise effects (Bishop 2007). We use a time varying linear state space model to facilitate the analysis such that we have

$$\boldsymbol{x}_t = \boldsymbol{F}_t \boldsymbol{p}_t + \boldsymbol{n}_t, \tag{3}$$

where $\boldsymbol{F}_t$ is the observation matrix (Anderson and Moore 1979) corresponding to the particular user, $\boldsymbol{n}_t$ is i.i.d. with

$$E\left[\boldsymbol{n}_t \boldsymbol{n}_r^T\right] = \delta_{t-r} \boldsymbol{R},$$

where $\boldsymbol{R}$ is the auto-correlation matrix. The auto-correlation matrix $\boldsymbol{R}$ is assumed to be known, since it can be readily estimated from the data (Anderson and Moore 1979) in a straightforward manner. We do not explicitly show the effect of $\boldsymbol{s}$ on $\boldsymbol{F}$ for notational simplicity.

Based on prior preferences, different user effects and trends, the preferences of the user change. We represent this change as

$$\boldsymbol{p}_{t+1} = G_t(\boldsymbol{p}_t, \boldsymbol{s}) + \boldsymbol{n}_t, \quad (4)$$

with an appropriate $G_t(.)$ function. To facilitate the analysis, we also use a state space model

$$\boldsymbol{p}_{t+1} = \boldsymbol{G}_t \boldsymbol{p}_t + \boldsymbol{v}_t, \quad (5)$$

where $\boldsymbol{G}_t$ is the state update matrix, which is usually close to an identity matrix since the preferences of user cannot rapidly change (Sun et al. 2015; Brodersen et al. 2015). Here, $\boldsymbol{v}_t$ models the random fluctuations or independent changes in the preferences of users, where it is i.i.d. with

$$E\left[\boldsymbol{v}_t \boldsymbol{v}_r^T\right] = \delta_{t-r}\boldsymbol{Q}$$

and $\boldsymbol{Q}$ is the auto-correlation matrix. The auto-correlation matrix $\boldsymbol{Q}$ is assumed to be known, since it can be readily estimated from the data (Anderson and Moore 1979) in a straightforward manner. The model without the feedback effects is shown in Figure 2.

Remark 1: To include local trends and seasonality effects, one can use $\boldsymbol{v}_t = \boldsymbol{B}_t \boldsymbol{u}_t$, where $\boldsymbol{B}_t$ may not be full rank local trends exist (local trends can cause some data points to be derived from others). Also, $\boldsymbol{u}_t$ is an i.i.d. noise process. Our derivations in the next sections can be generalized to this case by considering an extended parameter set.

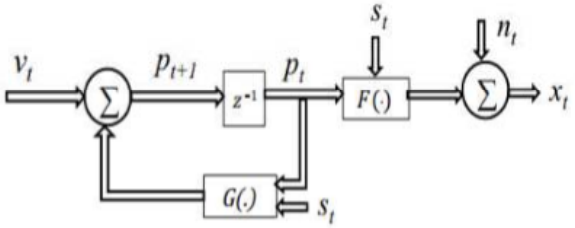


Figure 2: A state-space model to represent evaluation of the user preferences without feedback effects.

In the following, we model the effect of the actions of the machine learning algorithm in the "observation" (3) and "evolution" (5) equations.

Causal Inference Through the Actions of the Machine Learning System

Based on the collected data $\boldsymbol{x}_t$, the algorithm takes an action represented by a_t. The action of the machine learning system or the platform can be either discrete or continuous valued depending on the application (Bishop 2007). As an example, if the action represents a campaign advertisement to be sent to a particular Facebook user, then the set of campaign ads is finite. On the other hand, the action of the machine learning system can be continuous such as providing money incentives to particular users to perform certain tasks such as filling questionnaires. We model the action as a function of the observations as

$$a_t = W_t^T(\boldsymbol{x}_t),$$

where $W(.)$ may correspond to different regression methods (Bishop 2007). To facilitate the analysis, we model the action generation using a linear regression model as

$$a_t = W_t^T \boldsymbol{x}_t, \quad (6)$$

If we have a finite set of actions, i.e. $a_t \in \{1,\ldots, K\}$, we replace (6) by

$$a_t = Q(\boldsymbol{w}_t^T \boldsymbol{x}_t), \quad (7)$$

which is similar to saturation or sigmoid models (Kozat et al. 2007), where $Q(.)$ is an appropriate quantizer. The linear model in (7) can be replaced by more complex models since $\boldsymbol{x}_t$ can contain discrete entries such as gender and age. However, we can closely approximate any such complex relations by piecewise linear models (Vanli and Kozat 2014). The piecewise linear extension of (7) is straightforward (Vanli and Kozat 2014).

Based on the actions of the machine learning algorithm (and prior preferences), we assume that the preferences of the user changes in a linear state space form with an additive model for the causal effect (Wang et al. 2015b; Brodersen et al. 2015; Sun et al. 2015), which yields the following state model:

$$\boldsymbol{p}_{t+1} = \boldsymbol{G}_t \boldsymbol{p}_t + \boldsymbol{v}_t + \boldsymbol{c}_t a_t, \quad (8)$$

where $\boldsymbol{c}_t$ is the unknown causal effect. The complete linear state space model is illustrated in Figure 3. Although, there exists other models for the feedback, apart from the linear feedback, the linear feedback was found to accurately model a wide range of real life scenarios provided that causal effects are moderate (Brodersen et al. 2015), which is typically the case for social networks, i.e., advertisements usually do not have drastic effects on user preferences (Sun et al. 2015; Brodersen et al. 2015). Our linear feedback model can be extended to piecewise linear models to approximate smoothly varying nonlinear models in a straightforward manner.

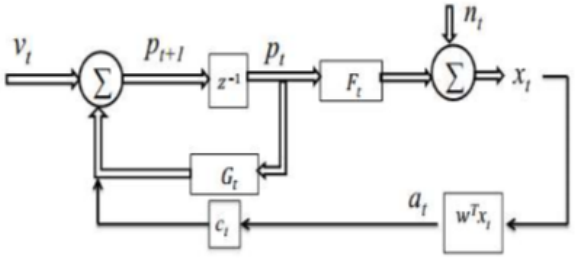


Figure 3: A complete state-space model of the system with action generation and feedback effects.

Remark 2: We can also use a jump state model to represent the causal effects for the case where a_t is coming from a finite set. In this case, as an example, the causal effects will change the state behavior of the overall system through a jump state model as

$$\boldsymbol{p}_{t+1} = \boldsymbol{G}_{t,Q(\boldsymbol{w}^T \boldsymbol{x}_t)} \boldsymbol{p}_t + \boldsymbol{v}_t$$

Our estimation derivations in the following sections can also be extended to cover this case using a jump state-model (Anderson and Moore 1979).

Remark 3: For certain causal inference problems, the actions sequence a_t may be required to be predictive of some reference sequence d_t. In a traffic prediction context, to sway driver preferences $\boldsymbol{p}_t$ in a certain direction by disclosing estimates a_t for a certain road d_t, using some publicly available data $\boldsymbol{x}_t$. To account for these types of scenarios, we complement the model in (8) by introducing

$$d_t = H_t(\boldsymbol{p}_t) + \boldsymbol{\sigma}_t, \tag{9}$$

where $\boldsymbol{\sigma}_t$ is i.i.d. In this case, the feedback loop will be designed in order to tune d_t to a particular value.

In the following, we introduce algorithms that optimize $\boldsymbol{w}_t$ so that the overall system behaves in a desired manner given the corresponding mathematical system. However, we emphasize the overall system parameters including the feedback loop parameters are not known and should be estimated only from the available observations $\boldsymbol{x}_t$. Hence, we carry out both the estimation and design procedures together for a complete system design.

Design of the Overall System with Causal Inference

We consider the problem of designing a sequence of actions $\{a_t\}_{t\geq 1}$ in order to influence users based on our observations $\{\boldsymbol{x}_t\}_{t\geq 1}$, where behavior of the user is governed by his/her hidden preference sequence $\{\boldsymbol{p}_t\}_{t\geq 1}$. The machine learning system is required to choose the sequence $\{\boldsymbol{w}_t\}_{t\geq 1}$ in order to accomplish its specific goal. The specific goal naturally depends on the application. As an example, in Social Networks, the goal can be to change the opinions of users about a new product by sending the most appropriate content such as news articles and/or targeted tweets. In its more general form, we can represent this goal as a utility function and optimize the cumulative gain

$$\max_{\boldsymbol{w}_{t\geq 1}} \sum_{t=1}^{\infty} E[U_t], \tag{10}$$

where $U_t = U_t(\boldsymbol{p}_t)$ is an appropriate utility function for a specific application. To facilitate the analysis, we choose the utility function as the negative of the squared Euclidean distance between the actual consumer preference $\boldsymbol{p}_t$ and some desired state $\boldsymbol{q}_t$. We emphasize that, as shown later in the paper, our optimization framework can be used to optimize any utility function provided that it has continuous first order derivatives due to the stochastic gradient update. In this case (10) can be written as

$$\min_{\boldsymbol{w}_{t\geq 1}} \sum_{t=1}^{\infty} E[\|\boldsymbol{p}_t - \boldsymbol{q}_t\|^2]. \tag{11}$$

The overall system parameters, $\{\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{c}\}$, are not known and should be estimated from our observations. We introduce an Extended Kalman Filtering (EKF) approach to estimate the unknown parameters of the system. We separately consider the estimation framework without the feedback loop, i.e., $\boldsymbol{w} = \boldsymbol{0}$, and with the feedback loop, i.e., $\boldsymbol{w} \neq \boldsymbol{0}$. Clearly the estimation task for $\{\boldsymbol{F}, \boldsymbol{G}\}$ can be carried out before we produce our suggestions $\boldsymbol{w}$. In this case, we can estimate these parameters with a better accuracy without the feedback effects since we need to estimate a smaller number of parameters under less complicated noise processes. However, for certain scenarios where this feedback loop is already active, we also introduce a joint estimation framework for all parameters. A system with feedback is more general, realistic and comprehensive. And feedback is needed in order to tune or influence the preferences of a user in a desired manner. However, a system with feedback is more complex to design and analyze. Therefore, we first provide the analysis for a system without feedback and build on it for an analysis of a system with feedback. After we get the estimated system parameters, we introduce online learning algorithms in order to tune the corresponding system to a particular target internal state sequence, which can be time varying, nonstationary or even chaotic (Singer et al. 2002; Kozat et al. 2007).

Estimating the Unknown Parameters of the System Without Feedback

Without the feedback loop, the system is described by

$$\boldsymbol{p}_{t+1} = \boldsymbol{G}_t \boldsymbol{p}_t + \boldsymbol{v}_t, \tag{12}$$

$$\boldsymbol{x}_t = \boldsymbol{F}_t \boldsymbol{p}_t + \boldsymbol{n}_t, \tag{13}$$

where $\boldsymbol{v}_t$ and $\boldsymbol{n}_t$ are assumed to be Gaussian with correlation matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, respectively. We then define

$$\boldsymbol{\theta}_t \triangleq [\boldsymbol{G}_t(:); \boldsymbol{F}_t(:)],$$

Where $\boldsymbol{G}_t(:)$ is the vectorized $\boldsymbol{G}_t$, i.e., the columns of $\boldsymbol{G}_t$ are stacked one after another to get a full column vector. To jointly estimate $\boldsymbol{p}_t$ and $\boldsymbol{\theta}_t$, we formulate an EKF framework by considering

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \boldsymbol{\varepsilon}_t, \tag{14}$$

where $\boldsymbol{\varepsilon}_t$ is the noise in estimating $\boldsymbol{\theta}_t$ through the EKF. Then, using (12) and (14), and considering $\boldsymbol{p}_t$ and $\boldsymbol{\theta}_t$ as the joint state vector, we get

$$\begin{aligned} \boldsymbol{x}_t &= f_1(\boldsymbol{\theta}_t, \boldsymbol{p}_t) + \boldsymbol{n}_t \\ \begin{pmatrix} \boldsymbol{p}_{t+1} \\ \boldsymbol{\theta}_{t+1} \end{pmatrix} &= \begin{pmatrix} f_2(\boldsymbol{\theta}_t, \boldsymbol{p}_t) \\ \boldsymbol{\theta}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{v}_t \\ \boldsymbol{\varepsilon}_t \end{pmatrix}, \end{aligned} \tag{15}$$

where

$$f_1(\boldsymbol{\theta}_t, \boldsymbol{p}_t) \triangleq \boldsymbol{F}_t \boldsymbol{p}_t$$
$$f_2(\boldsymbol{\theta}_t, \boldsymbol{p}_t) \triangleq \boldsymbol{G}_t \boldsymbol{p}_t$$

are the corresponding nonlinear equations so that we require the EKF framework. The corresponding EKF equations to estimate the augmented states are recursively given as:

$$\begin{pmatrix} \boldsymbol{p}_{t|t} \\ \boldsymbol{\theta}_{t|t} \end{pmatrix} = \begin{pmatrix} \boldsymbol{p}_{t|t-1} \\ \boldsymbol{\theta}_{t|t-1} \end{pmatrix} + \boldsymbol{L}_t \left(\boldsymbol{x}_t - f_1(\boldsymbol{\theta}_{t|t-1}, \boldsymbol{p}_{t|t-1}) \right), \quad (16)$$

$$\boldsymbol{p}_{t+1|t} = f_2(\boldsymbol{\theta}_{t|t}, \boldsymbol{p}_{t|t}), \quad (17)$$

$$\boldsymbol{\theta}_{t+1|t} = \boldsymbol{\theta}_{t|t}, \quad (18)$$

$$\boldsymbol{L}_t = \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t (\boldsymbol{H}_t^T \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t + \boldsymbol{R})^{-1}, \quad (19)$$

$$\boldsymbol{P}_{t|t} = \boldsymbol{P}_{t|t-1} - \boldsymbol{L}_t \boldsymbol{H}_t^T \boldsymbol{P}_{t|t-1},$$

$$\boldsymbol{P}_{t+1|t} = \boldsymbol{D}_t \boldsymbol{P}_{t|t} \boldsymbol{D}_t^T + \boldsymbol{Q}, \quad (20)$$

where

$$\boldsymbol{p}_{t|t} \triangleq \tilde{E}[\boldsymbol{p}_t \mid \boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \ldots]$$
$$\boldsymbol{p}_{t|t-1} \triangleq \tilde{E}[\boldsymbol{p}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \ldots]$$
$$\boldsymbol{\theta}_{t|t} \triangleq \tilde{E}[\boldsymbol{\theta}_t \mid \boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \ldots]$$
$$\boldsymbol{\theta}_{t|t-1} \triangleq \tilde{E}[\boldsymbol{\theta}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \ldots]$$

are EKF terms that approximate the optimal "linear" MSE estimated values in the linearized case, $\boldsymbol{H}_t$ and $\boldsymbol{D}_t$ are the gradients for the first order Taylor expansion needed to linearize the nonlinear state equations in (15)

$$\boldsymbol{H}_t = \begin{pmatrix} (\nabla_{\boldsymbol{p}_t} f_1(\boldsymbol{\theta}_{t|t-1}, \boldsymbol{p}_{t|t-1}))^T \\ (\nabla_{\boldsymbol{\theta}_t} f_1(\boldsymbol{\theta}_{t|t-1}, \boldsymbol{p}_{t|t-1}))^T \end{pmatrix}$$

and

$$\boldsymbol{D}_t = \begin{pmatrix} \nabla_{\boldsymbol{p}_t} f_2(\boldsymbol{\theta}_{t|t}, \boldsymbol{p}_{t|t}) & \nabla_{\boldsymbol{\theta}_t} f_2(\boldsymbol{\theta}_{t|t}, \boldsymbol{p}_{t|t}) \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix}, \quad (21)$$

respectively. Here, $\boldsymbol{L}_t$ is the gain of the EKF, $\boldsymbol{P}_t$ is the error variance of the augmented state. The complete set of equations in (16) to (20) defines the EKF update on the parameter vectors. We next consider the case when there is feedback.

Estimating the Unknown Parameters of the System With Feedback

For estimating the parameters of the feedback loop, i.e., $\boldsymbol{c}_t$ (please see Figure 3), we have two different scenarios. In the first case, where we can control $\boldsymbol{w}$, we set $\boldsymbol{w} = \boldsymbol{\theta}$, estimate $\{\boldsymbol{F}, \boldsymbol{G}\}$ and then subsequently estimate $\boldsymbol{c}$ for fixed $\boldsymbol{w}$. For scenarios where the feedback loop is already present (or we cannot control it), i.e., $\boldsymbol{w} \neq \boldsymbol{\theta}$, we need to estimate all the system parameters under the feedback loop. Naturally, in this case the estimation process is more prone to errors due to compounding effects of the feedback loop on the noise processes. We consider both cases separately.

Using (6) in (8), we get

$$\begin{aligned} \boldsymbol{p}_{t+1} &= \boldsymbol{G}_t \boldsymbol{p}_t + \boldsymbol{v}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{x}_t \\ &= \boldsymbol{G}_t \boldsymbol{p}_t + \boldsymbol{v}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{F}_t \boldsymbol{p}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{n}_t \end{aligned}$$

Hence, the complete state space description with causal loop is given by

$$\boldsymbol{p}_{t+1} = (\boldsymbol{G}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{F}_t) \boldsymbol{p}_t + \boldsymbol{v}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{n}_t, \quad (22)$$

$$\boldsymbol{x}_t = \boldsymbol{F}_t \boldsymbol{p}_t + \boldsymbol{n}_t, \quad (23)$$

In (23), $\boldsymbol{w}_t$ is known, however, all the parameters including $\boldsymbol{c}$ are unknown. We have two cases:

Case 1

Since we can control $\boldsymbol{w}$, we set $\boldsymbol{w} = \boldsymbol{\theta}$ and estimate $\boldsymbol{\theta}$ as $\tilde{\boldsymbol{F}}_t$ and $\tilde{\boldsymbol{G}}_t$ as in the case without feedback. Then, use these estimated parameters in (23) yielding

$$\boldsymbol{p}_{t+1} = (\tilde{\boldsymbol{G}}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \tilde{\boldsymbol{F}}_t) \boldsymbol{p}_t + \boldsymbol{v}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{n}_t,$$
$$\boldsymbol{x}_t = \tilde{\boldsymbol{F}}_t \boldsymbol{p}_t + \boldsymbol{n}_t. \quad (24)$$

To estimate the $\boldsymbol{c}_t$, we introduce an EKF framework by considering $\boldsymbol{c}_t$ as another state vector

$$\boldsymbol{c}_{t+1} = \boldsymbol{c}_t + \boldsymbol{\rho}_t,$$

where $\boldsymbol{\rho}_t$ is the noise in the estimation process, yielding

$$\boldsymbol{x}_t = \tilde{\boldsymbol{F}}_t \boldsymbol{p}_t + \boldsymbol{n}_t,$$
$$\begin{pmatrix} \boldsymbol{p}_{t+1} \\ \boldsymbol{c}_{t+1} \end{pmatrix} = \begin{pmatrix} f_3(\boldsymbol{c}_t, \boldsymbol{p}_t) \\ \boldsymbol{c}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{v}_t \\ \boldsymbol{\rho}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_t \boldsymbol{w}_t^T \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{n}_t \quad (25)$$

where

$$f_3(\boldsymbol{c}_t, \boldsymbol{p}_t) \triangleq \left(\tilde{\boldsymbol{G}}_t + \boldsymbol{c}_t \boldsymbol{w}^T \tilde{\boldsymbol{F}}_t \right) \boldsymbol{p}_t$$

is the corresponding nonlinearity in the system.

In the state update equation (25), unlike the previous EKF formulation, the process noise depends on $\boldsymbol{c}_t$ as $\boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{n}_t$, which is unknown and part of the estimated state vector. Hence, the EKF formulation is more involved.

After several steps, we derive the EKF equations to estimate the augmented states for this case as

$$\begin{pmatrix} \boldsymbol{p}_{t|t} \\ \boldsymbol{c}_{t|t} \end{pmatrix} = \begin{pmatrix} \boldsymbol{p}_{t|t-1} \\ \boldsymbol{c}_{t|t-1} \end{pmatrix} + \boldsymbol{L}_t \left(\boldsymbol{x}_t - \tilde{\boldsymbol{F}}_t \boldsymbol{p}_{t|t-1} \right), \quad (26)$$

$$\boldsymbol{p}_{t+1|t} = f_3(\boldsymbol{c}_{t|t}, \boldsymbol{p}_{t|t}) + \boldsymbol{S}_t \boldsymbol{\Omega}_t^{-1} \left(\boldsymbol{x}_t - \tilde{\boldsymbol{F}}_t \boldsymbol{p}_{t|t-1} \right), \quad (27)$$

$$\boldsymbol{c}_{t+1|t} = \boldsymbol{c}_{t|t}, \quad (28)$$

$$\boldsymbol{L}_t = \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t \boldsymbol{\Omega}_t^{-1}, \quad (29)$$

$$\boldsymbol{S}_t = \boldsymbol{c}_{t|t-1} \boldsymbol{w}_t^T \boldsymbol{R}, \quad (30)$$

$$\boldsymbol{P}_{t|t} = \boldsymbol{H}_t^T \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t + \boldsymbol{R}, \quad (31)$$

$$\boldsymbol{P}_{t|t} = \boldsymbol{P}_{t|t-1} - \boldsymbol{L}_t \boldsymbol{H}_t^T \boldsymbol{P}_{t|t-1},$$

$$\boldsymbol{P}_{t+1|t} = \boldsymbol{D}_t \boldsymbol{P}_{t|t-1} \boldsymbol{D}_t^T - \boldsymbol{B}_t \boldsymbol{\Omega}_t^{-1} \boldsymbol{B}_t^T + \hat{\boldsymbol{Q}}_t, \quad (32)$$

$$\boldsymbol{B}_t = \boldsymbol{D}_t \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t + \begin{pmatrix} \boldsymbol{S}_t \\ \boldsymbol{0} \end{pmatrix}, \quad (33)$$

where

$$\boldsymbol{p}_{t|t} \triangleq \tilde{E}[\boldsymbol{p}_t \mid \boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \ldots]$$

$$\boldsymbol{p}_{t|t-1} \triangleq \tilde{E}[\boldsymbol{p}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \ldots]$$

$$\boldsymbol{c}_{t|t} \triangleq \tilde{E}[\boldsymbol{c}_t \mid \boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \ldots]$$

$$\boldsymbol{c}_{t|t-1} \triangleq \tilde{E}[\boldsymbol{c}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \ldots]$$

are EKF terms that approximate the optimal "linear" MSE estimated values in the lineraized case, $\boldsymbol{H}_t$ and $\boldsymbol{D}_t$ are the gradients for the first order Taylor expansion needed to linearize the nonlinear state equations in (25)

$$\boldsymbol{H}_t = \begin{pmatrix} (\nabla_{\boldsymbol{p}_t} (\tilde{\boldsymbol{F}}_t \boldsymbol{p}_{t|}))^T \\ (\nabla_{\boldsymbol{c}_t} (\tilde{\boldsymbol{F}}_t \boldsymbol{p}_{t|}))^T \end{pmatrix} = \begin{pmatrix} \tilde{\boldsymbol{F}}^T \\ \boldsymbol{0} \end{pmatrix}$$

and

$$\boldsymbol{D}_t = \begin{pmatrix} \nabla_{\boldsymbol{p}_t} f_3(\boldsymbol{c}_{t|t}, \boldsymbol{p}_{t|t}) & \nabla_{\boldsymbol{c}_t} f_3(\boldsymbol{c}_{t|t}, \boldsymbol{p}_{t|t}) \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix},$$

respectively. Here, $\boldsymbol{L}_t$ is the gain of the EKF, $\boldsymbol{P}_t$ is the error variance of the augmented state.

To obtain an expression for $\hat{\boldsymbol{Q}}_t$ in terms of $\boldsymbol{w}_t$, we define the composite error vector $\boldsymbol{b}_t$ for the state update equation so that

$$\hat{\boldsymbol{Q}}_t = E[\boldsymbol{b}_t \boldsymbol{b}_t^T \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \ldots]$$

with

$$\boldsymbol{b}_t \triangleq \begin{pmatrix} \boldsymbol{v}_t \\ \boldsymbol{\rho}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_t \boldsymbol{w}_t^T \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{n}_t.$$

After straightforward algebra, we get

$$\hat{\boldsymbol{Q}}_t = \begin{pmatrix} \boldsymbol{Q} + \boldsymbol{w}_t^T \boldsymbol{R} \boldsymbol{w}_t \boldsymbol{\Gamma}_t & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{U} \end{pmatrix},$$

where

$$\boldsymbol{U} = E[\boldsymbol{\rho}_t \boldsymbol{\rho}_t^T]$$

and

$$\boldsymbol{\Gamma}_t \triangleq \begin{pmatrix} \boldsymbol{0} & \boldsymbol{I} \end{pmatrix} \boldsymbol{p}_{t|t-1} \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{pmatrix} + \boldsymbol{c}_{t|t-1} \boldsymbol{c}_{t|t-1}^T.$$

These updates provide the complete EKF formulation with feedback. In the sequel, we introduce the complete estimation framework where we estimate all the parameters jointly.

Case 2

We can define a superset of parameters

$$\boldsymbol{\theta}_t \triangleq [\boldsymbol{G}_t(:); \boldsymbol{F}_t(:); \boldsymbol{c}_t],$$

and formulate an EKF framework for this augmented parameter vector with

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \boldsymbol{\varepsilon}_t,$$

yields

$$\boldsymbol{x}_t = f_4(\boldsymbol{\theta}_t, \boldsymbol{p}_t) + \boldsymbol{n}_t$$

$$\begin{pmatrix} \boldsymbol{p}_{t+1} \\ \boldsymbol{\theta}_{t+1} \end{pmatrix} = \begin{pmatrix} f_5(\boldsymbol{\theta}_t, \boldsymbol{p}_t) \\ \boldsymbol{\theta}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{v}_t \\ \boldsymbol{\varepsilon}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_t \boldsymbol{w}_t^T \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{n}_t, \quad (34)$$

where

$$f_4(\boldsymbol{\theta}_t, \boldsymbol{p}_t) \triangleq \boldsymbol{F}_t \boldsymbol{p}_t$$

$$f_5(\boldsymbol{\theta}_t, \boldsymbol{p}_t) \triangleq \left(\boldsymbol{G}_t - \boldsymbol{c}_t \boldsymbol{w}^T \boldsymbol{F}_t\right)$$

are the corresponding nonlinear equations so that we require EKF.

After some algebra, we get the complete EKF equations as

$$\begin{pmatrix} \boldsymbol{p}_{t|t} \\ \boldsymbol{\theta}_{t|t} \end{pmatrix} = \begin{pmatrix} \boldsymbol{p}_{t|t-1} \\ \boldsymbol{\theta}_{t|t-1} \end{pmatrix} + \boldsymbol{L}_t \left(\boldsymbol{x}_t - f_4(\boldsymbol{\theta}_{t|t-1}, \boldsymbol{p}_{t|t-1})\right)$$

$$\boldsymbol{p}_{t+1|t} = f_5(\boldsymbol{\theta}_{t|t}, \boldsymbol{p}_{t|t}) + \boldsymbol{S}_t \boldsymbol{\Omega}_t^{-1} \left(\boldsymbol{x}_t - f_4(\boldsymbol{\theta}_{t|t-1}, \boldsymbol{p}_{t|t-1})\right)$$

$$\boldsymbol{\theta}_{t+1|t} = \boldsymbol{\theta}_{t|t}$$

$$\boldsymbol{L}_t = \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t \boldsymbol{\Omega}_t^{-1}$$

$$\boldsymbol{S}_t = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \end{pmatrix} \boldsymbol{\theta}_{t|t-1} \boldsymbol{w}_t^T \boldsymbol{R}$$

$$\boldsymbol{\Omega}_t = \boldsymbol{H}_t^T \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t + \boldsymbol{R}$$

$$\boldsymbol{P}_{t|t} = \boldsymbol{P}_{t|t-1} - \boldsymbol{L}_t \boldsymbol{H}_t^T \boldsymbol{P}_{t|t-1}$$

$$\boldsymbol{P}_{t+1|t} = \boldsymbol{D}_t \boldsymbol{P}_{t|t-1} \boldsymbol{D}_t^T - \boldsymbol{B}_t \boldsymbol{\Omega}_t^{-1} \boldsymbol{B}_t^T + \hat{\boldsymbol{Q}}_t$$

$$\boldsymbol{B}_t = \boldsymbol{D}_t \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t + \begin{pmatrix} \boldsymbol{S}_t \\ \boldsymbol{0} \end{pmatrix}$$

,

where

$$\boldsymbol{H}_t = \begin{pmatrix} (\nabla_{\boldsymbol{p}_t} f_4(\boldsymbol{\theta}_{t|t-1}, \boldsymbol{p}_{t|t-1}))^T \\ (\nabla_{\boldsymbol{\theta}_t} f_4(\boldsymbol{\theta}_{t|t-1}, \boldsymbol{p}_{t|t-1}))^T \end{pmatrix},$$

and

$$\boldsymbol{D}_t = \begin{pmatrix} \nabla_{\boldsymbol{p}_t} f_5(\boldsymbol{\theta}_{t|t}, \boldsymbol{p}_{t|t}) & \nabla_{\boldsymbol{\theta}_t} f_5(\boldsymbol{\theta}_{t|t}, \boldsymbol{p}_{t|t}) \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix}.$$

To obtain an expression for $\hat{\boldsymbol{Q}}_t$ in terms of $\boldsymbol{w}_t$, we define the composite error vector $\boldsymbol{b}_t$ for the state update equation so that

$$\hat{\boldsymbol{Q}}_t = E[\boldsymbol{b}_t \boldsymbol{b}_t^T \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \ldots]$$

with

$$\boldsymbol{b}_t \triangleq \begin{pmatrix} \boldsymbol{v}_t \\ \boldsymbol{\varepsilon}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_t \boldsymbol{w}_t^T \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{n}_t .$$

After straightforward algebra, we get

$$\hat{\boldsymbol{Q}}_t = \begin{pmatrix} \boldsymbol{Q} + \boldsymbol{w}_t^T \boldsymbol{R} \boldsymbol{w}_t \boldsymbol{\Gamma}_t & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{U}_t \end{pmatrix},$$

where

$$\boldsymbol{Q} = E[\boldsymbol{v}_t \boldsymbol{v}_t^T]$$
$$\boldsymbol{U} = E[\boldsymbol{\varepsilon}_t \boldsymbol{\varepsilon}_t^T]$$
$$\boldsymbol{R} = E[\boldsymbol{n}_t \boldsymbol{n}_t^T]$$

and

$$\boldsymbol{\Gamma}_t \triangleq \begin{pmatrix} \boldsymbol{0} & \boldsymbol{I} \end{pmatrix} \boldsymbol{p}_{t|t-1} \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{pmatrix} + \begin{pmatrix} \boldsymbol{0} & \boldsymbol{I} \end{pmatrix} \boldsymbol{\theta}_{t|t-1} \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix} \boldsymbol{\theta}_{t|t-1}^T \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{pmatrix}$$

Given that the system parameters are estimated through the EKF formulation, we next introduce learning algorithms on $\boldsymbol{w}_t$ in order to change the behavior of the users in a desired manner.

Designing a Causal Inference System to Tune User Preferences

After the parameters are estimated through methods described in the previous sections, the complete system framework is given by

$$\boldsymbol{x}_t = \boldsymbol{F}_t \boldsymbol{p}_t + \boldsymbol{n}_t ,$$
$$\boldsymbol{p}_{t+1} = (\boldsymbol{G}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{F}_t) \boldsymbol{p}_t + \boldsymbol{v}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{n}_t , \quad (35)$$

with the estimated

$$\{\boldsymbol{F}_t = \tilde{\boldsymbol{F}}_t, \boldsymbol{G}_t = \tilde{\boldsymbol{G}}_t, \tilde{\boldsymbol{c}}_t = \boldsymbol{c}_t\} .$$

Our goal in this section is to design $\boldsymbol{w}_t$ such that the sequence of preferences $\boldsymbol{p}_t$ are tuned towards a desired sequence of preferences $\boldsymbol{q}_t$, e.g., one can desire to sway the preferences of a user to a certain product.

In order to tune the user preferences, we design $\boldsymbol{w}_t$ so that the difference between the preferences $\boldsymbol{p}_t$ and the desired $\boldsymbol{q}_t$ is minimized. We define this difference as the loss between the preferences and desired vectors as

$$\sum_{k=1}^{t} l(\boldsymbol{p}_k, \boldsymbol{q}_k),$$

where *l(.)* is any differentiable loss function. As an example, for the square error loss, this yields

$$\sum_{k=1}^{t} \left\| \boldsymbol{p}_k - \boldsymbol{q}_k \right\|^2 . \quad (36)$$

To minimize the difference between these two sequences, we introduce a stochastic gradient approach where $\boldsymbol{w}_t$ is learned in a sequential manner. In the stochastic gradient approach, we have

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \mu \nabla_{\boldsymbol{w}} l(\boldsymbol{p}_k, \boldsymbol{q}_k), \quad (37)$$

where $\mu > 0$ is a appropriate learning rate coefficient. The learning rate coefficient is usually selected as time varying with two conditions

$$\mu_t \to 0 \text{ as } t \to \infty \text{ and}$$
$$\sum_{k=1}^{t} \mu_k \to \infty \text{ as } t \to \infty ,$$

e.g. $\mu_t = 1/t$.

If these two conditions are met, then the estimated parameters $\boldsymbol{w}_t$ through the gradient approach will converge to the optimal $\boldsymbol{w}$ (provided that such an optimal point exists) (Bishop 2007). To facilitate the analysis, we set

$$l(\boldsymbol{p}_k, \boldsymbol{q}_k) = \left\| \boldsymbol{p}_k - \boldsymbol{q}_k \right\|^2$$

and get

$$\begin{aligned} \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \mu \nabla_{\boldsymbol{w}_t} \left\| \boldsymbol{p}_k - \boldsymbol{q}_k \right\|^2 \\ &= \boldsymbol{w}_t - 2\mu (\nabla_{\boldsymbol{w}_t} \boldsymbol{p}_t)(\boldsymbol{p}_t - \boldsymbol{q}_t) \end{aligned} . \quad (38)$$

In (38), since $\boldsymbol{p}_t$ is unknown, we use $\boldsymbol{p}_{t|t-1}$ from the causal loop case, i.e. with feedback, and get

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - 2\mu (\nabla_{\boldsymbol{w}_t} \boldsymbol{p}_{t|t-1})(\boldsymbol{p}_{t|t-1} - \boldsymbol{q}_t) . \quad (39)$$

To get

$$\nabla_{\boldsymbol{w}_t} \boldsymbol{p}_{t|t-1},$$

we use the EKF recursion as

$$\boldsymbol{p}_{t|t-1} = (\boldsymbol{G}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{F}_t)(\boldsymbol{p}_{t-1|t-2} + \boldsymbol{L}_{t-1}[\boldsymbol{x}_{t-1} - \boldsymbol{H}_{t-1} \boldsymbol{p}_{t-1|t-2}])$$
$$\boldsymbol{p}_{t|t-1} = \boldsymbol{K}_t \boldsymbol{p}_{t|t-1} + \boldsymbol{M}_t , \quad (40)$$

where

$$\boldsymbol{K}_t = (\boldsymbol{G}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{F}_t)(\boldsymbol{I} - \boldsymbol{L}_{t-1} \boldsymbol{H}_{t-1}),$$

and

$$\boldsymbol{M}_t = (\boldsymbol{G}_t + \boldsymbol{c}_t \boldsymbol{w}_t^T \boldsymbol{F}_t) \boldsymbol{L}_t x_{t-1} .$$

Using (40), we get a recursive update on the gradient as

$$\nabla_{\boldsymbol{w}_t} \boldsymbol{p}_{t|t-1} = \nabla_{\boldsymbol{w}_t} \boldsymbol{K}_t \boldsymbol{p}_{t|t-1} + \boldsymbol{K}_t \nabla_{\boldsymbol{w}_t} \boldsymbol{p}_{t|t-1} \nabla_{\boldsymbol{w}_t} \boldsymbol{M}_t , \quad (41)$$

From (39), (40) and (41), we get the complete recursive update as

$$\boxed{\begin{aligned} \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - 2\mu (\nabla_{\boldsymbol{w}_t} \boldsymbol{p}_{t|t-1})(\boldsymbol{p}_{t|t-1} - \boldsymbol{q}_t) \\ \boldsymbol{p}_{t|t-1} &= \boldsymbol{K}_t \boldsymbol{p}_{t|t-1} + \boldsymbol{M}_t \\ \nabla_{\boldsymbol{w}_t} \boldsymbol{p}_{t|t-1} &= \nabla_{\boldsymbol{w}_t} \boldsymbol{K}_t \boldsymbol{p}_{t|t-1} + \boldsymbol{K}_t \nabla_{\boldsymbol{w}_t} \boldsymbol{p}_{t|t-1} \nabla_{\boldsymbol{w}_t} \boldsymbol{M}_t \end{aligned}} . (42)$$

This completes the derivation of the stochastic gradient update for online learning of the tuning regression vector.

Experiments

In this section, we share our simulation results to show that estimated parameters of the system converge to the real values, proving that a system can be designed with the right parameters which allows a sequence of actions or interventions to tune the preferences of a user in a desired manner. Since our goal is mainly to establish a pathway to the possibility of designing a system that can steer user preferences in a desired manner, we suffice our simulation cases to a basic set based on the mathematical proof we provided in the form of EKF formulations. The true parameters of the system are known to us since we are running our experiments in the form of simulations. Specifically, the preferences of the user, which are not directly observable in real life, are known in case of simulations. We run simulations for the EKF formulations we derived in the previous sections to show that our estimation of the preferences converge to the real preference values. We illustrate the convergence of our algorithms under different scenarios.

In the first scenario, we have the case where the corresponding system has no feedback. As the true system, we choose a second order linear state space model, where $\boldsymbol{G} = 0.95\boldsymbol{I}$ and $\boldsymbol{F} = \boldsymbol{I}$ with $\boldsymbol{Q} = 3 \times 10^{-3}\ \boldsymbol{I}$ and $\boldsymbol{R} = 3 \times 10^{-3}\ \boldsymbol{I}$. For the EKF formulation, we choose two different variances for ε_t, e.g., 10^{-3} and 10^{-4} to demonstrate the effect of this design parameter on the system. We emphasize that neither $\boldsymbol{F}$ or $\boldsymbol{G}$ are known, hence as long as the system is observable, particular choices of $\boldsymbol{F}$ and $\boldsymbol{G}$ only change the convergence speed and the final MSE. However, we choose $\boldsymbol{F}$ to make the system stable.

In Figure 4, we plot the square error difference between the estimated preferences and the real preferences

$$trE[\left\| \boldsymbol{p}_t - \boldsymbol{p}_{t|t-1} \right\|^2]$$

with respect to the number of iterations, where we produce the MSE curves after averaging over 100 independent trials. We also plot the cumulative MSE normalized with respect to time, i.e.,

$$\frac{\sum_{k=1}^{t} trE[\left\| \boldsymbol{p}_t - \boldsymbol{p}_{t|t-1} \right\|^2]}{t},$$

to show that as the iteration count increases, the averaged MSE steadily converges. The plot includes both the average MSE and the cumulative MSE normalized in time for estimation of $\boldsymbol{F}$ and $\boldsymbol{G}$. We observe that the estimation of $\boldsymbol{F}$ and $\boldsymbol{G}$ are more prone to errors due to the multiplicative uncertainty, single observation and state update equations. However, both the estimated preferences vectors as well as the system parameters converge.

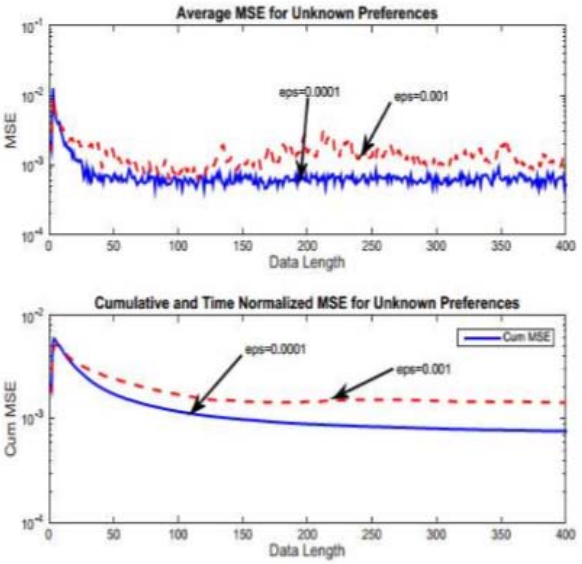


Figure 4: Estimation of the underlying preferences vector when there is no feedback. The results are averaged over 100 independent trials. Here, we have no feedback and parameters of both the state equation as well as the observation equation are unknown. The results are shown for two different noise variances for the EKF formulation.

In the second set of experiments, we have feedback present, i.e., $\boldsymbol{w} \neq \boldsymbol{0}$. For this case, we now have similar parameters as in the first set of experiments, except $\boldsymbol{G} = 0.9\boldsymbol{I}$ to give more decay due to presence of feedback. For this case, we choose two different scenarios, where $\boldsymbol{w}_t$ and $\boldsymbol{c}_t$ are fixed or randomly chosen provided that the overall system stays stable after the feedback, i.e., $(\boldsymbol{G} + \boldsymbol{c}\ \boldsymbol{w}_t\ \boldsymbol{F})$ corresponds to a stable system. Note that this can be always forced by choosing an appropriate $\boldsymbol{w}$. However, we choose randomly initialized $\boldsymbol{w}$ to avoid any bias in our experiments. Here, although $\boldsymbol{w}$ is known to us, the feedback amount $\boldsymbol{c}$ as well as the hidden preferences are unknown. In Figure 5, we plot the MSE between the estimated preference vectors and the true ones. We observe from these simulations that although the feedback produces a multiplicative uncertainty in the state equation and greatly enhance the nonlinearity in the update equation, we are able to recover the true values through the EKF formulation. We observe that although due to feedback we have more colored noise in the state equation, we recover true values due to the whitening effects of the EKF. The MSE error between the estimated feedback and the true one are plotted, where the MSE curves are produced after 100 independent realizations.

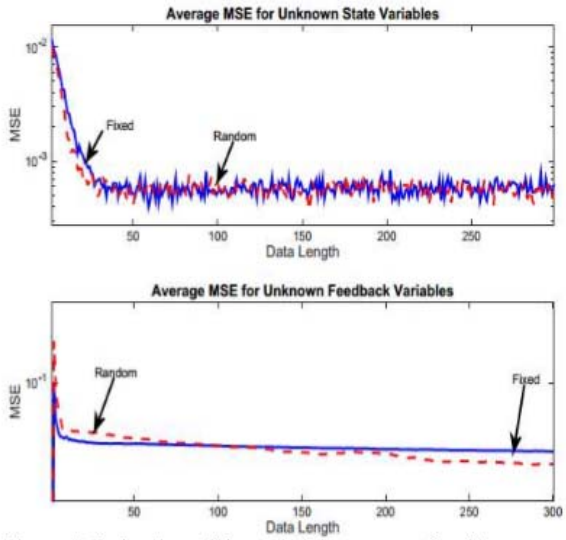


Figure 5: Estimation of the underlying vector of preferences and the feedback parameters when there is feedback. The results are averaged over 100 independent trials. Two different configurations are simulated for the feedback as well as for the linear control parameters, e.g., the fixed and random initial cases. For both scenarios, our estimation process converges to the true underlying processes.

Conclusions

In this paper, we model the effects of the machine learning algorithms such as recommendation engines on users through a causual feedback loop. To this end, we introduce a complete state space formulation modeling: (1) evolution of preference vectors, (2) observations generated by users, and (3) the causal feedback effects of the actions of machine learning algorithms on the system. All these parameters are jointly optimized through an Extended Kalman Filtering framework. We introduce algorithms to estimate the unknown system parameters with and without feedback. In both cases, all the parameters are estimated jointly. We emphasize that we provide a complete set of equations covering all the possible scenarios. To tune the preferences of users towards a desired sequence, we also introduce a linear feedback and introduce an optimization framework using stochastic gradient descent algorithm. Unlike previous work that only use the observations to predict certain desired quantities, we specifically design outputs to "update" the internal state of the system in a desired manner. Through a set of experiments, we demonstrate the convergence behavior of our proposed algorithms in different scenarios.

We consider our work as a significant theoretical first step in designing a system with the right parameters which allows a sequence of actions or interventions to tune the preferences of a user in a desired manner. We emphasize that the main goal of our study is to establish a pathway to designing such a system. We achieve this by first providing mathematical proof and then through a basic set of simulations.

A next step in future studies can be to make the system more stable and also to make the design process easy and practical for system designers. Further analysis on the convergence of the system along with more simulations, experiments and numerical analysis are needed to take our results to the next level. A direct comparison to previous studies is not possible for this first step of our study since, to the best of our knowledge, this is the first time a task of this nature is being undertaken. Our main success criteria is the fact that estimated parameters converge to the real parameter values. However, as our framework evolves, we will be able to track its relative performance.

Another area of focus for future studies is the optimal selection of action sequences. This can be particularly challenging since user preferences can change over time due to the abundance of new products and services. Algorithms to optimally select actions may require online learning and decision making in real time to accommodate these changes.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Acknowledgements

We would like to thank Koc University Graduate School of Social Sciences and Humanities for their support. This work was also supported by the BAGEP Award of the Science Academy.

References

Achbany, Y.; Jureta, I. J.; Faulkner, S.; and Fouss, F. 2008. Continually learning optimal allocations of services to tasks. *IEEE Transactions on Services Computing* 1 (3) (2008) 141–154.

Anderson, B. D. O., and Moore, J. B. 1979. *Optimal Filtering*, Prentice-Hall, N.J., 1979.

Bishop, C. 2007. Pattern Recognition and Machine Learning, Springer, NY, 2007.

Bottou, L., and Cun, Y. L. 2005. Online learning for very large data sets. *Applied Stochastic Models in Business and Industry* 21 (2005) 137–151.

Bottou, L., and Bousquet, O. 2007. The tradeoffs of large scale learning. In *Advances in Neural Information Processing (NISP)*, 2007, pp. 1–8.

Bottou, L.; Peters, J.; Charles, D. X.; Chickering, D. M.; Portugaly, E.; Ray, D.; and Simard, P. 2013. Counterfactual

reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research* 14 (2013) 3207–3260.

Brodersen, K. H.; Gallusser, F.; Koehler, J.; Remy, N; and Scott, S. L. 2015. Inferring causal impact using bayesian structural time-series models. *The Annals of Applied Statistics* 9 (1) (2015) 247-274.

Chan, D.; Ge, R.; Gershony, O.; Hesterberg, T.; and Lambert, D. 2010. Evaluating online ad campaigns in a pipeline: Causal models at scale. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 418–459.

Epstein, R., and Robertson, R. E. 2015. The search engine manipulation effect (SEME) and its possible impact on the outcomes of elections. In *PNAS*.

Gupta, V.; Kedia D.; Varshney, D.; Jhamtani, H.; and Karwa, S. 2014. Identifying Purchase Intent from Social Posts. In *ICWSM 2014*.

Jahrer, M.; Toscher, A.; and Legenstein, R. 2010. Combining predictions for accurate recommender systems. In *Proceedings of SIGKDD*, 2010, pp. 693-702.

Kozat, S. S.; Singer, A. C.; and Zeitler G. C. 2007. Universal piecewise linear prediction via context trees. *IEEE Transactions on Signal Processing* 55 (7) (2007) 3730–3745.

Peng, H.; Liang, D.; and Choi, C. 2013. Evaluating parallel logistic regression models. In *2013 IEEE International Conference on Big Data*, 2013, pp. 119–126. DOI: http://dx.doi.org/10.1109/BigData.2013.6691743.

Ruta, D. 2014. Automated trading with machine learning on big data. In *Proceedings of the 2014 IEEE International Congress on Big Data (BigData Congress)*, 2014, pp. 824–830.

Salah, A. A.; Lepri, B.; Pianesiand, F; and Pentland, A. S. 2011. Human behavior understanding for inducing behavioral change: Application perspectives," In *2nd International Workshop on Human Behavior Understanding*, 2011.

Singer, A. C.; Kozat, S. S.; and Feder M. 2002. Universal linear least squares prediction: upper and lower bounds. *IEEE Transactions on Information Theory* 48 (8) (2002) 2354–2362. DOI: http://dx.doi.org/10.1109/TIT.2002.800489.

Subakana, Y. C.; Kurt, B.; Cemgil, A. T.; and Sankur B. 2014. Probabilistic sequence clustering with spectral learning. *Digital Signal Processing* 29 (2014) 1–19.

Sun, W.; Wang, P.; Yin, D.; Yang, J.; and Chang, Y. 2015. Causal inference via sparse additive models with application to online advertising. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI 2015)*, 2015.

Toscher, A.; Jahrer, M.; and Legenstein, R. 2008. Improved neighborhood-based algorithms for large-scale recommender systems. In *2nd Netflix-KDD Workshop*, 2008, pp. 14–21.

Wang, P.; Yin, D.; Yang, J.; Chang, Y.; and Meytlis, M. 2015. Rethink Targeting: Detect 'smart cheating' in online advertising through causal inference. In *Proceedings of the 24th International World Wide Web Conference (WWW 2015)*, 2015.

Wang, P.; Sun, W.; Yin, D.; Yang, J.; and Chang, Y. 2015. Robust tree-based causal inference for complex ad effectiveness analysis. In *Proceedings of the 8th ACM Conference on Web Search and Data Mining (WSDM 2015)*, 2015.

Wang, Y., and Djuric, P. M. 2015. Social learning with heterogeneous agents and sequential decision making. *Digital Signal Processing* 47 (2015) 17–24.

Vanli, N. D., and Kozat, S. S. 2014. A comprehensive approach to universal piecewise nonlinear regression based on trees. *IEEE Transactions on Signal Processing* 62 (20) (2014) 5471–5486.

Yan, J.; Liu, N.; Jian, Y.; and Chen, Z. 2009. How much can behavioral targeting help online advertising? In *Proceedings of the 18th international conference on World wide web, 2009*, pp. 261-270.

Zarsky, T. Z. 2004. Thinking outside the box: Considering transparency, anonymity, and pseudonymity as overall solutions to the problems of information privacy in the internet society. University of Miami Law Review, vol. 58, pp. 1301-1354, 2004.