Applied Machine Learning in Operations Management

Hamsa Bastani[1], Dennis J. Zhang[2], Heng Zhang[3]
[1]Wharton Business School, University of Pennsylvania
[2]Olin Business School, Washington University in St. Louis, St. Louis, MO, United States
[3]W. P. Carey School of Business, Arizona State University
[1]hamsab@wharton.upenn.edu, [2]denniszhang@wustl.edu, [3]hengzhang24@asu.edu

The field of operations management has witnessed a fast-growing trend of data analytics in recent years. In particular, spurred by the increasing availability of data and methodological advancement in machine learning, a large body of recent literature in this field takes advantage of machine learning techniques for analyzing how firms should operate. In this chapter, we review applications of different machine learning methods, including supervised learning, unsupervised learning, and reinforcement learning, in various areas of operations management. We highlight how both supervised and unsupervised learning shape operations management research in both descriptive and prescriptive analyses. We also emphasize how different variants of reinforcement learning are applied in diverse operational decision problems. We then identify several exciting future directions at the intersection of machine learning and operations management.

Key words: Machine Learning, Supervised Learning, Unsupervised Learning, Reinforcement Learning, Operations Management, Data Analytics, Healthcare, Revenue Management, Supply Chain Management

1. Introduction

In recent years, the abundance of data being collected and stored, together with more affordable and faster computing power, has driven the rapid development of algorithms to find trends or patterns in data. This has given rise to the field of machine learning (ML). Born as a sub-field of computer science, it uses light modeling assumptions and relies on data, statistics and computational theory to build scalable algorithms capable of dealing with large data sets to discover useful decision rules. As more algorithms are developed to cover a wide range of applications from business to engineering, ML is helping most academic fields to substantially improve their use of data sets, and the field of operations management (OM) is no exception. On the one hand, ML has helped OM researchers to better solve estimation problems in operations design and optimization; on the other hand, it has triggered OM researchers to rethink OM problems — besides using mathematical models to derive insights, one can also combine ML algorithms and data to facilitate accurate predictions, discover diagnoses, and directly find solutions for the problem at hand.

In this chapter, we aim to provide a preliminary introduction to the modern ML literature (for a more in-depth discussion of ML literature, readers can refer to [31, 70, 78]) and to shed light on how the ML

literature has helped to reshape the OM literature in recent years. The ML literature can be classified into *supervised learning*, *unsupervised learning* and *reinforcement learning*. In *supervised learning*, ML models are trained with labeled data to learn a function mapping from inputs, often referred to as features, and outputs (i.e., labels) based on example input-output pairs. In *unsupervised learning*, the ML algorithm tries to look for data patterns in a data set with no pre-existing labels. For example, one may consider day-to-day operations of an online advertising platform that helps manage the advertisers' campaigns. The platform collects a huge number of data records of users visiting websites and clicking on ads. Supervised learning helps the platform to predict whether a given user will click on a given advertisement. In this case, the focal user's historical behaviors, as well as demographics and the characteristics of ads, are features, and the label is the focal user's action towards an ad, such as a click or conversion. In this same setting, one can also utilize unsupervised learning to identify patterns of previous consumer behaviors and to classify consumers into different categories and adjust ad exposure accordingly. This is unsupervised learning since the customer categories are learned from the features rather than the labels.

The third type of ML problem, more related to the large dynamic optimization literature in OM, is *reinforcement learning*. It applies to settings in which an agent interacts with the environment, takes different actions, and learns from the resulting reward. Unlike supervised learning, where rewards are immediately defined (based on whether the label is correct), the rewards in reinforcement learning are more long-term since the state can be dynamically altered by actions. The *multi-arm bandit* (MAB) problem is an important and special case of reinforcement learning, where there is only one state. As an example, in online advertising platforms, reinforcement learning can be used to continuously update advertising strategies during consumer interactions to maximize long-term cumulative revenue and minimize the long-term impact of ads on consumer experience.

On the OM side, broadly speaking, one can divide the application of ML models in OM settings into two types, *descriptive analysis* and *prescriptive analysis*, depending on the desired research goals. In the former, the results from ML models either carry important managerial insights by themselves or can be directly used in operational decision making. In the latter, ML models are embedded as part of an optimization problem to solve typical operations problems. Nowadays, all three types of ML models are applied in descriptive and prescriptive analyses in OM.

The remainder of this chapter will be organized as follows. In Section 2, we provide a parsimonious introduction to the history of the ML literature. We discuss supervised learning in Section 3, unsupervised learning in Section 4, and reinforcement learning in Section 5; we provide brief background on each topic, and overview how these tools are used in the OM literature. In Section 6, we conclude the chapter and discuss future possible research directions at the intersection of ML and OM.

2. A Brief History of ML

Early ML techniques emerged from ideas in the statistical modeling literature. Two prominent examples in this literature are linear regression, whose creation was often attributed to Adrien-Marie Legendre and Carl Gauss in the 19th century, and Bayesian inference, which was founded by Thomas Bayes and Pierre-Simon Laplace in the 18th century [9, 23]. Both examples utilize statistical models to extract patterns from data that are useful for decision making, and such computation can easily be conducted by researchers at the time of creation. The real interest in using machines to solve statistical models and find patterns—or in other words, "learn"—did not occur until the 1950s and 1960s. Arthur Samuel, an American pioneer who worked for IBM, created the first computer learning programs designed to play checkers in 1952 and coined the term "machine learning" in 1959 [118]. In 1957, Frank Rosenblatt created the very first neural network for computers. In 1963, Donald Michie started using reinforcement learning to play Tic-tac-toe [163]. 1967 saw the invention of nearest neighbor algorithms, which is often regarded as the start of modern pattern recognition [117]. During this period, the ML community also showed renewed interest in Bayesian methods. Then in the 1970s, the ML community was rather silent, perhaps due to the first 'AI winter' caused by pessimism about the ability of ML to solve real-world problems.

Researchers' interest in ML began to pick up again in the 1980s. In 1982, John Hopfield popularized Hopfield networks, a type of recurrent neural network that can serve as content-addressable memory systems [89]. The important step of using back propagation in neural networks came in 1986, when David Rumelhart, Geoffrey Hinton and Ronald J. Williams extended an earlier algorithm created in 1962 [130]. This allowed multiple layers to be used in a neural network and dramatically increased the power of neural networks in approximating complex functions. At the same time, the seminal work [36] on decision trees, published in 1984, marked the start of tree-based learning methods. In 1989, Christopher Watkins developed Q-learning, which greatly improved the performance of reinforcement learning [160].

The popularity of ML kept increasing in the research community in the 1990s, and many important discoveries were made during this time. An important shift in ML, triggered by the rapid growth of computing power and the availability of data, was from a knowledge-driven approach to a data-driven approach. The concept of boosting was first presented in a paper [134] by Robert Schapire. In 1995, Tin Kam Ho published a paper discussing random decision forests [86], and the influential work on supporter vector machines (SVM) by Corinna Cortes and Vladimir Vapnik was published [52]. Sepp Hochreiter and Jürgen Schmidhuber invented long short-term memory (LSTM) recurrent neural networks in 1997, greatly improving the efficiency and practicality of recurrent neural networks [87]. In that same year, a remarkable event took place that marked the first super-human behavior of machines in human games—the IBM computer Deep Blue, which utilized decision rules and statistical models to play chess, beat the world chess champion.

Named by Geoffrey Hinton in 2006, deep learning is a type of supervised learning built on large neural networks [85], and it has played an essential role in ML development in the 21st century. ImageNet was created by Fei-Fei Li in 2009 [59]. It is a large visual database, often deemed as the catalyst for the deep learning boom of the 21st century, since many deep learning researchers test their work using this database. The influential paper [77] was published in 2011, showing that neurons based on the rectified linear unit (ReLU) can result in both better training and better performance of deep neural networks. The generative adversarial network (GAN) was proposed in 2014 [79]. These developments greatly helped popularize the application of deep learning in both academia and industry. Many leading information technology firms realized the importance of ML and its enormous potential for their business growth and decided to join the field. Several large projects, such as GoogleBrain by Google (2012), DeepFace by Facebook (2014), and DeepMind by Google (2014), led the development of ML in this new era in the industry. In particular, in 2014, Facebook researchers published their work on DeepFace, which could identify faces with 97.35% accuracy, rivaling human performance [146]. In 2016, the AlphaGo algorithm developed by DeepMind defeated a professional player at the Chinese board game Go, which is considered the most complex board game in human history. Later, AlphaZero, which extended the techniques from AlphaGo and combined them with reinforcement learning to train itself from zero basis, was able to surpass the ability of AlphaGo with only three days' training.

In the OM literature, the practice of applying a ML-related approach may have started with the early applications of time-series methods in demand forecasting for inventory problems, such as the well-known Box-Jenkins method [32] and other subsequent extensions [53, 73]. Regression models, often regarded as of one of fundamental techniques in ML, usually serve as the building blocks in such applications. The last decade has witnessed a growing trend of data analytics in OM research, owing to the increasing availability of data and computing power. This has led to a fast-growing literature, which we discuss in detail in the next three sections.

3. Supervised Learning

In this section, we focus on supervised learning. After providing a brief introduction, we will review applications in both descriptive and prescriptive problems in OM.

3.1. General Introduction to Supervised Learning

Supervised learning algorithms are designed to "learn by example"—i.e., to infer a function from a set of labeled training examples. These training examples serve as "supervisors" to make sure that the algorithm generates a model that can produce a high-quality forecast of these labels. More formally, each training example can be written in the form of $(\mathbf{x}_i, y_i)$, with $\mathbf{x}_i$ as the observed feature vector and y_i as the observed label, which can be either discrete or continuous. We usually assume that $(\mathbf{x}_i, y_i)$ are i.i.d. realizations from certain unknown distribution. For example, $\mathbf{x}_i$ may describe the characteristics of a customer visiting a

website, such as gender, age, location and even the brand of the mobile device being used; and $y_i = 1$ if the consumer clicks an advertisement, and 0 otherwise. A supervised learning algorithm would choose a function f from a class of functions, say $\mathcal{F}$, such that $y_i \approx f(\mathbf{x}_i)$. The quality of learning is usually measured by the difference between the predicted labels and the true label in the data (commonly referred to as a loss function), such as the widely used mean squared error (MSE). In this case, the training loss of any function $f \in \mathcal{F}$ can be described as

$$\sum_i (y_i - f(\mathbf{x}_i))^2.$$

Then, in the training process, we aim to find $\hat{f} \in \mathcal{F}$ that minimizes or approximately minimizes the chosen error metric. Once the function $\hat{f}$ is trained, it is usually straightforward to apply it in a new testing example with feature vector $\mathbf{x}$ and to predict its label as $f(\mathbf{x})$.

It is not always desirable to specify a very complex function class $\mathcal{F}$, so that one can drive the training loss as small as possible, or even achieve zero training loss. A function class of higher complexity usually has low bias, meaning that it entails a function that can better mimic the true underlying relationship between $\mathbf{x}_i$ and y_i. However, with high complexity, the output of the training process, $\hat{f}(\cdot)$, usually is more sensitive to small fluctuations in the training data and, therefore, exhibits larger variance. This usually leads to a model with inadequate generalizability that over-fits the noise of training data and performs poorly on the new testing data. This well-known phenomenon is called *bias-variance trade-off*.

In fact, one can show that the mean squared error on the underlying true distribution, which arguably represents what a modeler is truly interested in, can be exactly decomposed as the sum of the error due to bias and variance. Therefore, a key element to the discovery of high-performance models lies in how to balance bias and variance to obtain models with high generalizability. Different techniques to address this issue and control model complexity are designed for different supervised learning models, as reviewed in detail in [70]. A typical method that is core to many such techniques is the data-driven procedure of *cross-validation*. In a typical cross-validation process, one partitions the training data into complementary subsets — training models with different complexity parameters on one subset and validating these models on the other. To reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are combined (e.g., averaged) over the rounds to give estimates of the predictive performance of these models. Then, an appropriate model complexity parameter is chosen to train the final output model on the entire training data set.

Supervised learning has been the most widely applied method in ML. In the business world, it has become the workhorse of many modern industries. For instance, recall the running example of online advertising we discussed in Section 1: supervised learning algorithms are used to make predictions of consumer clicking behaviors, which enables large platforms to run auctions selling display advertisement opportunities. This has evolved into an industry that generated about $130 billion across the United States in 2019 [128]. Without

supervised learning, it may not be possible to run such a precisely targeted advertisement auction in this industry.

Supervised learning has also been one of the most active research areas in the past few decades. This realm is so large that most algorithms go beyond the scope of our review. We now briefly introduce several important classes of supervised learning models that have recently received attention in OM applications. For a more comprehensive list of well-known supervised learning models and relevant work in the computer science literature, please refer to Table 1.

Class	**Model**	**Papers**
Regression-Based Methods	LASSO Regression	[149]
	Ridge Regression	[88]
	Generalized Linear Models	[125]
	Generalized Additive Models	[83]
Local Methods	k-nearest neighbors (KNN)	[68]
	Local Linear Regression	[48, 49]
	Classification and Regression Trees	[36]
Ensemble Methods	Bagging	[34, 37, 86]
	Random Forest	[35]
	Boosting	[69]
Other Methods	Naive Bayes and Bayesian Networks	[71, 110, 133]
	Support Vector Machine (SVM)	[52]
Deep Neural Networks	Deep FeedForward Neural Network (D-FFNN)	[72, 114]
	Convolutional Neural Network (CNN)	[109, 135]
	Residual Network	[84]
	Long Short Term Memory (LSTM) Network	[87]
	Radial Basis Function (RBF) Network	[39]

Table 1 Some well-known supervised learning models and seminal papers.

One may consider *linear-regression-based models* the simplest class of supervised learning models. In its most basic form, $f(\mathbf{x})$ is modeled as a linear function of $\mathbf{x}$, namely $\beta \cdot \mathbf{x}$, for some β to be estimated from data by minimizing the training error. There are several variations of linear regression models, such as logistic regression models suitable for classification tasks, or regression models with regularization (e.g., LASSO or ridge regressions) that are designed to control model complexity and perform better for high-dimensional problems. Another popular class of models is *non-parametric local learning models*, in which local information of training examples is used to model the relationship between $\mathbf{x}$ and y. A brute force implementation of such models is the k-nearest-neighbor model (KNN), in which, given any testing data, we find a few training examples that are the most similar to it in terms of the feature vectors. This model has significant memory and computational requirements to find neighbors, so more efficient models are more popular in practice. Specifically, tree-based models extend these local models by using a tree-like structure

to effectively store local information. They are easy to implement with good empirical performance and are amenable to visualization and interpretation. Therefore, they are often viewed as an "off-the-shelf ML method." However, a disadvantage of decision tree models is that they often over-fit the data, as trees can be arbitrarily complex. Ensemble methods, such as random forests or gradient boost trees, effectively overcome this challenge.

It is widely recognized that the most powerful method for supervised learning developed so far is probably deep learning. It uses artificial neural networks, a model ensemble architecture inspired by the human brain, to simulate complex functional forms. The term 'deep' comes from the fact that a network with deep layers usually works well for complex ML problems. Recently, deep learning has taken off as the most popular ML method, because of its capability to model complex functional forms, its superior ability to process large numbers of features, and its insuperable prediction accuracy when supported with enough training data. For a comprehensive discussion of deep learning, please refer to [78].

3.2. Supervised Learning for Descriptive Analysis in OM

In the OM literature, the goal of descriptive analysis typically is either *prediction* or *inference*. In prediction settings, researchers make use of ML models to forecast an outcome variable that can be used later in operational decisions; here, the accuracy of prediction is the foremost quality measure. In contrast, in inference problems, we are interested in understanding how an outcome variable is generated as a function of the input data. Typically, this function is specified by a model, and our goal is to learn the underlying model parameters. Causal inference — i.e., evaluating the effect of different treatments or policies on subjects — is central to this literature.

3.2.1. Prediction with Supervised Learning Many authors in OM develop supervised-learning-based prediction models for operational problems. Often, ML models have to be customized for operational settings to achieve higher accuracy. One typical application is demand forecasting in inventory management. For example, to predict the demand for new products (for which demand data are scarce), [13] develops a ML model that pools comparable products together to "borrow" information to forecast with higher precision. The innovation lies in the simultaneous determination of the product clusters and demand forecast, which is implemented through an iterative algorithm that alternates between learning and optimal cluster assignment. The authors demonstrate, both theoretically and empirically through real data, that the model outperforms traditional ML models in terms of prediction accuracy. Another example in supply chains is [54], which uses different ML models to forecast the demand of fashion products utilizing social media information. The authors specifically adapt the model to the context and estimate of the value of social media information in demand forecasting. One can also incorporate data from auxiliary sources to improve prediction, otherwise known as *transfer learning* in the ML literature [126]. Along these lines, [65] uses tensor completion to

improve product recommendations from multiple customer responses, while [14] uses high-dimensional techniques to improve prediction from proxy data sources.

In healthcare, [8] proposes the Q-LASSO method for hospital emergency department wait-time forecasting, by mixing the celebrated LASSO approach with queuing theory. Using real data, the authors show that Q-LASSO outperforms known forecasting methods in predicting waiting times by a large margin. [129] empirically study the relationship between technology enabled continuity of care, patient activation measures (which describes patients' skills, knowledge, and motivation to actively engage in their health care), and patient readmission. In this study, they use the SVM model to predict patient activation measures for individual patients and show that technology enabled continuity of care is a very important predictor.

An interesting discussion regarding machine-learning-based predictions versus experience-based human forecast in healthcare settings is provided in [92]. It shows that a combined predictor that integrates physicians' forecast of surgery duration with that from data-based models performs better than either forecast. The follow-up study [93] discusses this phenomenon in more general settings with a theoretical model. It is proven that, rather than directly merging a human forecast with that from data-driven algorithms, more accurate predictions can be obtained by carefully leveraging private human information about how the algorithm should adjust its forecast to account for the information that only the human has access to.

In revenue management, ML techniques are used to obtain more predictive models. For example, [43] proposes a nonparametric choice model that relaxes the rationality assumption of consumer choice behavior. It assumes that each customer type can be represented by a binary decision tree—which describes a decision process for making a purchase based on checking for the existence of specific products in the assortment—and that there is a distribution over different customer types. [124] discusses cancellation rate forecasting, which plays an important role in revenue management with selling perishable service products with fixed capacity through a fixed booking horizon. They show that different relevant variables in different stages of the booking horizon can be fed into supervised learning algorithms—for example, kernel logistic regression, SVM and random forest—to achieve improved forecasting of cancellation rates.

3.2.2. Causal Inference with Supervised Learning One is often interested in evaluating the *causal* effect of different treatments or policies on subjects. A typical use case of ML methods is to precisely estimate *heterogeneous* treatment effects across different subjects, and target treatments to subjects that will generate the largest reward. For example, in healthcare, an important question is: what are the effects of different medical treatments on a given patient, and how can we develop personalized treatment policies to cater to the needs of different patients? As an example, [27] studies personalization for patients with diabetes. The authors use the KNN method to determine the most similar patients to a focal patient in terms of their medical attributes. Then, these "neighbors" are used to estimate the impact of a drug on the patient's HbA1C (a measure of a patient's baseline blood glucose level), under different drug choices with regression-based methods. Data from Boston Medical Center validates their approach.

Recent work in economics, operations and statistics have moved beyond KNN to focus on developing more flexible algorithms adapted from decision trees to analyze heterogeneous treatment effects [45, 10, 155]. The basic idea is to use trees to recursively divide subjects based on their attributes and observed heterogeneity in their response. The implementation of this idea requires solving several challenges. First, the treatment effect is never simultaneously observed on the same individual, and, therefore, it is not obvious how to construct a loss function for tree splitting. Second, treatments on subjects are often endogenous, or subject to selection bias. [10] and [155] overcome these challenges — under the conditional independence assumption — by modifying the loss function used for tree splitting in CART. [156] and [157], extend this literature by incorporating instrumental variables into the causal tree framework proposed in [10] to correct for potential endogeneity bias. They validate their approach by examining heterogeneous health care outcomes of cardiovascular surgical procedures.

Another important application of ML in causal inference is to generate variables that researchers may use as dependent or independent variables. This is especially valuable if the researchers are using unstructured data, such as image, text, audio and video, or when labeling is costly. For example, in [167], the authors assess the impact of having an AirBnB room listing's photos verified by a professional photographer. Using a difference-in-differences approach, they find that the effect of photo verification is positive and significant on room demand. To separate the effect of photo quality from the effect of verification, the authors build a supervised learning algorithm that can predict an image's aesthetic quality for a large number of images. This allows the authors to show that improving photo quality in listings alone can result in significant additional revenue. As another example, in [54], the authors use natural language processing techniques to label the sentiment of Facebook posts and comments, in order to forecast demand. Similarly, [113] uses textual information from financial documents to estimate the impact of sentiment on stock returns.

Researchers also utilize ML models to directly estimate average treatment effects when units in the control conditions are rare [153] or the response function is highly nonlinear [41]. For example, [153] proposed a method for constructing synthetic control units based on supervised learning. In particular, when there are only treated unit observations in a time period, one can instead *predict* the counterfactual under the control arm using the covariates of the treated units. Comparing the results allows estimation of the treatment effect. In a similar spirit, [41] applies ML on high-frequency panel data to forecast the counterfactual energy consumption paths of schools in the absence of any energy-efficiency investments. This enables the authors to study the treatment effect of energy-efficiency investments. The authors compare their method with standard panel fixed-effect approaches and find that the latter's estimates are sensitive to the set of observations included as controls and to the fixed effects included in the specification, while ML methods are substantially more stable.

3.3. Supervised Learning for Prescriptive Analysis

A salient characteristic of data analytic work in OM is its focus on transforming raw data into *prescriptive* operational decisions.

3.3.1. Prediction, then Prescription Some work uses a "prediction, then prescription" approach; here, an ML model is trained in the first stage, and then, its predictions are utilized in an optimization problem for decision making in the second stage.

A classical example is the assortment optimization problem under the multinomial logit (MNL) model, in which we want to determine the optimal assortment of products to be offered to a consumer in order to maximize total revenue. Under the MNL model, the purchase probability of a product given the assortment is described by a multi-class logistic regression model. While the model training can be easily solved in a standard way—i.e., a gradient descent algorithm on the logit loss function—one requires an operational lens to determine how to optimize the resulting assortment. [147] and [131] utilize the special structure of the problem and show that it can be solved very efficiently.

But would this approach offer advantages over other ML models in practice where the optimization model is simpler and less structured? [94] provides an affirmative answer by conducting a large-scale field experiment on Alibaba for finding the optimal set of products to display to customers landing on Alibaba's online marketplaces. Alibaba uses a sophisticated ML algorithm to estimate the purchase probabilities of each product for each consumer, trained with thousands of product and customer features. [94] shows that, despite the lower prediction power of the MNL-based approach, it in fact generates significantly higher revenue per consumer visit, largely due to the closer integration of MNL with the downstream optimization problem.

In a similar vein, [67] use ML to estimate the demand of new products for online fashion retailers in promotion events. They use an interpretable regression tree model trained with many features (e.g., product price, relative prices of competing products) to perform demand forecasting, and use the resulting model to optimize prices. Due to the non-parametric nature of the regression tree model and the cross-dependence of products' demand and price, a naive formulation of the pricing problem would require an exponentially large decision variable space. Nevertheless, leveraging the structure of the tree model, the authors transform the problem into an integer optimization problem with a much smaller variable space and develop an efficient algorithm to solve the problem. For optimization under generic trees ensemble models, if decision variables in the optimization problem are also used as independent variables, [120] shows that one can design optimization algorithms that are based on a mixed-integer linear program and perform well even for large-scale instances. [30] studies a similar problem under the random forest model.

[76] describes how a tree-based prediction together with optimization can be used in the study of optimizing spatio-temporal location optimization. [112] minimizes the delay in last-mile delivery services, using delivery data and ML models to predict uncertain driver travel times, which affect optimal order assignments. The authors identify several predictors for travel time, which unfortunately are influenced by the order assignment decision; this makes the multiperiod order assignment problem particularly challenging.

The authors discuss classes of tractable prediction models as well as optimization reformulations that can be efficiently solved.

[6] goes a step further, and combines ML, causal inference, and optimization towards improved operational decision making in a revenue management application. The goal is to estimate price sensitivity when pricing tickets in a secondary market. Because of the heterogeneous nature of tickets, the unique market conditions at the time each ticket is listed, and the sparsity of available tickets, demand estimation needs to be done at the individual ticket level. The paper introduces a double/orthogonalized ML method for classification that isolates the causal effects of pricing on the outcome by removing the conditional effects of the ticket and market features. Furthermore, the paper embeds this price sensitivity estimation procedure into an optimization model for selling tickets in a secondary market.

3.3.2. Better Prescriptiveness Instead of taking a predict-then-optimize approach, an important facet of some recent work is directly incorporating ML models into optimization, which often leads to superior prescriptions. This is because ML tools are typically designed to reduce prediction error without taking into account how the predictions will be used, i.e., good out-of-sample prediction error does not necessarily coincide with good out-of-sample decisions.

[26] demonstrates this point by studying a stochastic optimization problem with historical data $\{x_i, y_i\}_{i=1}^N$. After observing a new feature vector $X = x$, the decision maker makes a choice z to minimize $\mathbb{E}[c(z;Y)|X = x]$, where $c(\cdot)$ is some known function. A typical example under this framework is the classical inventory ordering problem: Y denotes the uncertain demand, X is some auxiliary observable that can be used for demand forecasting, and z is the order quantity. Also, $c(\cdot)$ describes the total inventory cost.

A traditional approach would be to first use $\{x_i, y_i\}_{i=1}^N$ to build a point forecast of $\hat{Y}$, and then minimize $c(z;\hat{Y})$. As pointed out in [26], this approach ignores the uncertainty around $\hat{Y}$ and can lead to sub-optimal decisions. Instead, the authors propose to choose a decision

$$\hat{z}_N^{\text{local}}(x) \in \arg\min \sum_{i=1}^{N} w_{N,i}(x) c(z; y_i),$$

in which $w_{N,i}(x)$ is a weight assigned to observed instance i (the weights are larger for x_i's that are closer to x). Within this framework, several well-known supervised learning models can be used to find the weights, such as KNN [70], local linear regression [49], CART [36], or random forest [35]. The authors show that the proposed approach improves the quality of the decision significantly.

[61] proposes a general framework called Smart "Predict, then Optimize" (SPO) to better integrate prediction and prescription. In SPO, the loss function in the ML training process takes into account the decision error in the downstream optimization problem induced by ML prediction. To handle the computational challenge in training with the SPO loss, the paper proposes a surrogate loss that is tractable and is statistically consistent with the SPO loss. The authors shows that this new estimation approach leads to decisions that

exhibit significant improvement over those derived from traditional methods in several classical optimization problems. [116] extends SPO to solve large-scale combinatorial optimization problems, such as the weighted knapsack and scheduling problems.[62] focuses on training decision trees under the SPO framework and proposes a tractable and interpretable methodology. Relatedly, [47] adapts decision tree models for optimal stopping problems.

4. Unsupervised Learning

Unlike supervised learning, unsupervised ML algorithms deal with data sets without reference to known, or labeled, outcomes. A common theme in unsupervised learning algorithms is to detect the underlying structure of the data that are previously unknown. In this section, we first introduce the general concepts in unsupervised learning and then discuss the important use cases of unsupervised learning in the OM literature.

4.1. General Introduction to Unsupervised Learning

A widely applied class of unsupervised learning algorithms is the *clustering analysis*. The goal of such analysis is to group a set of data points, $\{\mathbf{x}_i\}_{i=1}^n$, such that those data points in the same group or cluster are more similar to each other than to those in other clusters. Similarity between two data points i and j is measured by some notion of distance, for example the Euclidean distance $\|\mathbf{x}_i - \mathbf{x}_j\|_2$. Often, such clusters represent data groups with distinctive characteristics and, therefore, form a logical structure, on which to base deeper nuts-and-bolts analysis and to design operational policies.

Classical examples of clustering include hierarchical clustering methods, which date back to the 1960s. For example, one may initialize each data point as a cluster and build a tree in a bottom-up fashion by merging the clusters that are similar. This leads to the well-known hierarchical agglomerative clustering (HAC) algorithm [158]. Another widely used algorithm of clustering analysis is the k-means clustering algorithm. Taking k as an input to the algorithm, it divides the data into k clusters by iterating between two steps until convergence. In the first step, given the assignments of data points to the k clusters, we calculate the center of each cluster. In the second, given the center of each cluster, we assign each data point to the cluster whose center is the closest to that data point. The choice of parameter k is usually subjective, but the overarching principle is to strike a balance between the in-cluster-similarity and model complexity. The k-means clustering falls into the categories of centroid-based clustering, in which a cluster is defined by a center. Other well-known algorithms within this family include the k-medroids algorithm [101], the k-Harmonic means algorithms [165], and the fuzzy c-means algorithm [29]. The key difference among these algorithms lies in how to define the centers and how to determine cluster assignments. A criticism of these methods is that they tend to favor clusters that are sphere-like, and have great difficulty with anything else. This motivates other algorithms, such as the density-based spatial clustering of applications with noise

(DBSCAN) algorithm, which can give arbitrary-shaped clusters and requires no prior knowledge of the number of clusters [63].

Another broad class of unsupervised learning algorithms are *latent variable models*, which assume that the observable data are generated as the result of some underlying latent variables. Local independence is often assumed, meaning that once these latent variables are controlled for, the variables manifested in the data are purely randomly generated with noise. Depending on the purpose of the analysis, the goal is either to uncover the data-generating process or to pinpoint the latent variables. One important example in latent variable models is mixture of models. In such models, we assume that each data point $\mathbf{x_i}$ is generated by one of the several underlying distributions without knowing the actual distributions and the membership of data points. The membership of data points to these distributions are the latent variables we do not observe. We are usually interested in these latent variables, as well as in the parameters of the distributions. The starting point to estimate such a model is to note that different specifications of the underlying distributions lead to a different likelihood of the observed data. Therefore, one may resort to the maximum log-likelihood estimation (MLE) method in the statistics literature. The central difficulty, however, is that since the latent variables are not observed, the MLE on marginal distributions is usually hard to optimize. The famous expectation–maximization (EM) algorithm solves this issue by alternating between performing an expectation (E) step, which creates a function for the expectation of the full log-likelihood function using the current estimate for the parameters and the posterior distribution of latent variables, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found in the E step [58, 123]. One can show that EM necessarily leads to a (local) maximum likelihood. Alternatively, one can also estimate such models with moment matching [57]. Another well-known model in this class is the hidden Markov chain model (HMM): one observes sequential data that are assumed to be governed by an unobservable Markov chain [19]. The parameters of the HMM can also be recovered by the EM algorithm.

Using the concept of latent variable models, we often want to infer a latent but much simpler or structured representation of the complex, unstructured, or high-dimensional data we observe. This is referred to as *representation learning*, which can serve as the basis for building classifiers or other predictors. For instance, this can be particularly useful when dealing with the curse of dimensionality. In predicting whether a consumer will click on an advertisement, we may have more features than the size of the data set. Therefore, one would wish to work with low-dimensional data while keeping the primary information in the original data, and this is where such learning techniques come in handy. A canonical analysis for representation learning is principal component analysis (PCA), in which we assume that the high-dimensional data are generated by a linear combination of low-dimensional latent variables that are orthogonal to each other. Based on the eigenvalue decomposition of the covariance matrix, one can approximately recover these low-dimensional components. Alternatively, one can use singular value decomposition or matrix factorization with Markov-Chain Monte-Carlo. Recent developments in representation learning for applications such as

speech recognition, signal processing and natural language processing highlight the role of deep neural networks. For example, [22] uses deep neural networks to learn a distributed representation for each word, called a word embedding. For a review of representation learning, please refer to [21].

Other applications of unsupervised ML techniques include *anomaly detection*, which can automatically discover unusual data points to pinpoint fraudulent transactions, discover faulty pieces of hardware, or identify an outlier caused by a human error; or *association mining*, which identifies patterns that frequently occur together in the data and is frequently used by retailers for basket analysis to discover goods often purchased at the same time. A recent milestone of using deep learning methods for unsupervised learning is the generative adversarial network (GAN), which, given a training set, learns to generate new data statistically similar to data in the training set [78]. For example, a GAN can be used for voice impersonation, mimicking not only the pitch and other perceivable signal qualities, but also the style of the target speaker. It is proven very useful for strengthening the performance of both supervised learning and reinforcement learning algorithms. These unsupervised ML techniques are not directly relevant to the OM literature (so far), so we will not review their details (please refer to Table 2).

Class	Model	Papers
Clustering	Hierarchical Agglomerative Clustering (HAC)	[158]
	k-Means Clustering	[115]
	k-Medoids	[101]
	k-Harmonic Means	[165]
	Fuzzy c-Means	[29]
	Density-Based Spatial Clustering of Applications with Noise (DBSCAN)	[63]
latent variable models	Mixture of Models	[57]
	Expectation Maximization (EM) Algorithm	[58, 123]
	Hidden Markov Chain Model (HMM)	[19]
Representation Learning	Principal Componant Analysis (PCA)	[127]
	Word Embedding	[22]
Other Unsupervised Models	Anomaly Detection	[38, 106]
	Association Ruling Learning	[2]
	Autoencoder	[108]
	Generative Adversarial Network (GAN)	[79]
	Deep Belief Network	[85]

Table 2 Some well-known unsupervised learning models and seminal papers.

4.2. Unsupervised Learning for Descriptive Analysis

4.2.1. Unsupervised Learning for Prediction Unsupervised learning techniques often play an important role in structuring the data to aid predictions. For example, clustering analysis allows the analyst to "divide and conquer" in prediction tasks — once data are appropriately grouped, one can customize the

prediction model for each group to achieve higher accuracy. Predictions based on clustering often outperforms naive implementation of predictions. This research strategy is adopted in [111]. The authors consider the forecast of product returns based on return merchandise authorization (RMA) information. They first conduct a clustering analysis to segment customers based on their historical RMA records and then resort to counting regression models to generate a forecast for each customer cluster. In this process, the clustering analysis allows the authors to fully exploit customer heterogeneity and leads to improved forecast accuracy in comparison with two benchmark models. This point is further illustrated by a paper [91] in the setting of demand forecasts for electronic retailers. The authors present a cluster-then-predict approach to forecast customer orders of new products that are similar to past products, which leads to mean absolute errors approximately 2%–3% below the partner firm's existing forecasts. The same clustering method is adopted by [142] for exploratory analysis, which investigates the impact of different cross-border fulfillment options, such as the fulfillment by Amazon (FBA) option and in-house fulfillment on the sales and bottom line of e-retailers. In a similar vein, [50] also utilizes clustering for the prediction of customer demand. More specifically, [50] studies the demand forecasting problem for retailers in a scenario in which certain products have a large amount of historical data, while others are newly introduced and have scarce demand-related data. The authors propose a procedure that first applies the maximum likelihood estimation approach to estimate a different coefficient vector for each product, and then combines hypothesis testing and k-means clustering to identify the correct aggregation level for each coefficient.

Latent variable models are also used in prediction problems in the OM literature. For example, in [137], the authors develop dimension-reduction methods for forecasting incoming call volumes. Their approach is to treat the intra-day call volume profiles as a high-dimensional vector time series. They propose, first, to reduce the dimensionality by singular value decomposition of the matrix of historical intraday profiles and, then, to apply time series and regression techniques. The authors show that their method is very competitive in out-of-sample forecast comparisons using two real data sets. The hidden Markov model (HMM) is used in [122] to identify unobserved on-shelf out-of-stock (OOS) by detecting changes in sales patterns resulting from unobserved states of the shelf. They identify three latent states, one of which characterizes an OOS state, specify the model using a hierarchical Bayes approach, and use a Monte Carlo–Markov chain methodology to estimate the model parameters. Their HMM approach performs well in predicting out-of-stocks, combining high detection power (63.48%) and low false alerts (15.52%). A recent paper [42] introduces Product2Vec, a method based on the representation learning technique Word2Vec, as discussed in [119], to study product-level competition when the number of products is large. Their model takes consumer shopping baskets as inputs and, generates a low-dimensional vector for every product that preserves important product information.

Some particularly interesting research work that combines latent variable models and clustering analysis for prediction is presented in [96]. The authors study the problem of segmenting a large population of customers into diverse clusters based on customer preferences, using preference observations such as purchases,

ratings and clicks. In real applications, the universe of items can be excessively large and unstructured, while individual customer data are highly sparse, which prevents the applicability of existing techniques in marketing and ML. Their proposed method proceeds in two steps: *embed* and *cluster*. In the *embed* step, they squeeze individual customer data into the low-dimensional space by calculating the likelihoods of each customer's observed behavioral data regarding each of a small number of product categories under a representative probabilistic behavior model. In the *cluster* step, the clustering analysis is applied to low-dimensional embedding. The authors derive the necessary and sufficient conditions to guarantee asymptotic recovery of the true segments under a standard latent class setup, and they show empirically that their method outperforms standard latent variable methods.

4.2.2. Using EM Algorithms in Choice Model Estimation As we reviewed in Section 4.1, an important technique to deal with latent variable models is the EM algorithm. While this algorithm is derived from solving unsupervised learning problems, it has been widely used in the choice modeling literature in OM. In choice modeling, while customer purchases are often observed through transaction or point-of-sales data, customers who enter the store but do not purchase are often not observable to researchers, especially in brick-and-mortar retailing. Such missing data can be viewed as latent variables, and there is a large stream of OM literature that builds different EM algorithms to estimate customers' arrival process.

This literature started with the seminal work [154] that proposes a method for estimating substitute and lost demand when only sales and product availability data are observable, and only a selected subset of items are offered to the consumer in each period. The model considers an MNL choice model with a non-homogeneous Poisson model of consumer arrivals and applies the EM algorithm to estimate the model parameters; it does so by treating the observed demand as an incomplete observation of the demand that would have been observed if all products had been available in all periods. It shows that all limit points of the procedure are stationary points of the incomplete data log-likelihood function.

Subsequently, the EM algorithm is used to estimate the parameters of various choice models. In [139], the authors adapts the EM algorithm to estimate the parameters of the Markov chain choice model. The parameters of the Markov chain choice model are the probability that the customer arrives in the system to purchase each one of the products and the transition probabilities. The authors treat the path that a customer follows in the Markov chain as the latent variables. For the E step, they show how to compute the probability of consumer purchase and the expected number of transitions from a certain product to another, conditional on the final purchase decision of a customer. For the M step, they show how to efficiently solve the optimization problem that appears in the M step. Numerical experiments demonstrate that their algorithm, together with the Markov chain choice model, leads to better predictions of customer choice behavior compared with other commonly used choice models. Several other authors consider the estimation of the rank-based model in which each customer has a ranked list of products in mind and he purchases the most preferred available product with the EM algorithm [64, 95, 97, 151, 152].

In summary, EM algorithms have proven to be effective in the estimation of many choice models. In fact, they have become an off-the-shelf method that one may consider for dealing with choice model parameter estimation.

4.3. Unsupervised Learning for Prescriptive Analysis

In this section, we will review several recent works on applying unsupervised learning for prescriptive analysis. This literature is still relatively young and sparse, but we expect it to quickly grow in the near future, in view of the wide application of unsupervised learning in other fields.

[24] investigates the personal assortment optimization when there are heterogeneous customers with unknown product preferences and known features. The authors consider a dynamic clustering policy embedded into an exploration-exploitation framework with MNL as the assumed choice model. The clustering policy aims to learn both the underlying mapping of profiles to clusters and the preferences of each cluster with the Dirichlet process mixture model, which can be estimated by a Markov Chain Monte Carlo (MCMC) sampling scheme. The case study presented in [24] suggests that the benefits of such a strategy are substantial. Compared with a data-driven policy that treats customers independently and a linear-utility policy that assumes that product mean utilities are linear functions of available customer attributes, it generates over 37% and 27% more transactions, respectively.

Another recent work [80] studies a multi-location newsvendor network when only first- and second-moment information of demand is known by using a distribution-robust model to find inventory levels that minimize the worst-case expected cost among the distributions consistent with this information. The authors show that the problem is NP-hard, but they develop a computationally tractable upper bound on the worst-case expected cost if the costs of fulfilling demands follow a nested structure. They propose an algorithm that can approximate general fulfillment cost structures by nested cost structures, which gives a computationally tractable heuristic for the optimization problem. To show that nested structures offer sufficient modeling flexibility, they develop a simple algorithm that stems from the HAC algorithm to approximate any general distance-based fulfillment cost structure as a nested cost structure.

5. Bandits and Reinforcement Learning

In both supervised and unsupervised learning, the training dataset is assumed to be provided through an exogenous process. However, in general, the decision maker's actions can *affect* the data that we observe, thereby affecting the quality of the downstream predictive model. For instance, let us return to our earlier example on supervised learning for advertising: $\mathbf{x}_i$ describes the characteristics of a customer who is offered an advertisement, and $y_i = 1$ if the consumer clicks on this advertisement. Yet we observe that this dataset is available *only if* the platform shows some customers this specific advertisement. If not, the platform does not observe the resulting customer response, and consequently, cannot make predictions for new customers. In other words, the platform's current decisions directly affect the data it observes, and thereby its ability to

make good decisions in the future. Multi-armed bandits (MAB) and reinforcement learning (RL) provide a general framework and near-optimal algorithms for making these types of sequential decisions in the face of uncertainty.

5.1. Multi-Armed Bandits

The term "multi-armed bandits" comes from a stylized gambling scenario in which a gambler faces several slot machines that yield different payoffs. In the classical multi-armed bandit, the decision maker has access to K actions (referred to as "arms"). As a running example, let each arm be a new treatment for a disease. Each arm i is associated with an expected reward μ_i—i.e., the expected improvement in patient outcomes from this treatment. We sequentially observe T homogeneous new patients, and the decision maker chooses one arm for each patient. Upon assigning a patient a treatment, she immediately observes a noisy signal of the treatment's reward μ_i — thus, if she assigns this arm to a sufficient number of patients, she can obtain a good estimate $\hat{\mu}_i$ of the arm's expected reward. This is termed *exploration*—i.e., a good policy must try each arm in order to learn if it is promising. However, to maximize long-term performance across all patients, a good policy must also *exploit* the data it has accumulated thus far, and offer the estimated best arm to patients. This *exploration-exploitation tradeoff* underlies the design of all bandit algorithms.

Table 3 provides some common applications of bandits, as well as their corresponding actions (arms) and the reward (feedback).

Application	Action	Reward
clinical trials	which drug to prescribe	patient outcome
web design	font color, page layout, etc.	#clicks, engagement time, etc.
content optimization	featured items/articles	#clicks, engagement time, etc.
web search	search results for a given query	user satisfaction
advertising	which ad to display	revenue from ads
recommender systems	which products to recommend	user satisfaction
dynamic pricing	product prices	revenues
procurement	which items to buy	utility from items and costs
auction/market design	which reserve price to use	revenue
crowdsourcing	which tasks to give to which workers	quality of completed tasks
datacenter design	which server to route the job to	job completion time
Internet	which TCP settings to use	connection quality
radio networks	which radio frequency to use	rate of successful transmission

Table 3 Some common applications for multi-armed bandits, adapted from [140].

The celebrated Gittins indices [75] show that the optimal solution to an infinite-horizon Bayesian multi-armed bandit satisfies an index-based policy. However, this solution is generally computationally intractable, and thus, the near-optimal and computationally simpler Upper Confidence Bound (UCB) algorithm [12] became the workhorse bandit algorithm. UCB leverages the "principle of optimism" to trade off exploration and exploitation: when uncertain about an arm's reward, act assuming that it has the highest possible

reward based on its confidence set. This approach naturally induces exploration since arms that have few observations will have large confidence sets; and at the same time, it avoids wasteful exploration since only arms that have some reasonable probability of being the best will be played. Recently, Thompson Sampling has risen to prominence, demonstrating superior empirical performance as well as similar near-optimality guarantees [148, 132]

There are numerous variants of the classical multi-armed bandit described above. Of particular importance to the OM community, the contextual bandit framework is an important extension, allowing decision makers to *personalize* decisions. In this setting, individuals at each time t are associated with a feature vector X_t that captures individual-specific information that may affect their response to the choice of arm (e.g., customer or patient histories). Each arm is then associated with an unknown function $f : X \to Y$, which maps from the individual's feature vector to her observed reward. The most widely studied setting is where f is a linear (or generalized linear) model [1], but there has also been work studying nonparametric but smooth choices for f [81]. Surprisingly, [16] shows that, when there is sufficient randomness in the observed features, the exploration-exploitation tradeoff disappears. This is because randomness in the features can induce *free exploration*, allowing a greedy algorithm to perform comparably to or better than bandit algorithms that explore.

5.1.1. Popular Variants The bandit framework makes a number of assumptions that may not hold in practice. Thus, as bandit algorithms are increasingly being deployed in practice (for A/B testing, recommender systems, etc.), a number of variants have been proposed to bridge the gap between theory and practice.

For instance, the rewards of arms may *change* over time (e.g., news articles or fashion products may become outdated). Policies that "forget" old data have been designed for this non-stationary environment [28].

Outcomes may not be observed immediately; for example, in a clinical trial, it may be many months before the patient's outcome is observed [7]. Good policies must account for assignments that are already in the pipeline but are yet to be observed when designing their exploration strategy [100].

The observed individual-level features may be *high-dimensional*. In this case, one generally cannot design good policies unless there is additional structure. A popular assumption to impose is that the arm rewards are *sparse*—i.e., only a subset of the many observed features is predictive of the rewards. Then, [15] bridges high-dimensional estimation and bandits to design good policies in this setting.

In many recommender systems, the arms may not be a single product, but a combinatorial *assortment* over a set of products. In this case, whether or not the user clicks on an item depends not only on the item, but also on the other items in the assortment set. This type of feedback has been studied in the classical MNL choice model. [3] bring these ideas to the bandit setting. Another issue with recommender systems is

that users may *disengage* from the platform if they are initially offered poor recommendations. [17] shows that it may be favorable in these instances to use customer information to constrain exploration upfront.

In practice, there may be a number of constraints that govern the decision-making process. For instance, in ad allocations, a firm may request the platform for a minimum expected click-through rate, or in resource allocation problems, the decision maker may face capacity or budget constraints. [4] studies policies under general global convex constraints and concave objective.

5.1.2. Dynamic Pricing Consider a monopolist that seeks to dynamically price a new product with unknown market demand. This problem suffers from the semi-bandit feedback: if a customer purchases a product at price p, one can infer that she would have purchased it at any lower price $p' < p$, but we do not observe her purchase decision for higher prices $p' > p$; conversely, if a customer does *not* purchase a product at price p, we do not know if she would have purchased it for any lower prices. Thus, the decision maker must navigate an exploration-exploitation tradeoff in her pricing decisions in order to learn the market demand and converge to an optimal price. This particular problem and its variants have received significant attention in the OM community [103, 105].

In collaboration with the fashion e-retailer Rue La La, [67], which we have discussed in Section 3.3, also demonstrates the potential economic value of dynamic pricing policies in practice, particularly for new products with uncertain demand and stringent inventory constraints. Yet, in practice, a large platform must solve the pricing problem over a large number of products. In this case, one may wish to leverage contextual information in the form of product features [51]. However, the variation in product demand may not be captured by product features alone: for example, there may be different demand for two different black dresses due to factors that are hard to measure such as fit or style. For these settings, [18] adopts an empirical Bayes approach to learn a shared prior across demand parameters for related products.

5.2. Reinforcement Learning

In the multi-armed bandit setting, while the actions taken by the decision maker affect the data they observe, they do not affect the world. However, this assumption does not hold in many settings. For instance, if a platform shows too many advertisements to a consumer, she may decide to leave the platform. In this case, decisions made at the current step have long-term consequences beyond the immediate reward collected. Thus, decision makers not only need to address the exploration-exploitation tradeoff, but must also account for how their actions affect the underlying state of the world. This property is characteristic of many practical problems, including queuing (where the state encodes the current capacity of the queue), inventory management (where the state encodes the amount of inventory currently available), and dynamic pricing (where the state encodes the remaining inventory). Reinforcement learning algorithms are designed to solve these kinds of problems [56, 74].

The standard framework for formalizing a reinforcement learning problem is a *Markov decision process (MDP)*. At a high level, an MDP consists of a set of states representing the possible states of the world. For instance, the state might represent whether the customer has left the platform. Similar to multi-armed bandits, an MDP also has a set of actions that the decision maker can take. It also has a reward function; in contrast to the multi-armed bandits setting, rewards now depend not only on the action taken but also on the current state of the system. For instance, if the customer has already left the platform, then the reward is zero regardless of what action is taken. Finally, an MDP also has a (probabilistic) transition function that describes how the state changes depending on the action taken. For instance, the customer might leave the platform with higher probability if they are shown an advertisement.

The goal of reinforcement learning is to compute the optimal policy, which specifies the action to take in each state in a way that maximizes the cumulative reward collected over the time horizon of the problem. For infinite-horizon problems, a discount factor is applied to future rewards to ensure that the cumulative reward collected is finite. When the MDP reward function and transition function are known, the decision maker observes the current state, and the state and action spaces are finite and small; thus, we can efficiently solve for the optimal policy using value iteration (also known as dynamic programming) [20]. At a high level, this approach characterizes the optimal policy in terms of the value function encoding the optimal cumulative reward achievable at each state, establishes a recursive equation, known as Bellman's equation, characterizing the value function, and then solves this equation using an iterative procedure that is guaranteed to converge.

There are three reasons that this approach may no longer work. The first is due to the *curse of dimensionality*. When the state space is very large or continuous, then value iteration is no longer computationally tractable. In this case, approximate dynamic programming algorithms for solving the MDP have been studied; these algorithms are not guaranteed to compute the optimal policy, but often work very well in practice. Continuous states remain a major challenge, but are common in predictive analytics applications; for instance, they might encode the current customers' attributes that can be used to predict the probability they will leave the platform.

The second is due to *partial observability*. The decision maker may not always observe the state of the world; for instance, he might not know immediately whether the customer has left the platform or temporarily become unresponsive. In this case, the problem must instead be formalized as a *partially observed MDP (POMDP)*. Algorithms exist for solving for the optimal policy in a POMDP, though do not scale nearly as well as value iteration.

The third reason is that the transition and reward functions are unknown. For instance, the decision maker might not know ahead of time the probability that the customer will leave the platform if shown an advertisement. Instead, the decision maker must learn this information by taking exploratory actions; thus, this setting combines bandit feedback with MDPs. Reinforcement learning typically refers to this setting;

however, as we discuss below, certain reinforcement learning algorithms can be useful for addressing the previous two issues as well.

Reinforcement learning algorithms can be divided into two approaches: model-based and model-free. Intuitively, a model-based algorithm is one that first estimates the MDP transition and reward functions, and then uses value iteration to compute the optimal policy based on these estimates. As the estimates converge to the true transition and reward functions, the computed policy converges to the true optimal policy. Most algorithms that provide theoretical guarantees are model-based. [102] introduced the E^3 algorithm, which was the first provable near-optimal polynomial time algorithm for learning in MDPs. [33] proposed the much simpler R-MAX algorithm, which formally justified a solution strategy similar to bandits, based on optimism under uncertainty. This work then paved the way for improved algorithms for reinforcement learning based on UCB [11] and Thompson Sampling [5].

In contrast, model-free algorithms avoid estimating the transitions and rewards; instead, they directly learn a representation of the policy. These algorithms tend to be less sample-efficient compared to model-based algorithms. However, a key benefit of these approaches is that they are very general since they are agnostic to the structure of the MDP; for instance, they readily apply to MDPs with large or continuous state spaces. Model-free algorithms can furthermore be subdivided into two kinds. First, Q-learning [159] is an algorithm that aims to learn the Q-function, which encodes the cumulative reward achieved by taking an action in a given state and then acting optimally thereafter. The Q-function implicitly encodes the optimal policy since we can greedily choose the action with the highest value according to the Q-function in the current state. The Q-function can be learned using the method of temporal differences [143], which establishes a recursive formula for the Q-function based on Bellman's equation. In the finite state setting, there has been recent work proving that Q-learning can efficiently recover the optimal policy [141, 98]. Second, policy optimization algorithms rely on directly optimizing the policy. A classical approach is to use a parametric policy class such as neural networks, and then to use gradient descent on to optimize these parameters [161]. Recent work has proposed additional improvements, including actor-critic approaches that combine policy optimization with Q-learning [144, 107, 121] and algorithms that rely on trust region optimization [136]. Policy optimization algorithms are very general; for instance, unlike Q-learning, they can be applied to POMDPs. The drawback of these algorithms is that they are highly susceptible to local minima, since the optimization problem tends to be highly non-convex.

6. Future Directions

We envision that the increasing availability of data will continue to drive the development of ML-based research in OM. Before we close this chapter, we identify several research areas at the intersection of ML and OM that we believe will continue to attract more research attention. We will start with supervised learning.

First, a large amount of observational data from actual business practice is being accumulated in many business disciplines and is gradually being shared with business researchers [138, 166]. Such data often

include a large number of covariates available to describe the context of the data. Combined with advanced ML techniques, we think that the causal inference literature will focus increasingly on heterogeneous treatment effects (HTE) and provide personalized decision making in operations, such as individualized medical choices or personalized assortment optimization/pricing. Two major difficulties are how to utilize the representation power of ML tools to infer treatment effects for heterogeneous individuals that can overcome the endogeneity issue universally found in observational data; and, how to overcome the high-dimensionality in data. Previously mentioned work [156] and [157] aim to address the first issue. However, there is still a lot of room for new developments. For example, existing approaches in HTE analysis deal mainly with cross-sectional settings, while a large body of causal inference literature has demonstrated the value of exploiting temporal variation on an individual level for average treatment effect analysis. Therefore, an important research direction is to develop ML-based HTE estimators tailored to *panel data*. Moreover, most work in HTE analysis uses tree-based methods. Can other techniques—e.g., *deep learning*—that are known to be more powerful when more training data are available, be used for such purposes? Furthermore, advancements have been made recently in applying ML techniques to identify critical covariates in *high-dimensional data* for causal inference, such as those discussed in [44]. Can we tailor these methods to OM problems?

Second, we also expect that OM researchers will utilize more recent ML developments to improve traditional optimization techniques. In Section 3.3, we discussed the literature on incorporating certain ML models into optimal decision making. Nevertheless, other ML models, especially those developed recently with more prediction power, can also be considered for such purposes. For example, it is sensible that models such as deep learning can better describe consumer choice behaviors than such classical choice models as MNL, but it is far from trivial to figure out solution techniques for optimization problems such as assortment optimization or pricing under such models.

Third, we also anticipate that OM researchers will contribute to the ML literature by developing prescriptive systems with more business constraints. Price fairness [25] and discrimination [55] has been long considered in the OM literature. Such topics become more and more essential in developing prediction and optimization algorithms using ML, since important decisions in our society, such as information acquisitions and hiring, are increasingly made by algorithms. Therefore, we believe that OM researchers will contribute to the ML literature by working on business applications that utilize supervised learning methods with business and ethnics constraints, such as fairness and privacy.

With regard to supervised learning, a promising research direction is to utilize its power in learning representation and clustering to improve field experiment analyses. While field experiments have been the “gold standard" in getting causal inference towards business and policy questions, experiments on platforms, especially two-sided platforms with limited demand or supply, or social networks suffer from interactions between units [60, 99, 164]. One exploration in this direction is found in [150]. It considers the estimation gof

lobal average treatment effect in the presence of network inference. While unbiased and consistent estimators have been constructed for this problem, these estimators suffer from extreme variance when the experiment design is poor. The authors propose a randomized graph cluster randomization that enables substantial mean squared error reduction in the estimator, as compared with existing approaches.

In terms of dynamic learning (both multi-armed bandits and reinforcement learning), one important challenge is to satisfy safety constraints—i.e., how can we ensure that exploration does not lead to damaging or irrecoverable outcomes? Several different notions of safety have been studied. One natural notion applied to platforms is specifying a set of states that should not be reached—e.g., we want to ensure that a customer does not leave the platform with high probability [17]. Fairness can also be thought of as a safety constraint—i.e., ensuring that the algorithm does not unfairly discriminate against minorities [82]. For instance, recent work has shown that in certain settings, tradeoffs exist between exploring and ensuring fairness [104]. Finally, in the setting of multi-armed bandits, recent work that has studied algorithms under the constrained exploration is *conservative*—i.e., it is guaranteed to outperform a baseline policy for the entire time horizon [162]. In general, characterizing the tradeoffs between exploration and satisfying practical constraints remains an important challenge in the field.

Another important challenge in dynamic learning is *policy evaluation*, of which the goal is to estimate the performance of a policy from historical data. Policy evaluation is needed to understand how well the bandit is doing compared to alternative approaches or baseline strategies. This line of work builds on the causal inference literature [145] to devise algorithms that produce unbiased estimates of the performance of the policy. A closely related problem is *offline learning* (or *batch learning*), where we want to actually learn the policy from observational data—i.e., data collected from another, possibly unknown, policy. For instance, we might collect patient outcomes from actions taken by a doctor; and, we want to learn a treatment policy based on these data without any active exploration. This approach is also related to safety since, in many domains, exploration is highly constrained due to ethical considerations (e.g., healthcare or legal and financial decision making).

References

[1] Abbasi-Yadkori Y, Pál D, Szepesvári C (2011) Improved algorithms for linear stochastic bandits. *Advances in Neural Information Processing Systems*, 2312–2320.

[2] Agrawal R, Imieliński T, Swami A (1993) Mining association rules between sets of items in large databases. *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, 207–216.

[3] Agrawal S, Avadhanula V, Goyal V, Zeevi A (2019) Mnl-bandit: A dynamic learning approach to assortment selection. *Operations Research* 67(5):1453–1485.

[4] Agrawal S, Devanur NR (2019) Bandits with global convex constraints and objective. *Operations Research* 67(5):1486–1502.

[5] Agrawal S, Jia R (2017) Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. *Advances in Neural Information Processing Systems*, 1184–1194.

[6] Alley M, Biggs M, Hariss R, Herrmann C, Li M, Perakis G (2019) Pricing for heterogeneous products: Analytics for ticket reselling. Working paper.

[7] Anderer A, Bastani H, Silberholz J (2019) Adaptive clinical trial designs with surrogates: When should we bother? Working paper.

[8] Ang E, Kwasnick S, Bayati M, Plambeck EL, Aratow M (2016) Accurate emergency department wait time prediction. *Manufacturing & Service Operations Management* 18(1):141–156.

[9] Angrist JD, Pischke JS (2008) *Mostly harmless econometrics: An empiricist's companion* (Princeton, NJ: Princeton University Press).

[10] Athey S, Imbens G (2016) Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences* 113(27):7353–7360.

[11] Auer P, Jaksch T, Ortner R (2009) Near-optimal regret bounds for reinforcement learning. *Advances in Neural Information Processing Systems*, 89–96.

[12] Auer P, Ortner R (2010) UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica* 61(1-2):55–65.

[13] Baardman L, Levin I, Perakis G, Singhvi D (2017) Leveraging comparables for new product sales forecasting. Working paper.

[14] Bastani H (2020) Predicting with proxies: Transfer learning in high dimension. *Forthcoming in Management Science* .

[15] Bastani H, Bayati M (2020) Online decision making with high-dimensional covariates. *Operations Research* 68(1):276–294.

[16] Bastani H, Bayati M, Khosravi K (2020) Mostly exploration-free algorithms for contextual bandits. *Forthcoming in Management Science* .

[17] Bastani H, Harsha P, Perakis G, Singhvi D (2018) Sequential learning of product recommendations with customer disengagement. Working paper.

[18] Bastani H, Simchi-Levi D, Zhu R (2019) Meta dynamic pricing: Learning across experiments. Working paper.

[19] Baum LE, Petrie T (1966) Statistical inference for probabilistic functions of finite state markov chains. *The Annals of Mathematical Statistics* 37(6):1554–1563.

[20] Bellman R (1957) A markovian decision process. *Journal of Mathematics and Mechanics* 6(5):679–684.

[21] Bengio Y, Courville A, Vincent P (2013) Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(8):1798–1828.

[22] Bengio Y, Ducharme R, Vincent P, Jauvin C (2003) A neural probabilistic language model. *Journal of Machine Learning Research* 3:1137–1155.

[23] Berger JO (2013) *Statistical decision theory and Bayesian analysis* (New York, NY: Springer Science & Business Media).

[24] Bernstein F, Modaresi S, Sauré D (2019) A dynamic clustering approach to data-driven assortment personalization. *Management Science* 65(5):2095–2115.

[25] Bertsimas D, Farias VF, Trichakis N (2011) The price of fairness. *Operations Research* 59(1):17–31.

[26] Bertsimas D, Kallus N (2020) From predictive to prescriptive analytics. *Management Science* 66(3):1025–1044.

[27] Bertsimas D, Kallus N, Weinstein AM, Zhuo YD (2017) Personalized diabetes management using electronic medical records. *Diabetes Care* 40(2):210–217.

[28] Besbes O, Gur Y, Zeevi A (2014) Stochastic multi-armed-bandit problem with non-stationary rewards. *Advances in Neural Information Processing Systems*, 199–207.

[29] Bezdek JC (1973) *Fuzzy mathematics in pattern classification*. Ph.D. thesis, Cornell University, Ithaca, NY.

[30] Biggs M, Hariss R (2018) Optimizing objective functions determined from random forests. Working paper.

[31] Bishop CM (2006) *Pattern recognition and machine learning* (Berlin, Heidelberg: Springer-Verlag).

[32] Box GEP, Jenkins G, Reinsel GC (1970) *Time Series Analysis, Forecasting and Control* (San Francisco, CA: Holden-Day, Inc.), ISBN 0816211043.

[33] Brafman RI, Tennenholtz M (2002) R-max—a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research* 3:213–231.

[34] Breiman L (1996) Bagging predictors. *Machine Learning* 24(2):123–140.

[35] Breiman L (2001) Random forests. *Machine Learning* 45(1):5–32.

[36] Breiman L, Friedman J, Stone CJ, Olshen RA (1984) *Classification and regression trees* (Monterey, CA: Wadsworth and Brooks).

[37] Breiman L, et al. (1996) Heuristics of instability and stabilization in model selection. *The Annals of Statistics* 24(6):2350–2383.

[38] Breunig MM, Kriegel HP, Ng RT, Sander J (2000) Lof: identifying density-based local outliers. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, 93–104.

[39] Buhmann MD (2003) *Radial basis functions: theory and implementations*, volume 12 (Cambridge University Press).

[40] Burges C, Shaked T, Renshaw E, Lazier A, Deeds M, Hamilton N, Hullender G (2005) Learning to rank using gradient descent. *Proceedings of the 22nd International Conference on Machine Learning*, 89–96.

[41] Burlig F, Knittel C, Rapson D, Reguant M, Wolfram C (2020) Machine learning from schools about energy efficiency. *Journal of the Association of Environmental and Resource Economists* 7(6):1181–1217.

[42] Chen F, Liu X, Proserpio D, Troncoso I (2020) Product2vec: Understanding product-level competition using representation learning. Working paper.

[43] Chen YC, Mišić VV (2019) Decision forest: A nonparametric approach to modeling irrational choice. Working paper.

[44] Chernozhukov V, Chetverikov D, Demirer M, Duflo E, Hansen C, Newey W, Robins J (2018) Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal* 21(1):C1–C68.

[45] Chipman HA, George EI, McCulloch RE, et al. (2010) Bart: Bayesian additive regression trees. *The Annals of Applied Statistics* 4(1):266–298.

[46] Chouakria AD, Nagabhushan PN (2007) Adaptive dissimilarity index for measuring time series proximity. *Advances in Data Analysis and Classification* 1(1):5–21.

[47] Ciocan DF, Mišić VV (2020) Interpretable optimal stopping. *Forthcoming in Management Science* .

[48] Cleveland WS (1979) Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association* 74(368):829–836.

[49] Cleveland WS, Devlin SJ (1988) Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association* 83(403):596–610.

[50] Cohen M, Jiao K, Zhang RP (2019) Data aggregation and demand prediction. Working paper.

[51] Cohen MC, Lobel I, Paes Leme R (2020) Feature-based dynamic pricing. *Forthcoming in Management Science* .

[52] Cortes C, Vapnik V (1995) Support-vector networks. *Machine Learning* 20(3):273–297.

[53] Croston JD (1972) Forecasting and stock control for intermittent demands. *Journal of the Operational Research Society* 23(3):289–303.

[54] Cui R, Gallino S, Moreno A, Zhang DJ (2018) The operational value of social media information. *Production and Operations Management* 27(10):1749–1769.

[55] Cui R, Li J, Zhang DJ (2020) Reducing discrimination with reviews in the sharing economy: Evidence from field experiments on airbnb. *Management Science* 66(3):1071–1094.

[56] Dai J, Shi P (2019) Inpatient overflow: An approximate dynamic programming approach. *Manufacturing & Service Operations Management* 21(4):894–911.

[57] Day NE (1969) Estimating the components of a mixture of normal distributions. *Biometrika* 56(3):463–474.

[58] Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* 39(1):1–22.

[59] Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255 (Ieee).

[60] Doudchenko N, Zhang M, Drynkin E, Airoldi E, Mirrokni V, Pouget-Abadie J (2020) Causal inference with bipartite designs. Working paper.

[61] Elmachtoub AN, Grigas P (2017) Smart "predict, then optimize". Working paper.

[62] Elmachtoub AN, Liang JCN, McNellis R (2020) Decision trees for decision-making under the predict-then-optimize framework. Working paper.

[63] Ester M, Kriegel HP, Sander J, Xu X, et al. (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD*, volume 96, 226–231.

[64] Farias VF, Jagabathula S, Shah D (2013) A nonparametric approach to modeling choice with limited data. *Management Science* 59(2):305–322.

[65] Farias VF, Li AA (2019) Learning preferences with side information. *Management Science* 65(7):3131–3149.

[66] Ferreira K, Parthasarathy S, Sekar S (2019) Learning to rank an assortment of products. Working paper.

[67] Ferreira KJ, Lee BHA, Simchi-Levi D (2016) Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Management* 18(1):69–88.

[68] Fix E (1951) *Discriminatory analysis: nonparametric discrimination, consistency properties* (San Francisco, CA: USAF School of Aviation Medicine).

[69] Freund Y (1995) Boosting a weak learning algorithm by majority. *Information and Computation* 121(2):256–285.

[70] Friedman J, Hastie T, Tibshirani R (2001) *The elements of statistical learning*, volume 1 (New York, NY: Springer Series in Statistics).

[71] Friedman N, Geiger D, Goldszmidt M (1997) Bayesian network classifiers. *Machine Learning* 29(2-3):131–163.

[72] Fukushima K (2013) Training multi-layered neural network neocognitron. *Neural Networks* 40:18–31.

[73] Gardner Jr ES (1985) Exponential smoothing: The state of the art. *Journal of Forecasting* 4(1):1–28.

[74] Gijsbrechts J, Boute RN, Van Mieghem JA, Zhang D (2019) Can deep reinforcement learning improve inventory management? Performance on dual sourcing, lost sales and multi-echelon problems. Working paper.

[75] Gittins JC (1979) Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society: Series B (Methodological)* 41(2):148–164.

[76] Glaeser CK, Fisher M, Su X (2019) Optimal retail location: Empirical methodology and application to practice. *Manufacturing & Service Operations Management* 21(1):86–102.

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

[78] Goodfellow I, Bengio Y, Courville A, Bengio Y (2016) *Deep learning*, volume 1 (MIT press).

[79] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2672–2680.

[80] Govindarajan A, Sinha A, Uichanco J (2020) Distribution-free inventory risk pooling in a multilocation newsvendor. *Management Science* .

[81] Gur Y, Momeni A, Wager S (2019) Smoothness-adaptive stochastic bandits. Technical report.

[82] Hardt M, Price E, Srebro N (2016) Equality of opportunity in supervised learning. *Advances in Neural Information Processing Systems*, 3315–3323.

[83] Hastie TJ, Tibshirani RJ (1990) *Generalized Additive Models*, volume 43 (London, UK: Chapman and Hall).

[84] He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, 770–778.

[85] Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. *Neural Computation* 18(7):1527–1554.

[86] Ho TK (1995) Random decision forests. *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, 278–282 (IEEE).

[87] Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Computation* 9(8):1735–1780.

[88] Hoerl AE, Kennard RW (1970) Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12(1):55–67.

[89] Hopfield JJ (1982) Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences* 79(8):2554–2558.

[90] Hopp WJ, Li J, Wang G (2018) Big data and the precision medicine revolution. *Production and Operations Management* 27(9):1647–1664.

[91] Hu K, Acimovic J, Erize F, Thomas DJ, Van Mieghem JA (2019) Forecasting new product life cycle curves: Practical approach and empirical analysis. *Manufacturing & Service Operations Management* 21(1):66–85.

[92] Ibrahim R, Kim SH (2019) Is expert input valuable? The case of predicting surgery duration. *Forthcoming in Seoul Journal of Business* .

[93] Ibrahim R, Kim SH, Tong J (2020) Eliciting human judgment for prediction algorithms. *Forthcoming in Management Science* .

[94] Jacob F, Zhang D, Liu X, Zhang N (2020) Customer choice models versus machine learning: Finding optimal product displays on alibaba. *Forthcoming in Operations Research* .

[95] Jagabathula S, Rusmevichientong P (2017) A nonparametric joint assortment and price choice model. *Management Science* 63(9):3128–3145.

[96] Jagabathula S, Subramanian L, Venkataraman A (2018) A model-based embedding technique for segmenting customers. *Operations Research* 66(5):1247–1267.

[97] Jagabathula S, Vulcano G (2018) A partial-order-based model to estimate individual preferences using panel data. *Management Science* 64(4):1609–1628.

[98] Jin C, Allen-Zhu Z, Bubeck S, Jordan MI (2018) Is q-learning provably efficient? *Advances in Neural Information Processing Systems*, 4863–4873.

[99] Johari R, Li H, Weintraub G (2020) Experimental design in two-sided platforms: An analysis of bias. Working paper.

[100] Joulani P, Gyorgy A, Szepesvári C (2013) Online learning under delayed feedback. *International Conference on Machine Learning*, 1453–1461.

[101] Kaufman L, PJ R (1987) Clustering by means of medoids. Delft university of technology technical report.

[102] Kearns M, Singh S (2002) Near-optimal reinforcement learning in polynomial time. *Machine Learning* 49(2-3):209–232.

[103] Keskin NB, Zeevi A (2014) Dynamic pricing with an unknown demand model: Asymptotically optimal semi-myopic policies. *Operations Research* 62(5):1142–1167.

[104] Kleinberg J (2018) Inherent trade-offs in algorithmic fairness. *Abstracts of the 2018 ACM International Conference on Measurement and Modeling of Computer Systems*, 40–40.

[105] Kleinberg R, Leighton T (2003) The value of knowing a demand curve: Bounds on regret for online posted-price auctions. *44th annual IEEE symposium on foundations of computer science, 2003. Proceedings.*, 594–605 (IEEE).

[106] Knorr EM, Ng RT, Tucakov V (2000) Distance-based outliers: algorithms and applications. *The VLDB Journal* 8(3-4):237–253.

[107] Konda VR, Tsitsiklis JN (2000) Actor-critic algorithms. *Advances in Neural Information Processing Systems*, 1008–1014.

[108] Kramer MA (1991) Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal* 37(2):233–243.

[109] LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation applied to handwritten zip code recognition. *Neural Computation* 1(4):541–551.

[110] Lewis DD, Ringuette M (1994) A comparison of two learning algorithms for text categorization. *Third Annual Symposium on Document Analysis and Information Retrieval*, volume 33, 81–93.

[111] Li KJ, Fong DK, Xu SH (2011) Managing trade-in programs based on product characteristics and customer heterogeneity in business-to-business markets. *Manufacturing & Service Operations Management* 13(1):108–123.

[112] Liu S, He L, Shen ZJM (2018) On-time last mile delivery: Order assignment with travel time predictors. *Forthcoming in Management Science* .

[113] Loughran T, McDonald B (2011) When is a liability not a liability? Textual analysis, dictionaries, and 10-ks. *The Journal of Finance* 66(1):35–65.

[114] Lu Z, Pu H, Wang F, Hu Z, Wang L (2017) The expressive power of neural networks: A view from the width. *Advances in Neural Information Processing Systems*, 6231–6239.

[115] MacQueen J, et al. (1967) Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, 281–297 (Oakland, CA, USA).

[116] Mandi J, Demirović E, Stuckey P, Guns T, et al. (2019) Smart predict-and-optimize for hard combinatorial optimization problems. Working paper.

[117] Marr B (2016) A short history of machine learning–every manager should read `http://tinyurl.com/gslvr6k`.

[118] McCarthy J, Feigenbaum EA (1990) In memoriam: Arthur samuel: Pioneer in machine learning. *AI Magazine* 11(3):10–10.

[119] Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

[120] Mišić VV (2020) Optimization of tree ensembles. *Operations Research* 68(5):1605–1624.

[121] Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning. *International Conference on Machine Learning*, 1928–1937.

[122] Montoya R, Gonzalez C (2019) A hidden markov model to detect on-shelf out-of-stocks using point-of-sale data. *Manufacturing & Service Operations Management* 21(4):932–948.

[123] Moon TK (1996) The expectation-maximization algorithm. *IEEE Signal Processing Magazine* 13(6):47–60.

[124] Morales DR, Wang J (2010) Forecasting cancellation rates for services booking revenue management using data mining. *European Journal of Operational Research* 202(2):554–562.

[125] Nelder JA, Wedderburn RW (1972) Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)* 135(3):370–384.

[126] Pan SJ, Yang Q (2009) A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22(10):1345–1359.

[127] Pearson K (1901) Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical Magazine and Journal of Science* 2(11):559–572.

[128] PwC (2020) Internet advertising revenue report: Full year 2019 results q1 2020 revenues `https://www.iab.com/wp-content/uploads/2020/05/FY19-IAB-Internet-Ad-Revenue-Report\_Final.pdf`.

[129] Queenan C, Cameron K, Snell A, Smalley J, Joglekar N (2019) Patient heal thyself: Reducing hospital readmissions with technology-enabled continuity of care and patient activation. *Production and Operations Management* 28(11):2841–2853.

[130] Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. *Nature* 323(6088):533–536.

[131] Rusmevichientong P, Shen ZJM, Shmoys DB (2010) Dynamic assortment optimization with a multinomial logit choice model and capacity constraint. *Operations Research* 58(6):1666–1680.

[132] Russo DJ, Van Roy B, Kazerouni A, Osband I, Wen Z (2018) A tutorial on thompson sampling 11(1):1–96.

[133] Sahami M, Dumais S, Heckerman D, Horvitz E (1998) A bayesian approach to filtering junk e-mail. *Learning for Text Categorization: Papers from the 1998 Workshop*, 98–105.

[134] Schapire RE (1990) The strength of weak learnability. *Machine Learning* 5(2):197–227.

[135] Scherer D, Müller A, Behnke S (2010) Evaluation of pooling operations in convolutional architectures for object recognition. *International Conference on Artificial Neural Networks*, 92–101 (Springer).

[136] Schulman J, Levine S, Abbeel P, Jordan M, Moritz P (2015) Trust region policy optimization. *International Conference on Machine Learning*, 1889–1897.

[137] Shen H, Huang JZ (2008) Interday forecasting and intraday updating of call center arrivals. *Manufacturing & Service Operations Management* 10(3):391–410.

[138] Shen ZJM, Tang CS, Wu D, Yuan R, Zhou W (2019) Jd. com: Transaction level data for the 2020 msom data driven research challenge. Working paper.

[139] Şimşek AS, Topaloglu H (2018) An expectation-maximization algorithm to estimate the parameters of the markov chain choice model. *Operations Research* 66(3):748–760.

[140] Slivkins A (2019) *Introduction to multi-armed bandits*. ArXiv preprint arXiv:1904.07272.

[141] Strehl AL, Li L, Wiewiora E, Langford J, Littman ML (2006) Pac model-free reinforcement learning. *Proceedings of the 23rd International Conference on Machine Learning*, 881–888.

[142] Sun L, Lyu G, Yu Y, Teo CP (2020) Cross-border e-commerce data set: Choosing the right fulfillment option. *Manufacturing & Service Operations Management* .

[143] Sutton RS (1988) Learning to predict by the methods of temporal differences. *Machine Learning* 3(1):9–44.

[144] Sutton RS, McAllester DA, Singh SP, Mansour Y (2000) Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, 1057–1063.

[145] Swaminathan A, Joachims T (2015) Counterfactual risk minimization: Learning from logged bandit feedback. *International Conference on Machine Learning*, 814–823.

[146] Taigman Y, Yang M, Ranzato M, Wolf L (2014) Deepface: Closing the gap to human-level performance in face verification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1701–1708.

[147] Talluri K, Van Ryzin G (2004) Revenue management under a general discrete choice model of consumer behavior. *Management Science* 50(1):15–33.

[148] Thompson WR (1933) On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25(3/4):285–294.

[149] Tibshirani R (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 58(1):267–288.

[150] Ugander J, Yin H (2020) Randomized graph cluster randomization. Working paper.

[151] van Ryzin G, Vulcano G (2015) A market discovery algorithm to estimate a general class of nonparametric choice models. *Management Science* 61(2):281–300.

[152] van Ryzin G, Vulcano G (2017) An expectation-maximization method to estimate a rank-based choice model of demand. *Operations Research* 65(2):396–407.

[153] Varian HR (2016) Causal inference in economics and marketing. *Proceedings of the National Academy of Sciences* 113(27):7310–7315.

[154] Vulcano G, Van Ryzin G, Ratliff R (2012) Estimating primary demand for substitutable products from sales transaction data. *Operations Research* 60(2):313–334.

[155] Wager S, Athey S (2018) Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association* 113(523):1228–1242.

[156] Wang G, Li J, Hopp WJ (2017) Personalized health care outcome analysis of cardiovascular surgical procedures. Working paper.

[157] Wang G, Li J, Hopp WJ (2018) An instrumental variable tree approach for detecting heterogeneous treatment effects in observational studies. Working paper.

[158] Ward Jr JH (1963) Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association* 58(301):236–244.

[159] Watkins DP C J (1992) Q-learning. *Machine Learning* 8(3–4):279–292.

[160] Watkins CJCH (1989) *Learning from Delayed Rewards*. Ph.D. thesis, King's College, Cambridge, UK.

[161] Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8(3-4):229–256.

[162] Wu Y, Shariff R, Lattimore T, Szepesvári C (2016) Conservative bandits. *International Conference on Machine Learning*, 1254–1262.

[163] Wylie C (2018) The history of neural networks and ai: Part ii `https://opendatascience.com/the-history-of-neural-networks-and-ai-part-ii`.

[164] Ye Z, Zhang D, Zhang H, Zhang RP, Chen X, Xu Z (2020) Cold start on online advertising platforms: Data-driven algorithms and field experiments. Working paper.

[165] Zhang B, Hsu M, Dayal U (1999) K-harmonic means-a data clustering algorithm. Hewlett-packard labs technical report hpl-1999-124.

[166] Zhang D, Hu M, Liu X, Wu Y, Li Y (2020) Netease cloud music data. *Forthcoming in Manufacturing & Service Operations Management* .

[167] Zhang S, Lee D, Singh PV, Srinivasan K (2016) How much is an image worth? An empirical analysis of property's image aesthetic quality on demand at airbnb. Working paper.