Data-Driven Decision-Making (D^{3}M): Framework, Methodology, and Directions

Jie Lu, *Fellow, IEEE*, Zheng Yan, Jialin Han, and Guangquan Zhang

***Abstract*—A decision problem, according to traditional principles, is approached by finding an optimal solution to an analytical programming decision model, which is known as model-driven decision-making. The fidelity of the model determines the quality and reliability of the decision-making; however, the intrinsic complexity of many real-world decision problems leads to significant model mismatch or infeasibility in deriving a model using the first principle. To overcome the challenges that are present in the big data era, both researchers and practitioners emphasize the importance of making decisions that are backed up by data related to decision tasks, a process called data-driven decision-making (D^{3}M). By building on data science, not only can decision models be predicted in the presence of uncertainty or unknown dynamics, but also inherent rules or knowledge can be extracted from data and directly utilized to generate decision solutions. This position paper systematically discusses the basic concepts and prevailing techniques in data-driven decision-making and clusters-related developments in technique into two main categories: programmable data-driven decision-making (P-D^{3}M) and nonprogrammable data-driven decision-making (NP-D^{3}M). This paper establishes a D^{3}M technical framework, main methodologies, and approaches for both categories of D^{3}M, as well as identifies potential methods and procedures for using data to support decision-making. It also provides examples of how D^{3}M is implemented in practice and identifies five further research directions in the D^{3}M area. We believe that this paper will directly support researchers and professionals in their understanding of the fundamentals of D^{3}M and of the developments in technical methods.**

***Index Terms*—Data-driven decision-making, decision support systems, computational intelligence.**

I. Data-Driven Decision-Making (D^{3}M): Framework

DECISION-MAKING requires the optimal or most satisfactory solution to a decision problem to be found in a process of identifying and evaluating alternatives, and subsequently selecting the "best" option [1], [2]. "Best" here means the optimal or most satisfactory solution for the particular decision problem. Decision problems are of various types, ranging from daily operational decisions to long-term strategy business decisions, and from internal single decisions to multi-level decisions or multi-organizational decisions [3]. Decision-making tasks differ in their characteristics and therefore are approached by a diverse range of techniques.

Huge amounts of static and streaming data are now generated in daily life by governments, industries and other sources, such as sensors and marketing activities [4]. With this explosion in data and the rapid development of information technology, decision makers are acquiring improved capabilities to collect, store, access, and analyze data [5]. Now, more than ever, data is better utilized, and it lies at the heart of decision-making [6], whether for huge multinationals, home-owned operations, or individuals. As a consequence, data-driven decision-making (D^{3}M) has become an emerging topic that is receiving considerable attention from both researchers and industry. D^{3}M aims to infer a decision based on deep analysis and learning historical and current data related to the decision problem, which is fundamentally different from traditional model-driven decision-making. Taking portfolio management as an example, the objective is to manage the portfolio of securities that maximizes the rate of return. To approach this problem via model-driven decision-making, decision variables such as the investment ratio of each security are first identified. Then a decision model such as a fractional programming is formulated [7]. The solution to this programming problem provides optimal decisions for the portfolio management. On the other hand, if we approach this problem via D^{3}M, data mining and machine learning techniques will be explored to infer future investment strategy directly from historical data.

A wide variety of D^{3}M methods have been investigated in the last decade and we cluster these approaches into two categories. Each category addresses two types of data: static data and streaming data.

The first category is Programmable Data-Driven Decision-Making (P-D^{3}M), which approaches decision-making in two main steps. Step 1) involves the derivation of a decision model based on data mining or statistical learning and a programmable model (also called a programming or optimization model) which is correspondingly formulated to provide decision support, e.g., a multi-objective decision model and a multi-level decision model [2], [3], [8]. The incorporation of data analytics in decision-model derivation is both necessary and useful because many model parameters encounter uncertain or unknown information due to the existence of hidden relationship patterns and stochastic factors [9], [10]. The objective functions, parameters and constraints of the decision model cannot be established by using first principles. Step 2) involves the development of solution algorithms to find an optimal solution to the resulting

Manuscript received October 15, 2018; revised February 27, 2019; accepted April 28, 2019. Date of publication June 7, 2019; date of current version July 22, 2019. This work was supported by the Australian Research Council under discovery Grants DP150101645 and DP170101632. *(Corresponding author: Jie Lu.)*

The authors are with the Decision Systems and e-Service Intelligence Lab, Centre for Artificial Intelligence, University of Technology Sydney, Ultimo, NSW 2007, Australia (e-mail: jie.lu@uts.edu.au; zheng.yan@uts.edu.au; jialinhan@jxufe.edu.cn; guangquan.zhang@uts.edu.au).

Digital Object Identifier 10.1109/TETCI.2019.2915813

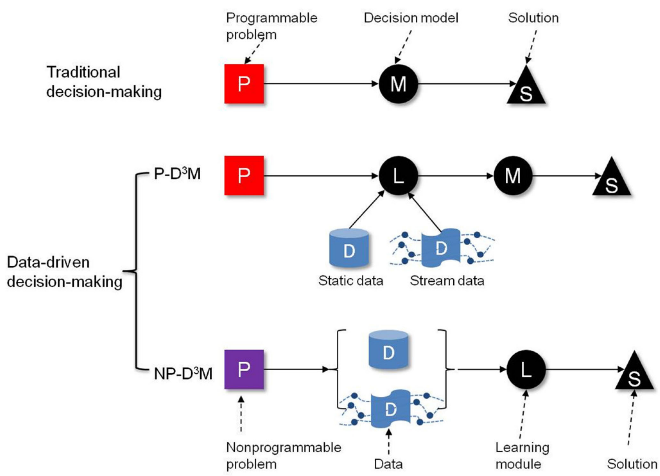

Fig. 1. The technical framework of data-driven decision-making (D^3M).

decision model. In my scenarios, computing the optimal solution given a well-formulated decision model is a demanding task. Computational intelligence methods have demonstrate promises for deriving the optimal decisions.

In contrast to P-D^3M, the second category of D^3M, Nonprogrammable Data-Driven Decision-Making (NP-D^3M), applies naturally to situations in which the derivation of a decision model is computationally infeasible or prohibitively costly. The dynamics and uncertainties in a large and highly complicated decision problem may lead to severe model mismatch or make the programmable model intractable [11]. To deal with this problem, NP-D^3M explores a learning mechanism that discovers rules and patterns from data to directly make a decision, which enables decision makers to strategize on the basis of data-based evidence.

To help researchers understand the basic concepts of D^3M and related technical developments, and to assist practitioners to maximize the use of various data for decision-making, Fig. 1 illustrates the technical framework of the two categories of D^3M. For Category 1 problems, historical data-based analytics can accurately predict the unknown parameters for a programmable decision model, and these parameters can feed, update and drive the model. For Category 2, that is, when a decision problem cannot be characterized by a programmable model, a machine learning–based data analysis can discover useful rules or knowledge/options from data for decision support. For either category of D^3M technique, this study carefully identifies potential methods and procedures for using data to support decision-making and provides examples to show how data-driven decision-making is implemented in practice. It is hoped that this will directly motivate and support researchers and practitioners to promote the application of data-driven decision-making techniques in a wide variety of domains.

The remainder of this paper is structured as follows. In Section II, P-D^3M methodologies and techniques are described and discussed. Section III addresses NP-D^3M methodology with detailed methods and algorithms. Section IV presents our comprehensive analysis on D^3M research challenges and directions.

TABLE I
THE STEPS FOR IMPLEMENTING P-D^3M

Steps	Descriptive methods	Predictive methods
1	Problem definition and formulation, parameter identification	
2	Data collection, preprocessing and partition	
3	Measures selection for descriptive analysis	Learning methods and architecture selection
4	Parameter estimation by summarizing what happened	Parameter estimation by predicting what will happen
5	Testing using goodness of fit tests and residual examination	Testing generalization performance using new data
6	Using the prediction result to update the decision model	

II. PROGRAMMABLE DATA-DRIVEN DECISION-MAKING (P-D^3M) METHODOLOGY

A. *Outline of the Main Methods*

In a P-D^3M problem characterized by a programmable model, the model parameters and decision variables in decision objectives and constraints often encounter uncertain and unknown information, which makes the overall problem more difficult to handle than its deterministic counterparts [12]. To overcome this issue, historical data can be harnessed to identify or generate the missing pieces. For example, a deterministic programmable decision-making problem considers an optimization problem in the form: $\min_{x\in D} f(x)$. Since $f(x)$ maybe partially decided by additional data ξ, the optimization problem associated with P-D^3M is actually $\min_{x\in D} f(x, \mathbb{E}[\xi])$. In order to perform mathematical programming for this problem, machine learning techniques have to be developed to make an inference on $\mathbb{E}[\xi]$. In some cases, the evaluation of $f(x)$ is more essential than its expression, where a surrogate model $\tilde{f}$ is derived to evaluate the fitness values of the candidate solutions and to provide hints about the location of promising candidate solutions [13], [14]. Surrogate-assisted evolutionary algorithms are correspondingly developed to search for the optimal solution.

There are two main schools of thought in the contemporary P-D^3M universe on the development of data analytics: the first uses descriptive analytics based on mathematical statistics as the tool of choice, while the other uses predictive analytics with the aid of machine learning methods. Descriptive analytics captures the essence of what is happening through data querying, filtering, and quantitative analysis, whereas the aim of predictive analytics is to infer what is likely to happen through statistical learning methods such as maximum likelihood estimation [15], [16], logistic regression [17], kernel methods [18], and artificial neural networks [19].

B. *Procedures of P-D^3M*

The steps for implementing a typical P-D^3M method are summarized in Table I.

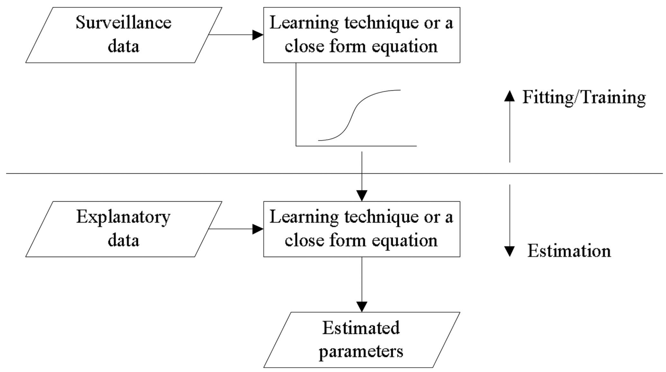

Fig. 2. Methodological framework of Steps 3–5 for data-driven prediction of the model parameters.

Although descriptive analytics and predictive analytics provide insights from different perspectives, their implementation process is the same, regardless of the approach used. First, we need to recognize and define the decision problem and use a decision model (e.g., a multilevel decision model) to characterize the decision-making process. Considering that unknown parameters may exist in the resulting decision model, steps are taken to collect and preprocess historical data, and to select a method to predict the model parameters using the data. Parameter prediction in Steps 3-5, in particular, can be divided into two phases in both methods: fitting (or training) and estimation, as shown in Fig. 2.

In the fitting step, portions of the available data (called surveillance data) are identified and the prediction model is formed, while the estimation step determines the unknown or uncertain parameters from the new data (known as explanatory data) using the proposed prediction model [20]. The prediction result can then be used to update the decision model, and an optimal solution to the decision problem can be found.

C. An Illustrative Instance

We use an example of production planning and scheduling in the manufacturing industry to illustrate the methods of establishing the parameters for a given decision model. Production planning and scheduling are two closely linked activities that belong to different decision-making levels in a job shop manufacturing system [21]. To improve the overall performance in terms of production efficiency and total cost, production planning and scheduling are always integrated and handled simultaneously, using, for example, bi-level decision modeling [22]. To provide a solution, the bi-level decision model clearly needs to know the related marketing and production parameters of the objective functions and constraints of the given model, e.g., the market demand for products during the forward planning period and the production machine rate. However, these parameters are commonly unknown or uncertain prior to order generation and production implementation [23]. For example, key production parameters, such as the processing and changeover times of tasks, are always subject to variation as a result of mechanized or human-centered factors, such as tasks that take more or less time than anticipated. To handle this, historical marketing and production data can be maximized and statistics learning methods can be applied to predict related unknown or uncertain parameters. For example, neural networks can be used to train the model parameters from a large percentage of the historical data, while the remaining historical data and the new data that is generated can respectively be used to evaluate prediction performance and generate the parameters for the given decision model. Lastly, the prediction result can be integrated into the bi-level decision model to generate an optimal solution for integrated production planning and scheduling.

D. *Decision Optimization* via *Computational Intelligence*

To obtain the optimal solution to a given decision model is not a trivial task. For example, as the decision model may continuously adapt to changes in the data pattern, the algorithms are expected to be capable of handling time-varying parameters and solving problems in real time. The main challenges faced in D^3M from the computational point of view include nonconvexity, discontinuity, non-smoothness, and the curse of dimensionality. The difficulties are significantly amplified when the optimal decision is required in real time. To mitigate these problems, computational intelligence approaches were developed and implemented [24].

Neural networks have been used to compute optimal solutions. This can be achieved by two families of techniques characterized by whether the neural networks developed are discrete-time or continuous-time. In the first class, learning to optimize takes place when a search formula is parameterized by a discrete-time neural network and a corresponding learning method is designed to find the search formula. This methodology was independently proposed in [25], [26] and has since received growing attention. The essence of the learning to optimize approach is that it allows an algorithm to learn to exploit structure in problems of interest in an automatic way which greatly improves search efficiency and optimality. For example, the traveling salesman problem is solved in [27] by using a neural network and reinforcement learning, a method which has outperformed many iterative methods on benchmark problems and could easily be scaled up to 200 nodes. The main advantage of the learning to optimize methods is that they converge much faster in the run-time, which comes at a price of high computational cost during training. The second class is neurodynamic optimization, in which a continuous recurrent neural network is designed as a goal-seeking model to compute the optimal solution [28]. The past decade has witnessed the success and maturity of neurodynamic optimization. Various neural network models with simple model complexity and good convergence properties have been developed for linear programming, quadratic programming, variational inequalities, pseudoconvex programming, nonsmooth optimization, global optimization, and distributed optimization [28]–[35]. Neurodynamic models exploit the distributed information processing of neural computation and can be implemented in designated hardware, leading to running times at orders of magnitude faster than popular optimization algorithms executed on digital computers.

In parallel to the research of neural network for optimization, evolutionary computing has demonstrate enormous success for global optimization of a decision-making problem, especially for multi-objective optimization [36]. This family of algorithms apply population-based trial and error techniques to search for a solution in a metaheuristic and stochastic way. There are two main classes of algorithms: the first class such as genetic algorithm, differential evolution, and evolutionary programming is inspired by natural evolution where recombination and mutation are the basis of the computing paradigm. The second class such as particle swarm optimization and ant colony is inspired by swarm intelligence that emphasizes on the communication of a population. The main advantage of evolutionary computing lies in its capabilities to carry out global search as well as dealing with multiple objectives. One critical limitation of evolutionary computing is that it generally requires large number of computing agents and iterations, which makes it relatively slow in the run-time.

III. Nonprogrammable Data-Driven Decision-Making (NP-D^{3}M) Methodology

Programmable models cannot be derived for all decision problems. Fortunately, in most situations, a variety of sources generate related data for nonprogrammable decision problems. A learning module can be developed to extract useful rules or knowledge from data, e.g., association rules and decision trees, to directly support decision-making; this is known as NP-D^{3}M. The difficulty of NP-D^{3}M mainly lies in how to build a learning module, as shown in Fig. 1, to support decision-making according to the properties of the data. In this section, we summarize three main categories of techniques capable of creating a bridge between data and decision-making: rule-based methods, preference-based decision-making, and reinforcement learning.

A. Rule-Based NP-D^{3}M Method

1) Concept: Rules are a good way of representing knowledge extracted from a data set [37], as the decision-making process can be effectively motivated via a set of IF-THEN rules. Rules can be clustered into three main categories according to their features and usage. 1) Decision rules are relationships induced in an environment where the meaning of the attributes in the IF part and in the THEN part is distinguished a priori. Decision rules are often extracted from a decision tree. 2) Association rules are relationships between any two disjoint subsets of attributes. A classical motivating example of association rules is the narrative between *beer* and *diaper*, where the association rule (i.e., *beer* = >*diaper*) discovered from a large number of market transactions is used to re-arrange the position of the products to increase sales. It should be noted that association rules induced from data are referred to as decision rules if the set of attributes in the consequent of any rule has been fixed [37]. 3) Fuzzy rules are conditional statements in which the IF part and the THEN part are linguistic values respectively determined by fuzzy sets on a given domain. For example, "IF (age = youth and income = medium), then (a computer will be bought)" is a fuzzy rule, in which "youth" and "medium" are fuzzy sets with a membership function and can be defined on the given domain (e.g., the age is between 22–30 and the monthly income is between 3000-5000AUD). As a synthetic, easily understandable, and generalized representation of knowledge, rules-based decision-making has been widely applied in many application domains in relation to classification, sorting, choice and ranking [38], e.g., airline service evaluation [39], heavy rail track maintenance [40], and energy consumption optimization [41]. The main limitation of rule-based algorithms is that they require strong domain experts and knowledge to formalize the rules.

2) Outline of the Main Algorithms: A decision rule is identified from a decision tree that constructs internal nodes from a root node to terminal nodes, in which each internal node specifies a condition to be tested, and each link represents one of the states of this condition [42]. Decision trees are always built recursively following a descending strategy. There are three main algorithms for constructing decision trees: the CART algorithm [43], the ID3 algorithm [44], and the C4.5 algorithm [45]. The main difference between the three algorithms lies in the splitting criteria [46], [47], i.e., how many splits are allowed at each level of the tree, how those splits are chosen when the tree is built, and how tree growth is limited to prevent over-fitting.

The key to association rule mining is to find a way to efficiently discover the rules from all possible candidates with exponential scale. There are a number of models or algorithms proposed in the literature, of which three are the most classical: Apriori [48], FP-growth [49], and Eclat [50].

In fuzzy rule mining, there are two main categories of inference systems for constructing fuzzy rules: Mamdani-Type and Sugeno-Type [51]. Mamdani-Type fuzzy rules consist of condition parts and conclusion parts. Both parts are represented by the linguistic terms described by the fuzzy sets with membership functions [52]. Sugeno-Type fuzzy rules consist of condition parts and conclusion parts where the former are represented as the fuzzy sets, but the latter are governed by the linear functions of the input attributes [53].

3) Procedures of Rules Induction: The overall strategy for a decision tree consists of three steps [54].

Step 1: Data preparation. The recorded data is prepared and preprocessed to reduce the complexity of the decision tree. For example, the documents collected from an online community are unstructured data mainly expressed in text format, so they need to be transformed into structured data for further analysis [42].

Step 2: Decision tree induction and evaluation. The training data set (e.g., $^2/_3$ of the data) is used to build the decision tree, while the remaining data, known as the testing data, is used to check the decision tree. Given an instance from the testing data set, a decision can be generated by following the path to the tree from the root node to a leaf node, using the sample values and the tree structure. Expert knowledge can also be used to evaluate the validity and physical soundness of the decision tree.

Step 3: The resulting decision tree is applied to decision-making on new instances. It can be used to handle new examples (cases whose classes are not known a priori), to detect patterns, or simply to gain a better understanding of the phenomenon being analyzed.

The procedure of association rule mining is usually composed of two steps because there are two evaluation metrics or selection criteria (known as support and confidence) for association rules.

Step 1: Discover frequent item sets from the original transaction database using support as the only criterion. Different algorithms will build their own data structure for this step, such as FP-tree in FP-growth and item-transaction mapping in Eclat. This step can extensively reduce the scale of possible association rules. Note that frequent item patterns are not association rules because there is no direction between the items in each pattern.

Step 2: Discover the association rules from the frequent item patterns using confidence as the criterion.

The design procedure for constructing fuzzy rules is summarized in three steps [55].

Step 1: Forming the conditions. A number of fuzzy sets with appropriate membership functions are constructed for each attribute. The condition part of the *k*th fuzzy rule can be formally represented as: "IF (X_1 is A_{1k} and X_2 is A_{2k} and $\ldots$ and X_n is A_{nk})", where $X = [X_1 \ldots X_n]$ is the input data, and A_{ik} is the fuzzy set corresponding to the input attribute X_i, $i = 1, 2, \ldots, n$.

Step 2: Constructing the conclusions. The methods for constructing the conclusion aspect of Mamdani-Type and Sugeno-Type fuzzy rules are different. In Mamdani-Type, the output attribute corresponds to the fuzzy sets, and the conclusion part of the *k*th fuzzy rule is described as: "Then *Y* is B_k", where *Y* is the output attribute, and B_k is a linguistic term (fuzzy set). In Sugeno-Type, the output is calculated by the linear function of the input attributes: "Then *Y* is $a_1X_1 + \cdots + a_2X_2$".

Step 3: Building fuzzy rules. Using the conditions and conclusions, the fuzzy rules are obtained as follows:

Mamdani-Type: IF (X_1 is A_{1k} and X_2 is A_{2k} and $\ldots$ and X_n is A_{nk}), THEN (*Y* is B_k).

Sugeno-Type: IF (X_1 is A_{1k} and X_2 is A_{2k} and $\ldots$ and X_n is A_{nk}), THEN (*Y* is $a_1X_1 + \cdots + a_2X_2$).

4) Illustrative Instances: We use an example to illustrate how to use rule-based approaches to tackle real-world decision-making problems. Consider direct marketing where companies try to establish and maintain a direct relationship with their customers in order to select likely buyers of certain products and to promote the products accordingly. This is a D^{3}M problem because large databases of customer and market data are built and maintained for deriving optimal decisions.

A decision tree can contribute to the classification of customers based on customer data and identified features such as gender, age, occupation and so on. Decision trees can be constructed to build a path from the characteristics of the customers to the categories of goods by recursively applying numerical splitting criterion to the training data. The testing data set and expert knowledge are used to evaluate the performance of the resulting decision tree. Lastly, the decision tree is applied to identify relationships between customers and items.

The association rules can also contribute to direct marketing. Consider a market basket analysis problem in which each record in a database is a successful customer transaction incorporating a number of items, e.g., <cola, beer, diaper, and bread>. Given this transaction database, managers may seek to learn which items are often purchased together and thus how to distribute promotion coupons to customers. With the help of association rule mining, a number of association rules can be obtained, e.g., beer $=>$ diaper, bread $=>$ milk, and cola $=>$ chips. These rules provide managers with sufficient information about the buying habits of their customers to assist their decision-making. Using this information, managers can reposition items for customer convenience, e.g., put bread and milk in close proximity, or disseminate promotional coupons to increase sales, e.g., distribute coupons for chips to customers who have previously bought cola.

To apply fuzzy rules, we first develop fuzzy set with membership functions to model linguistic terms such as client's age ("young", "middle aged", "old"), client's income ("low", "average", "high"), and client's rating ("loyal customer", "impulsive customer", "discount customer"). After applying fuzzy modelling, customer selection using fuzzy rules can be given as follows:

1) IF (the age is young and the income is high), THEN (impulsive customer).
2) IF (the income is low), THEN (discount customer).
3) IF (the age is middle and the income is average), THEN (loyal customer).

B. Preference-Based NP-D^{3}M

1) Concept: Preference-based decision-making refers to a set of problems in which the best decision is selected according to the decision-maker's preference rather than a quantitative measure [56]. For the same problem setting with the same dataset, different decision-makers may have a different rationale for their choices. A typical example for preference-based decision-making is a recommender system which attempts to recommend the most suitable items (products or services) to particular users (businesses or individuals) by predicting a user's interest in an item based on historical data [57]. The most important feature of recommender systems is their ability to "guess" a user's preferences and interests by analyzing the behavior of this user and/or the behavior of other users to generate personalized recommendations.

2) Outline of the Main Algorithms: In general, the generation of recommendation lists is based on user preferences, item features, past user-item interactions and additional information. Recommender system approaches are mainly categorized as collaborative filtering, content-based methods, or hybrid methods [58]. Collaborative filtering makes recommendations by learning from historical interactions, which is either explicit feedback (e.g., previous user ratings) or implicit feedback (e.g., browsing histories). Content-based recommendation is mainly based on comparisons across item and user features, and a range of features can be extracted from auxiliary information such as texts, images and videos. A hybrid model is a recommender system that integrates two or more types of recommendation strategies. Recently, many novel machine learning techniques have been applied to capture nonlinear and nontrivial user-item relationships to improve the quality of the recommendation list; for example, the fuzzy recommender systems developed in [59], [60] use fuzzy techniques to model the vagueness of user preferences. In [61–[63], various deep neural networks are developed for capturing the intricate relationships between users, items, and their interactions.

3) Procedures of Recommendation Systems: The essence of a recommender system is to capture user preferences and their association with items. The design procedures of a recommender system consist of four major steps.

Step 1: Collect and organize data on users and items, starting with the identification of users and the items they are using. A table is commonly used to link items and users as key/value pairs, where each item is a key and the numeric preference of each user on this item constitutes the value. The notation $C[i, j]$ represents the numerical rating of user *j* over item *i*. If a user has not rated an item, then $C[i, j]$ is null.

Step 2: Find the items that most satisfy the user's preferences. This step begins with data preprocessing, which aims to produce a reweighted rating that is more reflective of the quality of the data. Taking a movie rating dataset as an example, the popularity of a movie should be considered in addition to its raw rating to ensure that a movie with a rating of 9 from 10 voters should not be preferred over a movie with a rating of 8.9 from 10,000 voters. Based on the weighted rating, machine learning models such as autoencoders can be applied to learn a feature level representation of the user-item data. Next, a model is designed to compare a specific User A with all other users. This model creates a set of users that are most similar to User A. The model uses distance measures to compute the similarity of users. Common choices include Pearson correlation, cosine similarity, and Spearman correlation. Once similar users have been found, each user's vector can be examined to predict User A's rating on items that were null to A.

Step 3: Rank and recommend. The recommendation system can be extended by ranking the items that are recommended to User A. The greater the number of similar users who rate an item, the higher the rank assigned to that item. The system will then recommend the items that rank highest on the list to User A.

Step 4: Evaluate and test. Design measures to quantify the performance of the recommender system, and select a small number of users to act as "test users" to evaluate whether the system is working accurately.

4) An Illustrative Instance: We use the example of how YouTube provides personalized video recommendations to its users to illustrate the key elements of preference-based decision-making [64]. The aim is to provide a list of videos that will best fit the interests of a particular user according to their preferences. The data sources available for this decision-making problem include content data such as video titles, descriptions, and labels, and user activity data such as ratings, favorites, and view time. To generate recommendation candidates, the recommender determines a set of videos that a user is likely to watch. This is achieved by mining related videos based on a seed set containing videos watched and favorited by the user, and then comparing their similarity. YouTube has developed a similarity measure using polar distance. Once the recommendation candidates have been obtained, they are ranked according to a set of criteria such as video quality, user specificity, and diversification. The YouTube recommender system provides 60% of clicks on the homepage, which is a significant contribution to user engagement.

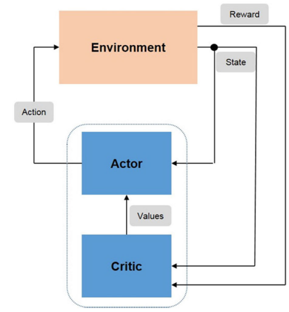

Fig. 3. Actor-critic structure for reinforcement learning-based D³M.

C. Reinforcement Learning-Based NP-D³M Methods

1) Concept: As many D³M problems have to be solved in a sequential manner, reinforcement learning (RL) provides a learning framework to tackle such problems based on interactions between an agent (e.g., decision maker) and environments [65]. The basis of RL is reward-driven behavior, in which an agent makes a decision by maximizing its future reward. The agent interacts with its environment and, by observing the consequences of the previously made decision, learns to modify its own decision in response to rewards received. RL originated in biology and psychology, and is capable of handling decision-making problems where there is no decision model, no utility function, or even no related examples. RL has demonstrated some success in the past, although this success is limited to low-dimensional problems. The proliferation of deep learning in recent years has enabled RL to handle complex high-dimensional decision-making problems. Its most thrilling success was the development of the AlphaGo player, which defeated a human world champion at the game of Go [66], [67]. RL has also shown promise for solving complex decision-making problems such as medical diagnoses [68].

2) Outline of the Main Algorithms: One class of RL methods that has achieved state-of-the-art performance in many decision-making tasks is that based on an Actor-Critic structure [69], as shown in Fig. 3. The key elements of this RL method are summarized as follows:

Problem setup: The actor-critic structure has an actor that applies an action a_t to the environment that is in state s_t at the time step *t*, and a critic that assesses the value V_t of that action. Based on this assessment and the update environment state s_{t+1}, the actor modifies its action a_{t+1} so that a new action can yield a value V_{t+1} that is better than the previous one. The optimal sequence of actions (i.e., decisions) $\{a_t, \ldots, a_T\} = \pi(s)$ is determined by the rewards R provided by the environment. Every time the state transits from s_t to s_{t+1}, an instantaneous reward r_{t+1} is provided. However, what RL aims to maximize is a long-term cumulative reward $R = \sum_{i=t}^{T} \gamma^{i-t} r_i$ where $\gamma \in (0, 1]$ is a discount factor.

Value function: A value function is a prediction of the expected accumulative future award measuring the goodness of each state-action pair. The value of a state following a policy π from state s is defined by $V_\pi(s) = \mathbb{E}[R_t|s_t = s]$. The main objective of RL is to determine a policy $\pi(a, s)$ to maximize the expected return

$$\pi^* = \arg\max_\pi V_\pi(s) = \arg\max_\pi \mathbb{E}\left[\sum_{i=t}^{t+T} \gamma^{i-t} r_i | a_t = a\right].$$

According to the Bellman equation,

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\left[r + \gamma V_\pi(s')\right].$$

This equation sets the foundation for many RL methods as it establishes a relationship between the current value in state s and the value in the next state s' following a policy $\pi(a, s)$.

Learning mechanism: The actor-critic structure produces a forward-in-time algorithm to iteratively find a policy [70]. The learning mechanism has two steps: policy evaluation, executed by the critic, followed by policy improvement, performed by the actor. The policy evaluation computes $V_\pi(s)$ to quantify the optimality of the policy π. The policy improvement finds a new policy π' that results in $V_{\pi'}(s) \geq V_\pi(s)$.

3) Procedures of RL-Based Decision-Making: The essence of RL is to find a function π to solve a decision-making problem. Many strategies have been developed to compute π. Here, we present the implementation procedures of a prevailing neural network-based approach, which consists of three main steps [69].

Step 1: Define an actor. A neural network is used to approximate the actor whose input data is the observation of the agent, and whose output is every action corresponding to a neuron in the output layer. The dimension of the output layer of the neural networks is equal to the dimension of the action space.

Step 2: Train the actor. Given an actor $\pi_\theta(s)$ where θ denotes the neural network parameters, the actor is trained by maximizing the expected reward R_θ. The agent and environment are first sampled to obtain a trajectory $\tau = \{s_1, a_1, r_1, \ldots, s_T, a_T, r_T\}$, and the corresponding reward becomes $R(\tau) = \sum_{i=1}^{T} r_i$. The sampling probability of each τ is denoted by $P(\tau|\theta)$. Once all training data have been sampled, the expected reward becomes $R_\theta = \sum_\tau R(\tau)P(\tau|\theta)$. Next, gradient ascent or more advanced optimization algorithms are applied to find the optimal $\theta \leftarrow \theta + \eta\nabla R_\theta$ such that R_θ is maximized. Noting that $\nabla R_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta)$, a critic is introduced to evaluate $R(\tau)$ based on the value function. The evaluation procedures based on temporal-difference learning are summarized as follows:

Input: the policy π_θ to be evaluated.
Output: value function V (i.e., $\mathbb{E}[R(\tau)]$).
Initialize *V*, *s*, do
  for *s* is not terminal do
    sample an action *a* given by $\pi_\theta(s)$
    taken action *a*, observe *r*, s'
    $V(s) \leftarrow V(s) + \beta(r + \gamma V(s') - V(s))$
    $s \leftarrow s'$
end

Step 3: Generate a decision using the trained actor-critic. After validation, the trained actor-critic is applied as a tool to generate decisions to NP-D^{3}M problems.

4) An Illustrative Instance: We use the problem of 3D bin packing as an example to illustrate how to use RL to directly generate decisions. In a 3D bin packing problem, a number of cuboid-shaped packages are put into a bin orthogonally [71]. The objective is to find a way to pack all the items using the minimal surface area of the bin. The most important decision to be made in this problem is the sequence in which to place each item. To apply RL to provide a solution, we use a neural network to model the actor whose input is the dimensions (i.e., length, width, height) of the item, and whose output is the packing sequence. The reward is quantified by the surface area of the bin. To train the actor, the process is sampled for the purpose of collecting training data. When an item is put into the bin, its location and orientation in all empty spaces is recorded to determine the resulting surface area. The surface area data is used as the reward to guide the RL process. Once the data sets have been prepared, the learning procedures are followed to obtain an actor which is used to generate the packing sequence for future items.

IV. The Challenges and Future of D^{3}M

D^{3}M is an emerging research area that is rapidly evolving in both theoretical development and applications. Generally speaking, D^{3}M faces two important challenges: 1) streaming data-based real time decision-making in which decisions are required to provide timely support but data streams may experience significant and unpredictable changes in distribution (called drift). In this case, the induced pattern of past data may not be relevant to the new data, thus to continue using models based on past data will lead to poor prediction and poor decision outcomes; 2) inadequate data-based decision-making, in which the data, especially the labelled data are insufficient for training a decision model or generating decision options. In view of the research opportunities and challenges, we have identified five future directions that warrant in-depth investigation.

A. *Prescriptive D^{3}M*

Prescriptive analytics provides answers for what we should do with big data resources. Unlike predictive analytics, which predicts the unknown information of an assumed decision model, prescriptive analytics provides a transformative framework to automatically discover and formalize the decision-making model by learning what to optimize (e.g., utility function and decision variables) and where to optimize it (feasible region and constraints) from the data. The decision-making process in prescriptive analytics begins with a coarse approximation of the unknown true model, and continues by iteratively re-sampling data, exploring new knowledge and refining the decision model [72]. The decision-making process supports the co-evolution of the optimality of the decision and the accuracy of the decision model. The challenges it presents include the definition and quantification of the impact of a decision, the embedding of the impact in the decision-making process, the latency of information, the scalability of the algorithms for high-dimensional decision spaces, and the incorporation of knowledge to facilitate the decision-making.

B. Effective D^{3}M Under Uncertainty

Uncertainty is a defining characteristic of big data, the consequences of which has wide-ranging impact on the quality of subsequent decisions. A D^{3}M system may face to three main layers of uncertainty: 1) In the data layer, uncertainty is mainly reflected by missing data, data insufficiency, outdated data, incomplete data, and ambiguous data; 2) In the modelling and learning layer, uncertainty is affected by the similarity measure, granular level, model selection, and parameter computation; 3) In the decision output layer, uncertainty is presented as an accompanying measure which allows the uncertainty to be judged. For data layer uncertainty, an important task is to deal with linguistic terms such as normal linguistic values “big”, subjective uncertain “it is possible”, description of impression “more than 70” and description of variables “high quality”. In the modelling layer, since the key of a D^{3}M problem is learning-based decision modeling in which a strong assumption is made that historical data (training data) and new observations must come from the same domain and have identical statistical properties. However, in many complex decision-making problems involving high levels of uncertainty, data insufficiency, and rapidly changing environments, this assumption is invalid. Data patterns can change over time and domain, particularly in stream data, which may lead to degraded predictive performance if a static predictive model is applied.

To make confident decisions, the decision-making process must take a range of uncertainties into consideration. The techniques for dealing with uncertainty are mainly fuzzy logic [73], Bayesian modelling [74], and robust optimization [10], [75]. Fuzzy logic handles data value uncertainty by fuzzy sets, and deals with cases without crisp boundaries using fuzzy clustering or fuzzy rule-based models. Bayesian modelling, which is based on the fact that chance plays a role in predicting future events, incorporates random variables and probability distributions into the model. Robust optimization aims to achieve robust performance under a different set of assumptions or the effect of parameter variation. Each method has its own merits and limitations. A disruptive technique that systematically combines the strengths of these methodologies is needed to escalate research into D^{3}M under uncertainty to the next level. Bayesian deep learning [76] and explainable machine learning [77], in particular, show promise for the learning and modeling of uncertain data, and these approaches endeavor to provide a reasonable interpretation of the learning outcome. The challenge here is how to effectively handle multiple types of uncertainties within massive domains to support D^{3}M.

C. Unsupervised Learning-Based D^{3}M

Despite their recent success in D^{3}M, most methods are reliant on the availability of sufficient labeled data. However, labeling is a laborious process that may be extremely costly. Especially in the context of decision-making where the definition of labels is not straightforward. Unsupervised learning, which discovers useful representations of the observed data without the need for labeled training data, is a viable solution to provide more insights to data and hence improve the quality of decisions. In [78], unsupervised learning was used to identify phases and phases transitions of many-body systems via clustering analysis in a feature space, which may contribute to the modelling processing of P-D^{3}M. In [79], unsupervised learning model was developed to predict physical interaction between robots and environments, which greatly improved the decision-making capability of intelligent agents. The main challenges include how to incorporate specific decision-making goals into the problem formulation of unsupervised learning, and how to quantify the confidence of learned representations.

D. Transfer Learning-Based D^{3}M

In some decision situations such as the launching of new markets or new products, there is insufficient data available. D^{3}M methods are still not well-suited to the data-shortage situation. Data-shortage is a dilemma encountered in D^{3}M tasks and is particularly evident in new businesses where a substantial amount of data is needed to train a D^{3}M model. A natural question is how the knowledge acquired from previous data can be efficiently reused for current prediction tasks in a data-shortage environment. If we have data from multiple yet somewhat similar domains (for example, domains that *exhibit different feature spaces*), how can the labeled data be leveraged to exploit the knowledge it has already yielded to assist with a current prediction task?

Transfer learning tackles this problem by leveraging the knowledge acquired from a source domain to improve the accuracy and efficiency of learning in the target domain. From the problem-setting perspective, transfer learning can be classified into several different categories, such as multi-task learning, domain adaptation, cross-domain adaptation, and heterogeneous learning [80]. Cross domain transfer learning is more suitable for D^{3}M. A possible method is to construct a latent space between the feature spaces of the source domain (which has enough data to train a model) and the target domain (which has insufficient data but needs to support decision-making), so that the two domains have the same distribution in the latent space and can therefore transfer knowledge learned from the source domain to support decision generation in the target domain. In [81], [82], a comprehensive transfer learning framework for granular fuzzy regression domain adaptation is proposed by changing the input and/or output space of the source domain's model using space transformation. In [83], [84], transfer learning is applied to adaptively transfer user and item group-consistency information for cross-domain recommendation. Transfer learning techniques such as memory network [85], multitask training [86], and curriculum learning [87] have also been incorporated to facilitate the training of reinforcement learning-based decision-making.

Transfer learning-based D^{3}M is therefore a convincing solution to D^{3}M when data is insufficient. The challenges confronting this type of learning include achieving heterogeneous unsupervised transfer learning for D^{3}M, improving the model's ability to generalize, and retaining the knowledge across many tasks and domains.

E. Adaptive D^3M Under a Stream Data Environment With Concept Drift

Governments and companies are generating huge amounts of streaming data and urgently need efficient data analytics to support their decision-making, while concept drift (unforeseeable changes in underlying streaming data distribution over time) inevitably manifests in almost all these data. Providing reliable decision-making facilities in an ever-changing big data environment has become a crucial issue. Current D^3M research has been largely directed towards learning with stationary data, drawn from fixed yet unknown distributions which are not adaptive in terms of their ability to react, or even detect, concept drift. When concepts drift, the induced pattern of past data may not be relevant to the new data, which will lead to a poor fit of static D^3M, as well as impaired decision outcomes. The phenomenon of concept drift has been recognized as the root cause of decreased effectiveness in many decision-related applications, such as customer churn prediction and fraud detection; therefore, handling concept drift in D^3M has become an extremely important and urgent issue.

Concept drift detection and adaptation is an effective strategy for alleviating this problem. Techniques to detect drift fall into two categories: error-based drift detection and distribution-based detection [88], [89]. Once drift has been detected, several techniques can be applied to adapt a decision model. Adaptive decision-making reacts to drift by adapting the knowledge in the decision model to suit the concept drift environment. The challenge is not only to rapidly and accurately detect unforeseeable changes in underlying streaming data distribution, but also to quickly adapt a decision model once drift has been identified, to ensure reliable decision-making.

V. Summary

In this position paper, we have reported the key elements of D^3M and the potential developments in P-D^3M and NP-D^3M techniques. P-D^3M applies descriptive or predictive data analytical methods to predict uncertain or unknown parameters in decision objectives and constraints, which assists decision makers to build practical and precise programmable models for handling decision problems. NP-D^3M uses various machine learning approaches for data analysis by characterizing the decision problem and finding connections between the problem variables (input, internal and output variables) without explicit knowledge of the physical behavior of the decision model; rules, preference, and reinforcement learning drive the learning module to directly generate a decision solution. We believe that this paper highlights the essential elements of D^3M for researchers or practitioners, and it also provides guidelines on how to apply data-driven decision-making techniques to deal with a range of decision activities in various complex domains.

References

[1] H. A. Simon, *The New Science of Management Decision*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1977, p. 175.

[2] J. Lu, G. Zhang, D. Ruan, and F. Wu, *Multi-Objective Group Decision Making: Methods, Software and Applications With Fuzzy Set Techniques*. Singapore: World Scientific, 2007.

[3] G. Zhang, J. Lu, and Y. Gao, *Multi-Level Decision Making*. Berlin, Germany: Springer–Verlag, 2015.

[4] P. Zhang, X. Zhu, Y. Shi, L. Guo, and X. Wu, “Robust ensemble learning for mining noisy data streams,” *Decis. Support Syst.*, vol. 50, no. 2, pp. 469–479, 2011.

[5] P. Chen and C.-Y. Zhang, “Data-intensive applications, challenges, techniques and technologies: A survey on big data,” *Inf. Sci.*, vol. 275, pp. 314–347, 2014.

[6] H. Wang, Z. Xu, H. Fujita, and S. Liu, “Towards felicitous decision making: An overview on challenges and trends of big data,” *Inf. Sci.*, vol. 367–368, pp. 747–765, 2016.

[7] Q. Liu, C. Dang, and T. Huang, “A one-layer recurrent neural network for real-time portfolio optimization with probability criterion,” *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 14–23, Feb. 2013.

[8] J. Lu, J. Han, Y. Hu, and G. Zhang, “Multilevel decision-making: A survey,” *Inf. Sci.*, vol. 346, pp. 463–487, 2016.

[9] A. E. Lim, J. G. Shanthikumar, and Z. M. Shen, “Model uncertainty, robust optimization, and learning,” *Tuts. Oper. Res.*, vol. 3, pp. 66–94, 2006.

[10] D. Bertsimas and A. Thiele, “Robust and data-driven optimization: Modern decision-making under uncertainty,” *Tuts. Oper. Res.*, vol. 3, pp. 95–122, 2006.

[11] J. Ding, H. Modares, T. Chai, and F. L. Lewis, “Data-based multiobjective plant-wide performance optimization of industrial processes under dynamic environments,” *IEEE Trans. Ind. Inf.*, vol. 12, no. 2, pp. 454–465, Apr. 2016.

[12] B. A. Calfa, I. E. Grossmann, A. Agarwal, S. J. Bury, and J. M. Wassick, “Data-driven individual and joint chance-constrained optimization via kernel smoothing,” *Comput. Chem. Eng.*, vol. 78, pp. 51–69, 2015.

[13] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, “A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization,” *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 129–142, Feb. 2018.

[14] Y. Jin, “Surrogate-assisted evolutionary computation: Recent advances and future challenges,” *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 61–70, 2011.

[15] Y. Liu, H. Z. Huang, and X. Zhang, “A data-driven approach to selecting imperfect maintenance models,” *IEEE Trans. Rel.*, vol. 61, no. 1, pp. 101–112, Mar. 2012.

[16] V. Ramanujam and H. Balakrishnan, “Data-driven modeling of the airport configuration selection process,” *IEEE Trans. Human-Mach. Syst.*, vol. 45, no. 4, pp. 490–499, Aug. 2015.

[17] B. Swiderski, J. Kurek, and S. Osowski, “Multistage classification by using logistic regression and neural networks for assessment of financial condition of company,” *Decis. Support Syst.*, vol. 52, no. 2, pp. 539–547, 2012.

[18] M. S. Gerber, “Predicting crime using Twitter and kernel density estimation,” *Decis. Support Syst.*, vol. 61, pp. 115–125, 2014.

[19] U. Yolcu, E. Egrioglu, and C. H. Aladag, “A new linear & nonlinear artificial neural network model for time series forecasting,” *Decis. Support Syst.*, vol. 54, no. 3, pp. 1340–1347, 2013.

[20] V. Papathanasopoulou and C. Antoniou, “Towards data-driven car-following models,” *Transp. Res. Part C, Emerg. Technol.*, vol. 55, pp. 496–509, 2015.

[21] Y. Yuan and H. Xu, “An integrated search heuristic for large-scale flexible job shop scheduling problems,” *Comput. Oper. Res.*, vol. 40, no. 12, pp. 2864–2877, 2013.

[22] Y. Chu, F. You, J. M. Wassick, and A. Agarwal, “Integrated planning and scheduling under production uncertainties: Bi-level model formulation and hybrid solution method,” *Comput. Chem. Eng.*, vol. 72, pp. 255–272, 2015.

[23] S. Wang, L. Wang, Y. Xu, and M. Liu, “An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time,” *Int. J. Prod. Res.*, vol. 51, no. 12, pp. 3778–3793, 2013.

[24] M. Espinilla, J. Montero, and J. T. Rodríguez, “Computational intelligence in decision making,” *Int. J. Comput. Intell. Syst.*, vol. 7, no. supp1, pp. 1–5, 2014.

[25] M. Andrychowicz *et al.*, “Learning to learn by gradient descent by gradient descent,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3981–3989.

[26] K. Li and J. Malik, “Learning to optimize,” 2016, *arXiv:1606.01885*.

[27] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," 2016, *arXiv:1611.09940*.
[28] Y. Xia and J. Wang, "A general methodology for designing globally convergent optimization neural networks," *IEEE Trans. Neural Netw*., vol. 9, no. 6, pp. 1331–1343, Nov. 1998.
[29] S. Qin, X. Le, and J. Wang, "A neurodynamic optimization approach to bilevel quadratic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2580–2591, Nov. 2017.
[30] X. Le and J. Wang, "A two-time-scale neurodynamic approach to constrained minimax optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 620–629, Mar. 2017.
[31] G. Li, Z. Yan, and J. Wang, "A one-layer recurrent neural network for constrained nonsmooth invex optimization," *Neural Netw.*, vol. 50, pp. 79–89, 2014.
[32] Q. Liu and J. Wang, "A one-layer projection neural network for nonsmooth optimization subject to linear equalities and bound constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 5, pp. 812–824, May 2013.
[33] Q. Liu, S. Yang, and J. Wang, "A collective neurodynamic approach to distributed constrained optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 8, pp. 1747–1758, Aug. 2017.
[34] Z. Yan, J. Fan, and J. Wang, "A collective neurodynamic approach to constrained global optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1206–1215, May 2017.
[35] Z. Yan, J. Wang, and G. Li, "A collective neurodynamic optimization approach to bound-constrained nonconvex optimization," *Neural Netw.*, vol. 55, pp. 20–29, 2014.
[36] B. Li, J. Li, K. Tang, and X. Yao, "Many-objective evolutionary algorithms: A survey," *ACM Comput. Surv.*, vol. 48, no. 1, 2015, Art. no. 13.
[37] A. Wieczorek and R. Słowiński, "Generating a set of association and decision rules with statistically representative support and anti-support," *Inf. Sci.*, vol. 277, pp. 56–70, 2014.
[38] M. Kadziński, R. Słowiński, and S. Greco, "Multiple criteria ranking and choice with all compatible minimal cover sets of decision rules," *Knowl.-Based Syst.*, vol. 89, pp. 569–583, 2015.
[39] J. J. H. Liou, L. Yen, and G.-H. Tzeng, "Using decision rules to achieve mass customization of airline services," *Eur. J. Oper. Res.*, vol. 205, no. 3, pp. 680–686, 2010.
[40] R. Santos, P. Fonseca Teixeira, and A. Pais Antunes, "Planning and scheduling efficient heavy rail track maintenance through a decision rules model," *Res. Transp. Econ.*, vol. 54, pp. 20–32, 2015.
[41] K.-H. Chang, "A decision support system for planning and coordination of hybrid renewable energy systems," *Decis. Support Syst.*, vol. 64, pp. 4–13, 2014.
[42] C. Lee, B. Song, and Y. Park, "Design of convergent product concepts based on functionality: An association rule mining and decision tree approach," *Expert Syst. Appl.*, vol. 39, no. 10, pp. 9534–9542, 2012.
[43] W. Y. Loh, "Classification and regression trees," *Data Mining Knowl. Discovery*, vol. 1, no. 1, pp. 14–23, 2011.
[44] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.
[45] J. R. Quinlan, *C4. 5: Programs for Machine Learning*. Amsterdam, The Netherlands: Elsevier, 2014.
[46] J. de Oña, G. López, and J. Abellán, "Extracting decision rules from police accident reports through decision trees," *Accident Anal. Prevention*, vol. 50, pp. 1151–1160, 2013.
[47] S. Lee, S. Lee, and Y. Park, "A prediction model for success of services in e-commerce using decision tree: E-customer's attitude towards online service," *Expert Syst. Appl.*, vol. 33, no. 3, pp. 572–581, 2007.
[48] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Data Bases*, 1994, pp. 487–499.
[49] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *ACM Sigmod Rec.*, vol. 29, no. 2, pp. 1–12, 2000.
[50] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New algorithms for fast discovery of association rules," in *Proc. SIGKDD Conf. Knowl. Discovery Data Mining*, 1997, vol. 97, pp. 283–286.
[51] R. Khosravanian, M. Sabah, D. A. Wood, and A. Shahryari, "Weight on drill bit prediction models: Sugeno-type and Mamdani-type fuzzy inference systems compared," *J. Natural Gas Sci. Eng.*, vol. 36, pp. 280–297, 2016.
[52] S. Thakur, S. N. Raw, and R. Sharma, "Design of a fuzzy model for thalassemia disease diagnosis: Using mamdani type fuzzy inference system," *Int. J. Pharmacy Pharmaceutical Sci.*, vol. 8, no. 4, pp. 356–361, 2016.
[53] Y. Chen, B. Yang, A. Abraham, and L. Peng, "Automatic design of hierarchical Takagi–Sugeno type fuzzy systems using evolutionary algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 3, pp. 385–397, Jun. 2007.
[54] R. Yan, Z. Ma, Y. Zhao, and G. Kokogiannakis, "A decision tree based data-driven diagnostic strategy for air handling units," *Energy Buildings*, vol. 133, pp. 37–45, 2016.
[55] L.-X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1994.
[56] J. Chomicki, "Preference formulas in relational queries," *ACM Trans. Database Syst.*, vol. 28, no. 4, pp. 427–466, 2003.
[57] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: A survey," *Decis. Support Syst.*, vol. 74, pp. 12–32, 2015.
[58] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowl.-Based Syst.*, vol. 46, pp. 109–132, 2013.
[59] Z. Zhang, H. Lin, K. Liu, D. Wu, G. Zhang, and J. Lu, "A hybrid fuzzy-based personalized recommender system for telecom products/services," *Inf. Sci.*, vol. 235, pp. 117–129, 2013.
[60] D. Wu, J. Lu, and G. Zhang, "A fuzzy tree matching-based personalized e-learning recommender system," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 6, pp. 2412–2426, Dec. 2015.
[61] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1235–1244.
[62] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 111–112.
[63] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2643–2651.
[64] J. Davidson *et al.*, "The YouTube video recommendation system," in *Proc. 4th ACM Conf. Recommender Syst.*, 2010, pp. 293–296.
[65] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, 1996.
[66] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, 2016.
[67] D. Silver *et al.*, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, pp. 354–359, 2017.
[68] G. Litjens *et al.*, "A survey on deep learning in medical image analysis," *Med. Image Anal.*, vol. 42, pp. 60–88, 2017.
[69] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
[70] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.
[71] S. Martello, D. Pisinger, and D. Vigo, "The three-dimensional bin packing problem," *Oper. Res.*, vol. 48, no. 2, pp. 256–267, 2000.
[72] S.-H. Cho, H. Jang, T. Lee, and J. Turner, "Simultaneous location of trauma centers and helicopters for emergency medical service planning," *Oper. Res.*, vol. 62, no. 4, pp. 751–771, 2014.
[73] R. R. Yager and L. A. Zadeh, *An Introduction to Fuzzy Logic Applications in Intelligent Systems*. Berlin, Germany: Springer Science & Business Media, 2012.
[74] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 681–688.
[75] D. Bertsimas, D. B. Brown, and C. Caramanis, "Theory and applications of robust optimization," *SIAM Rev.*, vol. 53, no. 3, pp. 464–501, 2011.
[76] A. Kendall and Y. Gal "What uncertainties do we need in bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5580–5590.
[77] H. J. Escalante *et al.*, "Design of an explainable machine learning challenge for video interviews," in *Proc. Int. Joint Conf. Neural Netw*., 2017, pp. 3688–3695.
[78] L. Wang, "Discovering phase transitions with unsupervised learning," *Phys. Rev. B*, vol. 94, no. 19, 2016, Art. no. 195105.
[79] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Proc. Adv. Neural Inf. Process. Syst.* 2016, pp. 64–72.
[80] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowl.-Based Syst.*, vol. 80, pp. 14–23, 2015.
[81] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Fuzzy regression transfer learning in Takagi–Sugeno fuzzy models," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 6, pp. 1795–1807, Dec. 2017.

[82] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, “Granular fuzzy regression domain adaptation in Takagi–Sugeno fuzzy models,” *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 2, pp. 847–858, Apr. 2018.
[83] Q. Zhang, D. Wu, J. Lu, F. Liu, and G. Zhang, “A cross-domain recommender system with consistent information transfer,” *Decis. Support Syst.*, vol. 104, pp. 49–63, 2017.
[84] P. Hao, G. Zhang, L. Martinez, and J. Lu, “Regularizing knowledge transfer in recommendation with tag-inferred correlation,” *IEEE Trans. Cybern.*, vol. 49, no. 1, pp. 83–96, Jan. 2019.
[85] S. Sukhbaatar, J. Weston, and R. Fergus, “End-to-end memory networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2440–2448.
[86] Y. Teh *et al.*, “Distral: Robust multitask reinforcement learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4499–4509.
[87] A. Pentina, V. Sharmanska, and C. H. Lampert, “Curriculum learning of multiple tasks,” in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 5492–5500.
[88] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation,” *ACM Comput. Surv.*, vol. 46, no. 4, 2014, Art. no. 44.
[89] N. Lu, G. Zhang, and J. Lu, “Concept drift detection via competence models,” *Artif. Intell.*, vol. 209, pp. 11–28, 2014.

Jie Lu received the Ph.D. degree from the Curtin University of Technology, Perth, Australia, in 2000.

She is a Distinguished Professor and the Director of Centre for Artificial Intelligence, University of Technology Sydney, Australia. She has published six research books and 400 papers in *Artificial Intelligence*, IEEE Transactions on Fuzzy Systems, and other refereed journals and conference proceedings. Her main research expertise is in fuzzy transfer learning, decision support systems, concept drift, and recommender systems. Dr. Lu has received more than 20 Australian Research Council discovery grants and other research grants of over $4 million. She serves as the Editor-in-Chief for *Knowledge-Based Systems* (Elsevier) and the Editor-in-Chief for *International Journal on Computational Intelligence Systems* (Atlantis). She has delivered 20 keynote speeches at international conferences, and has chaired ten international conferences. She is a fellow of IFSA.

Zheng Yan received the bachelor’s degree in automation and computer-aided engineering and the Ph.D. degree in computational intelligence from The Chinese University of Hong Kong, Hong Kong, in 2010 and 2014, respectively.

He is a Chancellor’s Research Fellow at the Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia. His research interests include neural networks, deep learning, optimization, and model predictive control. He has published 50 papers and filed eight patents. Dr. Yan serves as an Area Editor for the *International Journal of Computational Intelligence Systems*. He is a Chartered Engineer in Australia.

Jialin Han received the Ph.D. degree from the University of Technology Sydney, Sydney, Australia, in 2016.

He is a Lecturer with the School of Information Management, Jiangxi University of Finance and Economics, China. He has published more than 20 papers in refereed journals and conference proceedings. His main research interests include the area of management decision making and optimization including data-driven decision making, multilevel decision making, multiobjective optimization, and their applications in the real world.

Guangquan Zhang received the Ph.D. degree in applied mathematics from the Curtin University of Technology, Perth, Australia, in 2001.

He is an Associate Professor and the Director of the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory in Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. He has authored four monographs, five textbooks, and 350 papers including 160 refereed international journal papers. His research interests include fuzzy machine learning, fuzzy optimization, and machine learning and data analytics.

Dr. Zhang has been awarded seven Australian Research Council (ARC) Discovery Project grants and many other research grants. He was awarded an ARC QEII fellowship in 2005. He has served as a member of the editorial boards of several international journals, as a Guest Editor of eight special issues for IEEE Transactions and other international journals, and has co chaired several international conferences and workshops in the area of fuzzy decision-making and knowledge engineering.