Chapter 1

Concepts of Machine Learning

Yousef Farhang[1]

Abstract

This chapter introduces the fundamental concepts and principles of machine learning, serving as a theoretical foundation for the subsequent chapters of the book. It provides a comprehensive overview of the main learning paradigms, including supervised, unsupervised, semi-supervised, and reinforcement learning, along with essential terminology and notations commonly used in machine learning research and applications. Key topics such as data representation, preprocessing techniques, feature engineering, and the learning process are discussed to highlight how raw data is transformed into meaningful knowledge through computational models. The chapter also explains core concepts related to model training, generalization, overfitting, and the bias–variance tradeoff, which are critical for understanding model performance and reliability.

In addition, fundamental ideas in optimization and model evaluation are presented, including cost functions, gradient-based learning, and standard performance metrics. Ethical and practical considerations, such as data bias, interpretability, and privacy, are briefly addressed to emphasize responsible use of machine learning technologies. Overall, this chapter establishes a conceptual framework that enables readers to better understand, design, and critically evaluate machine learning systems.

1.1. Introduction

Machine Learning (ML) is originated from computer science (CS) and Artificial Intelligence (AI), which addresses systems that are able to learn from data, rather than merely following the programmed instructions explicitly. Moreover, ML has a close relationship with optimization and statistics, delivering both theory and methods to the field. ML is applied to various

1 Asst. Prof. Dr. Yousef Farhang, Istanbul Esenyurt University, Faculty of Engineering and Architecture, Department of Computer Engineering, youseffarhang@esenyurt.edu.tr, 0000-0001-9348-2624.

https://doi.org/10.58830/ozgur.pub1236.c5001

computing tasks in which designing and programming rule-based, explicit algorithms is impractical. In some situations, ML, pattern recognition, and data mining are conflated.

Arthur Samuel, in 1959, defined ML as a "field of study that gives computers the ability to learn without being clearly programmed". In general, ML and data mining are confused with each other since they use the same methods and they significantly overlap. These two can be described as follows. ML is focused on prediction based on recognized properties that are learned from training data. On the other hand, data mining is concentrated on discovering the (previously) unknown properties in the data. The two areas are overlapped in many aspects: in data mining, numerous machine learning methods are used, but with a diverse goal in mind. However, ML makes use of data mining methods as "unsupervised learning" or as a preprocessing step for the improvement of learner accuracy. The present research is focused on ML.

Regarding to the tasks, ML can be divided into three types, namely supervised learning, semi-supervised learning, and unsupervised learning. Figure 2.1 demonstrates the types of ML.

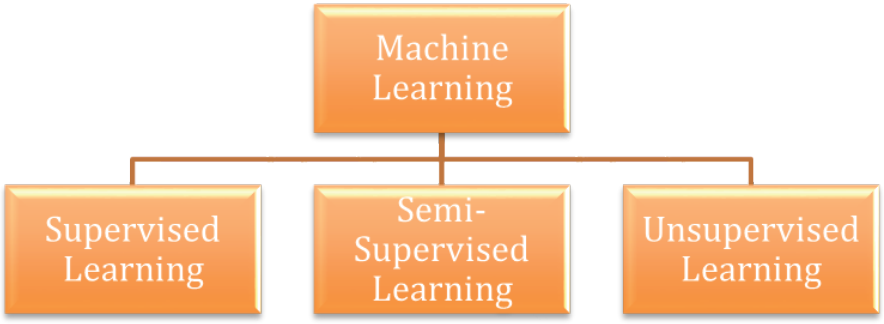

Figure 1.1. Types of Machine Learning

In the supervised learning, example inputs and desired outputs are provided by teacher, aiming at learning a general rule that maps inputs to outputs. The supervised learning is the ML task in which a function is inferred from labeled training data. One of appropriate examples of the supervised learning is classification in which labeled data are used for solving given problems. The supervised learning is the ML task of tracing a function from labeled training data that is consisted of a set of training instances. In the supervised learning, each instance is a pair that is composed of an input, which is typically a vector, and a desired output value, which is also known as supervisory signal. An algorithm that is based on supervised learning makes analysis on training data

and generates an inferred function; the function is applicable to mapping new instances. An optimal scenario helps the algorithm determine correctly the class labels for unseen instances. To this purpose, the learning algorithm should generalize from training data to unseen situations "reasonably".

In the semi-supervised learning, both labeled and unlabeled examples are combined for the generation of a proper function or classifier. The semi-supervised learning is considered as a class of the tasks and techniques of the supervised learning, which typically employs a small amount of labeled data together with a large amount of unlabeled data for training. The semi-supervised learning is situated between supervised learning (with entirely labeled training data) and unsupervised learning (with no labeled training data). Several researchers working on ML have shown that unlabeled data, when utilized together with a small amount of labeled data, are capable of producing significant improvement in the accuracy level of learning.

The performance of the unsupervised learning algorithms is based on unlabeled examples, i.e., input where there is not known desired output. In this case, the goal is discovering the structure of the data, for example using a cluster analysis, not generalizing a mapping from inputs to outputs. In the next section, the unsupervised learning is elaborated.

1.2. Supervised Learning

Supervised learning represents one of the most fundamental and widely used paradigms in machine learning, in which a learning algorithm is trained using labeled data. In this framework, each training instance consists of an input vector together with a corresponding target output, commonly referred to as a label. The primary objective of supervised learning is to learn a mapping function that accurately predicts outputs for previously unseen data.

Formally, supervised learning attempts to approximate an unknown function that maps input variables to output responses based on examples provided during the training phase. The learning process is guided by a supervisory signal that evaluates prediction errors and enables the model to iteratively adjust its internal parameters. Unlike unsupervised learning, where structure must be inferred without guidance, supervised learning benefits from explicit feedback regarding prediction correctness.

Supervised learning problems are generally categorized into two major types: classification and regression. Classification tasks involve assigning discrete labels to input data, such as identifying whether an email is spam or non-spam, diagnosing diseases based on medical records, or recognizing objects in images. Regression tasks, on the other hand, focus on predicting

continuous numerical values, including temperature forecasting, energy demand estimation, and financial trend prediction.

A wide variety of algorithms have been developed for supervised learning applications. Classical methods include linear regression, decision trees, k-nearest neighbors, and support vector machines, while modern approaches rely heavily on artificial neural networks and deep learning architectures. During training, model performance is typically evaluated using loss functions that quantify the difference between predicted outputs and true labels. One of the most important challenges in supervised learning is achieving strong generalization capability, meaning that the learned model performs well not only on training data but also on unseen datasets. Issues such as overfitting and under fitting arise when model complexity is not properly balanced with available data. Consequently, validation techniques, regularization methods, and cross-validation strategies are commonly employed to ensure reliable predictive performance. Due to its ability to learn predictive relationships directly from historical data, supervised learning has become a cornerstone of modern artificial intelligence systems and is extensively applied in areas such as medical diagnosis, speech recognition, autonomous systems, recommender systems, and intelligent decision support.

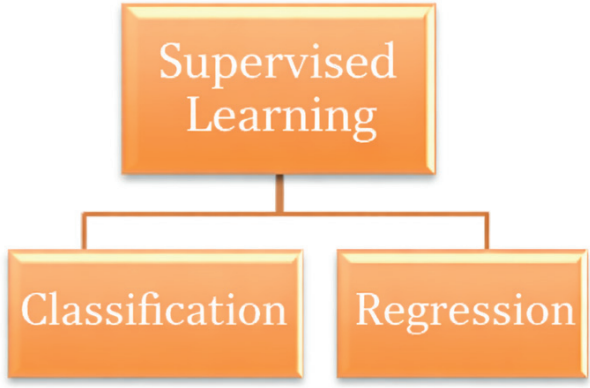

Figure 1.2. Types of the Supervised Learning

Figure 1.2. illustrates the fundamental structure of the supervised learning paradigm, where learning is performed using labeled training data. As shown in the figure, supervised learning problems are primarily categorized into two major types: classification and regression. Classification focuses on assigning discrete class labels to input data based on learned decision boundaries, while regression aims to predict continuous numerical values through functional

approximation. These two problem categories represent the core predictive tasks addressed by supervised learning algorithms. During the learning process, models utilize labeled examples to minimize prediction errors and improve performance through iterative optimization. The hierarchical structure presented in the figure highlights how supervised learning forms the foundation of predictive modeling systems widely applied in domains such as medical diagnosis, pattern recognition, financial forecasting, and intelligent decision-making systems.

1.3. Semi-Supervised Learning

Semi-supervised learning constitutes an intermediate learning paradigm that combines characteristics of both supervised and unsupervised learning. In many real-world applications, obtaining labeled data is expensive, time-consuming, or requires expert knowledge, whereas large quantities of unlabeled data are readily available. Semi-supervised learning addresses this limitation by utilizing a small set of labeled samples together with a substantially larger collection of unlabeled data during training.

The central assumption underlying semi-supervised learning is that unlabeled data contains valuable structural information about the underlying distribution of the dataset. By exploiting this structure, learning algorithms can improve classification or prediction accuracy beyond what is achievable using labeled data alone. This paradigm effectively reduces annotation costs while maintaining high model performance.

Several theoretical assumptions guide semi-supervised learning methods. The smoothness assumption states that data points located close to one another in feature space are likely to share similar labels. The cluster assumption suggests that decision boundaries should lie in low-density regions separating clusters of data. Additionally, the manifold assumption proposes that high-dimensional data often resides on lower-dimensional manifolds that can be exploited for learning.

A variety of techniques have been developed within the semi-supervised learning framework. Self-training methods iteratively label confident unlabeled samples and incorporate them into the training set. Co-training approaches employ multiple classifiers trained on different feature subsets to improve learning reliability. Graph-based methods model relationships among samples using similarity graphs, allowing label information to propagate through connected data points. More recently, pseudo-labeling and consistency-regularization techniques have gained popularity in deep learning applications.

Semi-supervised learning has demonstrated remarkable success in domains where labeled data is scarce, including medical imaging, speech processing, natural language understanding, remote sensing, and anomaly detection. By integrating supervised guidance with unsupervised structure discovery, semi-supervised learning provides a practical balance between learning accuracy and data annotation cost. Consequently, semi-supervised learning serves as a critical bridge between fully supervised and fully unsupervised paradigms, enabling scalable machine learning solutions in modern data-intensive environments.

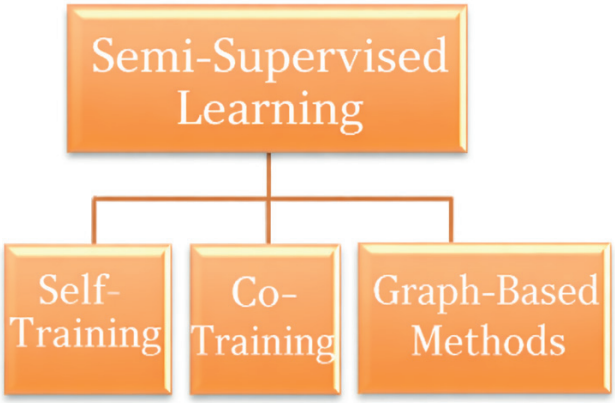

Figure 1.3. Types of the Semi-Supervised Learning

Figure 1.3. presents the conceptual framework of semi-supervised learning, which integrates both labeled and unlabeled data within the learning process. As depicted in the figure, semi-supervised learning techniques commonly include self-training, co-training, and graph-based methods. Self-training approaches iteratively expand labeled datasets by assigning pseudo-labels to confidently predicted unlabeled samples. Co-training methods employ multiple learners trained on complementary feature subsets to enhance learning reliability. Graph-based approaches model relationships among data samples using similarity graphs, enabling label information to propagate throughout the dataset. The structure illustrated in the figure demonstrates how semi-supervised learning bridges the gap between supervised and unsupervised paradigms by exploiting the intrinsic structure of unlabeled data while maintaining guidance from labeled examples. This capability significantly reduces annotation cost while improving model generalization in real-world applications where labeled data availability is limited.

1.4. Unsupervised Learning

In ML, the unsupervised learning problem is that of attempting to find the structure hidden in unlabeled data. As the examples that are provided for the learner are unlabeled, there is not error or reward signal for the evaluation of a potential solution. This makes difference between the unsupervised learning and the supervised learning and reinforcement learning. The unsupervised learning is in a close relationship with the density estimation problem in statistics. On the other hand, the unsupervised learning involves several other techniques used for summarizing and explaining the most important characteristics of data. Several methods that are used in unsupervised learning are designed based on data mining methods that are applied to preprocessing data.

The approaches based on the unsupervised learning can be divided into three types, namely dimensionality reduction, self-organizing map, and clustering algorithm. Figure 1.4. presents the types of the unsupervised learning.

Figure 1.4 Types of the Unsupervised Learning

In ML, dimensionality reduction and statistics is a process in which the number of random variables under consideration is reduced. This is separated into feature selection and feature extraction. In case of high-dimensional datasets, the dimensionality reduction is generally carried out before using a *K*-Nearest Neighbors (*K*-NN) algorithm to avoid the impacts of the dimensionality curse.

A self-organizing map (SOM) is a type of Artificial Neural Network (ANN) that is trained using the unsupervised learning in order to generate a low-dimensional, discretized representation of input space of the training samples, which is named a map. SOMs differ from other ANNs; SOMs employ a neighborhood function in order to preserve the topological properties of the input space.

Clustering refers to the task of grouping a set of objects in such a way that objects in the same cluster are more similar to each other than to those in other clusters. In the next section, clustering analysis will be fully described.

1.4.1. Dimensionality Reduction

Dimensionality reduction is an essential technique in unsupervised learning that aims to reduce the number of variables under consideration while preserving the intrinsic structure and meaningful information contained in the original dataset. In many real-world applications, datasets often contain a large number of features, some of which may be redundant, irrelevant, or noisy. The presence of such high-dimensional data increases computational complexity and may negatively affect learning performance, a phenomenon commonly referred to as the *curse of dimensionality*.

The primary objective of dimensionality reduction is to transform data from a high-dimensional space into a lower-dimensional representation that retains the most significant characteristics of the original data distribution. By reducing dimensionality, machine learning models become computationally efficient, less sensitive to noise, and more capable of generalization.

Dimensionality reduction techniques are generally categorized into two main approaches: feature selection and feature extraction. Feature selection focuses on identifying a subset of relevant original variables without modifying them, whereas feature extraction generates new variables through mathematical transformations or combinations of existing features.

Several well-established techniques have been developed for dimensionality reduction. Linear approaches such as Principal Component Analysis (PCA) aim to maximize variance preservation while minimizing information loss. Independent Component Analysis (ICA) attempts to identify statistically independent components within data. In contrast, nonlinear dimensionality reduction methods, including manifold learning techniques and auto encoders, capture complex relationships that cannot be represented through linear transformations.

Dimensionality reduction plays a critical role in visualization and exploratory data analysis by enabling high-dimensional datasets to be represented in two- or three-dimensional spaces. Furthermore, it is frequently employed as a preprocessing step prior to clustering algorithms such as K-means or hierarchical clustering in order to improve distance measurement reliability and clustering stability.

Consequently, dimensionality reduction serves as a fundamental bridge between raw high-dimensional data and efficient machine learning models, facilitating improved learning performance and interpretability in large-scale data environments.

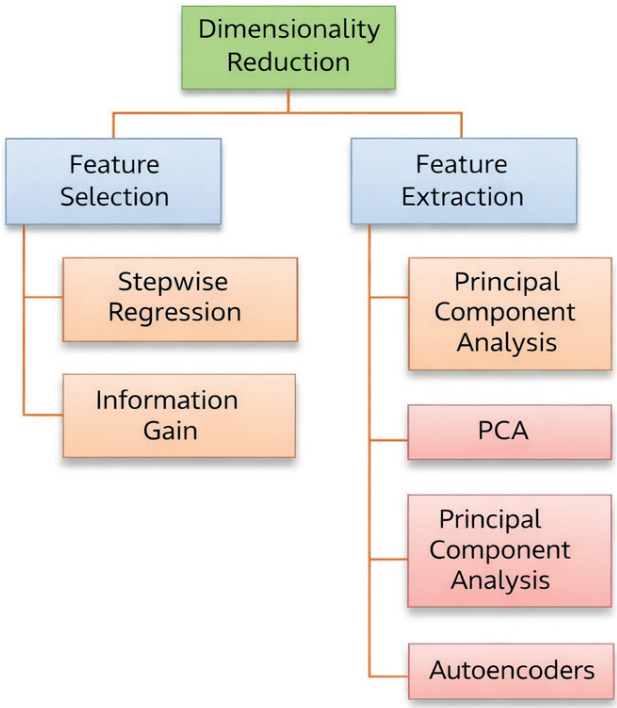

Figure 1.5. Types of the Dimensionality Reduction Techniques

Figure 1.5. illustrates the general taxonomy of dimensionality reduction techniques used in unsupervised learning. As shown in the figure, dimensionality reduction methods are broadly categorized into two principal groups: feature selection and feature extraction. Feature selection approaches aim to identify the most informative subset of original variables while eliminating redundant or irrelevant attributes that may negatively affect learning performance. Techniques such as stepwise regression and information gain analysis evaluate feature importance based on statistical or information-theoretic criteria. In contrast, feature extraction methods transform the original data into a new lower-dimensional representation through mathematical projection or nonlinear transformation. Classical approaches such as Principal Component Analysis (PCA) reduce dimensionality by maximizing variance preservation, whereas

modern techniques such as auto encoders learn compact representations using neural network architectures. Overall, dimensionality reduction improves computational efficiency, enhances visualization capability, and increases the stability of subsequent learning algorithms such as clustering and classification models.

1.4.2. Self-Organizing Map

The Self-Organizing Map (SOM) is a neural network–based unsupervised learning technique designed to produce a low-dimensional representation of high-dimensional input data while preserving the topological relationships among data samples. Originally introduced by Teuvo Kohonen, SOM provides an effective mechanism for visualization, clustering, and exploratory pattern discovery.

Unlike traditional artificial neural networks that rely on supervised learning, SOM operates through competitive learning. In this framework, neurons compete to represent input data, and only the most suitable neuron—referred to as the *winning neuron*—is activated during each training iteration. Neighboring neurons surrounding the winner are simultaneously updated according to a neighborhood function, enabling the network to maintain spatial relationships among similar data points.

The architecture of a self-organizing map typically consists of a two-dimensional grid of neurons, where each neuron is associated with a weight vector having the same dimensionality as the input data. During training, input vectors are repeatedly presented to the network, and neuron weights gradually adapt to approximate the distribution of the input space. As learning progresses, similar observations become mapped to nearby neurons, resulting in an organized representation of data structure.

One of the most important characteristics of SOM is its ability to preserve topology. This means that data points that are close in the original feature space remain close within the generated map. This property distinguishes SOM from many other clustering approaches and makes it particularly useful for analyzing complex multidimensional datasets.

Self-organizing maps are widely applied in numerous domains, including pattern recognition, image processing, bioinformatics, financial analysis, and anomaly detection. They are especially valuable when prior knowledge about class labels is unavailable and when visualization of hidden data structures is required.

In modern machine learning workflows, SOM can also serve as a preprocessing or clustering mechanism, supporting dimensionality reduction and facilitating subsequent analytical tasks. Due to its interpretability and visualization capability, SOM remains an important unsupervised learning tool despite the emergence of more advanced deep learning techniques.

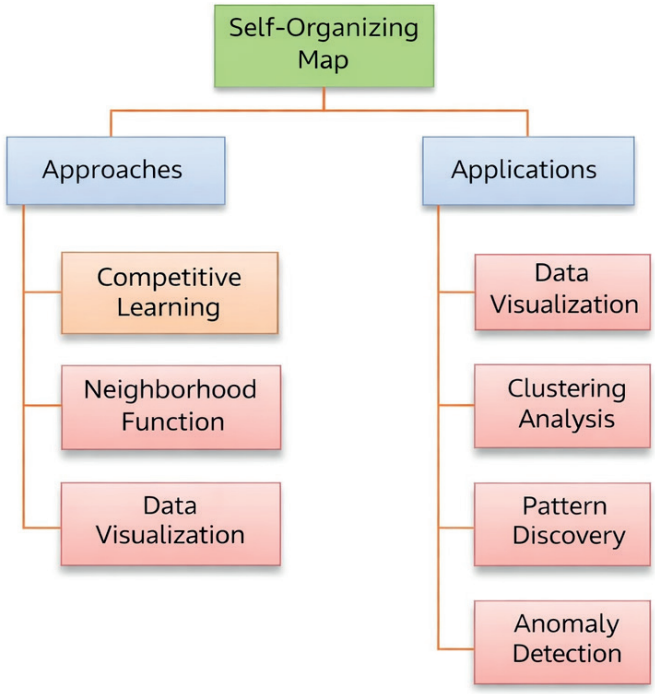

Figure 1.6. Types of the Self-Organizing Map Structure and Applications

Figure 1.6. presents the conceptual framework of the Self-Organizing Map (SOM) and highlights its fundamental operational components and application domains. The SOM learning mechanism is primarily based on competitive learning, where neurons compete to represent input patterns, followed by adaptation governed by a neighborhood function that preserves topological relationships among data samples. Through iterative learning, similar input vectors are mapped to neighboring regions of the network, enabling meaningful visualization of complex multidimensional datasets. As illustrated in the figure, SOM techniques are widely applied in data visualization, clustering analysis, pattern discovery, and anomaly detection tasks. The ability of SOM to simultaneously perform dimensionality reduction

and clustering makes it an effective exploratory analysis tool, particularly in situations where labeled data are unavailable. Consequently, SOM serves as an important bridge between neural computation and unsupervised data mining methodologies.

1.4.3. Clustering Algorithm

Cluster analysis or clustering algorithm is to group a set of objects so that objects in a group or cluster have a higher degree of similarity to each other compared to similarity to members of other clusters. Clustering is a mainly applied to exploratory data mining and this is a technique commonly used for statistical data analysis, and it is employed in several fields such as ML, information retrieval, pattern recognition, bioinformatics, and image analysis (Jain, 2010). Cluster analysis per se is not considered as an algorithm; rather it is taken onto account as a general task to be done. This task can be performed by different algorithms that significantly differ in the notion regarding to what constitutes a cluster and how to find them in an efficient way. The popular notions of clusters involve groups in which there are small distances amongst the members of cluster, dense areas of data space, intervals or certain statistical distributions.

Clustering is applied to several fields, for example information retrieval and knowledge discovery. This helps to find more quickly related information. This way, researchers can stay up to date with the latest findings in their fields. Currently, clustering has attracted the attention of many scholars as a tool for classification, decision making, information extraction, and pattern analysis.

Vladimir Estivill-Castro believes that the term "cluster" cannot be defined precisely, and this condition has led to emergence of so many clustering algorithms. The common denomination is a group of data objects. Though, different researchers make use of various cluster models for each of which various algorithms can be presented. As found by various algorithms, the notion of cluster significantly varies in its properties. To understand these "cluster models" is the most important point in understanding differences that exist among different algorithms.

Clustering analysis can be divided into several models: connectivity models, distribution models, density models, subspace models, group models, graph-based models, and centroid models. Figure 1.7 shows different types of clustering algorithm.

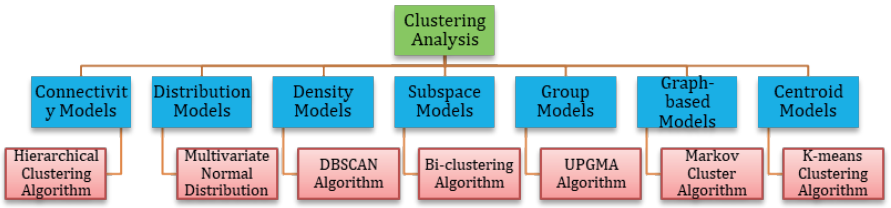

Figure 1.7. Types of Clustering Algorithm

Connectivity model is a model for clustering that is designed based on distance connectivity. One of major examples for connectivity model is hierarchical clustering algorithm. The hierarchical clustering is an algorithm of cluster analysis which attempts to build a hierarchy of clusters. Bottom up and top down are two strategies used in the hierarchical clustering.

Distribution model is a model for clustering in which clusters are modeled using statistical distributions. One of examples for distribution model is multivariate normal distributions used by the expectation-maximization algorithm.

Density model is a model for clustering in which clusters are defined as connected dense regions in data space. Two main examples for density model are Density-Based Spatial Clustering of Applications with Noise algorithm (DBSCAN algorithm) and Ordering Points to Identify the Clustering Structure algorithm (OPTICS algorithm).

Subspace model is a model for clustering in which clusters are modeled with both cluster members and relevant attributes. An example for subspace model is Bi-clustering algorithm that is also known as Co-clustering algorithm. Group model is a model for clustering in which algorithms do not provide a refined model for their results and just provide the grouping information. An example for group model is Un-Weighted Pair Group Method with Arithmetic Mean algorithm (UPGMA Algorithm).

Graph-based model is one of the models for clustering, which is a subset of nodes in a graph such that every two nodes in the subset are connected with an edge that can be considered as a prototypical form of cluster. One instance for this model is Markov Cluster Algorithm. Centroid is a model for clustering in which the similarity of two clusters is defined as the similarity of their centroids. One example for this model is *K*-means clustering algorithm that represents each cluster by a single mean vector. The *K*-means clustering algorithm is explained in the next section.

1.5. Basic Terminology and Notation

Machine learning systems rely on a set of fundamental terminologies and mathematical notations that provide a common framework for describing datasets, models, and learning processes. Understanding these concepts is essential for interpreting machine learning algorithms and evaluating their performance in practical applications.

A dataset represents a structured collection of observations used for training and evaluating learning models. Typically, datasets are organized in tabular form where rows correspond to individual samples and columns represent measurable attributes known as features. Each observation within the dataset is referred to as an instance or sample, describing a single entity such as a patient record, an image, or a sensor measurement.

A feature denotes an individual characteristic or measurable property of an instance that serves as input to the learning algorithm. Features may be numerical, categorical, ordinal, or binary depending on the nature of the problem domain. In supervised learning environments, each instance is associated with a label, representing the ground-truth output that the model attempts to predict.

Machine learning datasets are commonly divided into three subsets: the training set, validation set, and testing set. The training set is used to learn model parameters, the validation set assists in hyper parameter tuning and model selection, and the testing set provides an unbiased evaluation of generalization performance.

A model can be defined as a mathematical function that maps input space X to output space Y. Each possible configuration of model parameters represents a hypothesis, and the collection of all hypotheses forms the hypothesis space. Learning occurs through optimization of a loss function, which measures the discrepancy between predicted outputs and true values.

Standard notation frequently used in machine learning includes input vectors x_i, true outputs y_i, predicted outputs y^i, dataset D, and model parameters W. Consistent use of notation improves clarity, reproducibility, and scientific communication across machine learning research.

1.6. Data Representation and Preprocessing

Data representation and preprocessing constitute one of the most critical stages in the machine learning workflow. Real-world data is often incomplete, noisy, inconsistent, or heterogeneous, making preprocessing an essential prerequisite before applying learning algorithms.

Machine learning data appears in multiple formats, including numerical data, categorical attributes, textual information, and image-based representations. Numerical data consists of measurable quantities such as temperature or income values, whereas categorical data represents qualitative properties such as gender or geographic location. Since many learning algorithms operate on numerical inputs, categorical variables must be encoded using techniques such as label encoding or one-hot encoding. Textual data requires specialized preprocessing steps including tokenization, stop-word removal, and vectorization methods such as Bag-of-Words or TF–IDF representations. Similarly, image data is represented as multidimensional pixel matrices and typically undergoes normalization, resizing, and augmentation procedures prior to learning.

Data cleaning is another essential preprocessing step aimed at removing duplicate records, correcting inconsistencies, and detecting outliers. Additionally, handling missing values plays a crucial role in maintaining dataset integrity. Common strategies include deletion methods, statistical imputation, and model-based estimation.

Feature scaling techniques such as normalization and standardization ensure that variables contribute equally during model training. Without proper scaling, features with larger numerical ranges may dominate optimization processes, leading to unstable learning behavior. Effective preprocessing significantly improves model accuracy, accelerates convergence, and enhances generalization capability, making it a foundational component of modern machine learning systems.

1.7. Feature Engineering

Feature engineering refers to the process of transforming raw data into meaningful representations that improve machine learning performance. It is widely recognized that the success of a learning system often depends more on feature quality than on algorithmic complexity.

Feature engineering encompasses three primary operations: feature selection, feature extraction, and feature scaling. Feature selection aims to identify the most informative variables while eliminating redundant or irrelevant attributes that may introduce noise or increase computational cost. Common selection approaches include filter methods based on statistical measures, wrapper methods relying on predictive performance, and embedded techniques integrated within learning algorithms. Feature extraction, in contrast, generates new features by transforming original variables into compact representations. Techniques such as Principal Component Analysis, signal transformation

methods, and deep neural embedding's enable efficient representation of complex data structures.

Feature scaling ensures numerical consistency among variables, particularly in algorithms sensitive to distance calculations or gradient optimization. Methods such as Min–Max scaling, standardization, and robust scaling are widely applied to stabilize learning processes. Well-designed feature engineering improves interpretability, reduces overfitting risk, and enhances predictive accuracy. In many real-world applications, carefully engineered features enable simpler models to outperform more complex algorithms applied to poorly prepared data.

1.8. Conclusion

Machine learning has emerged as one of the most transformative technologies of modern computational science, enabling intelligent systems to learn from data and make informed decisions without explicit programming. This book chapter presented a comprehensive introduction to the fundamental concepts underlying machine learning, establishing a solid theoretical foundation for understanding advanced learning models and methodologies. The discussion began with an overview of machine learning principles and learning paradigms, including supervised, unsupervised, semi-supervised, and reinforcement learning approaches. These paradigms demonstrate how different learning strategies address diverse problem domains depending on data availability and learning objectives.

Essential terminology and notations were introduced to provide a common framework for describing datasets, models, hypotheses, and evaluation procedures. Proper understanding of these concepts is crucial for interpreting machine learning algorithms and conducting scientifically valid experiments. The chapter further emphasized the importance of data representation and preprocessing, highlighting how raw data must be transformed into structured and meaningful formats before learning can occur. Feature engineering techniques, including feature selection, extraction, and scaling, were discussed as key mechanisms for improving model efficiency and predictive performance.

The learning process was examined through model training, generalization capability, and challenges such as overfitting and under fitting. The bias–variance tradeoff was presented as a central principle governing model complexity and prediction reliability. In addition, systematic model evaluation techniques and optimization concepts were explored to demonstrate how learning algorithms achieve optimal performance through iterative improvement.

Finally, ethical and practical considerations—including data bias, interpretability, and privacy protection—were addressed to underline the responsibility associated with deploying machine learning systems in real-world environments. In conclusion, mastering the foundational concepts of machine learning is essential for researchers, engineers, and practitioners seeking to design robust, efficient, and trustworthy intelligent systems. The principles introduced in this chapter provide the conceptual basis for more advanced topics such as deep learning, hybrid intelligent systems, optimization strategies, and real-world machine learning applications discussed in subsequent chapters of this book.

References

Mitchell, T. M. (1997). Machine Learning. New York, USA: McGraw-Hill.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer, New York.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer.

Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.

Alpaydin, E. (2020). Introduction to Machine Learning (4th ed.). MIT Press.

Russell, S., & Norvig, P. (2021). Artificial Intelligence: A Modern Approach (4th ed.). Pearson.

Geron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media.

Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction (2nd ed.). MIT Press.

Han, J., Kamber, M., & Pei, J. (2011). Data Mining: Concepts and Techniques (3rd ed.). Morgan Kaufmann.

Kelleher, J. D., Namee, B. M., & D'Arcy, A. (2020). Fundamentals of Machine Learning for Predictive Data Analytics. MIT Press.

Zhou, Z.-H. (2021). Machine Learning. Springer Nature.

Farhang, Y. (2016) Hybrid Optimization for K-means Clustering Learning Enhancement.

Farhang, Y. (2017) Face Extraction from Image Based on K-means Clustering Algorithms.

Farhang, Y. (2017) Development of the Meta-Heuristic PSOGA with K-means Algorithm.