Algorithms of Machine Learning and Application for Signal Compensation

Yudong Peng *

Department of Electrical and Computer engineering, the Ohio state University, Ohio, United Sate

* Corresponding Author Email: peng.804@buckeyemail.osu.edu

Abstract. The advent of machine learning has inaugurated a new epoch, where computers acquire patterns and relationships from data, obviating the need for explicit programming. In this context, supervised learning stands as a cornerstone. This study investigates the importance of decision trees, K-Means, and boosting in the context of signal compensation scenarios. The synergy between these techniques is profound. Decision trees frequently serve as prime contenders for base learners in ensemble approaches like boosting, augmenting predictive precision while encapsulating complex temporal associations. Furthermore, K-Means' ability to segment data into temporal clusters can facilitate preprocessing, thereby enhancing subsequent analysis and boosting model efficacy. Within practical applications, these techniques synergistically address time compensation challenges. Imagine a scenario where historical data is harnessed to forecast time delays in financial transactions. Employing supervised learning through decision trees, key features contributing to delays could be discerned. Boosting could subsequently refine this prediction model by prioritizing instances with temporal disparities, thereby enhancing its accuracy. In parallel, K-Means could segment data into time-related clusters, revealing insights into the temporal patterns governing these delays. In summation, the triumvirate of supervised learning, unsupervised learning, and ensemble learning, enriched by decision trees, K-Means, and boosting, form the bedrock of machine learning's application in time compensation domains.

Keywords: Machine learning, signal compensation, K-Means, boosting, decision tree.

1. Introduction

The progression of machine learning algorithms entails the formation and enhancement of mathematical models and computational procedures that empower computers to acquire patterns, extract insights from data, and render predictions or decisions without requiring explicit programming. This field combines expertise from computer science, mathematics, and statistics to design algorithms that allow machines to improve their performance over time as they receive more data. The term "artificial intelligence" was coined at the Dartmouth Workshop who coined the term "machine learning" to describe a field of study that imparts learning capability to computers without the need for explicit programming, marking the formal beginning of AI as a field of study [1]. The development of decision tree algorithms, perceptron, and early neural networks laid the foundation for machine learning algorithms [2]. The development of machine learning shifted towards rule-based systems and expert systems, and the focus also turned to symbolic machine learning, Bayesian networks, and more advanced neural network architectures [3]. The advent of the internet and increased availability of digital data sparked renewed interest in data-driven machine learning. at the early 2000s, the expansion of the internet and the accessibility of extensive datasets fueled advancements in machine learning algorithms, particularly in areas like natural language processing and image recognition [4]. The rise of deep learning structures, notably convolutional neural networks (CNNs) and recurrent neural networks (RNNs), transformed the landscape of pattern recognition assignments. At late 2000s-2010s, the availability of powerful GPUs and advances in parallel computing greatly accelerated the training of complex machine learning models [5].

The development of machine learning algorithms from 2020 to the present has been marked by remarkable advancements and breakthroughs across various domains, and notable trends, techniques, and developments. The abundant availability of data from diverse origins such as social media, sensors, and the internet has stimulated the creation of algorithms capable of effectively managing

substantial datasets. Technologies like distributed computing and cloud infrastructure have facilitated the training and deployment of large-scale machine learning models [6]. Transfer learning has gained prominence as a technique where models undergo pretraining on extensive datasets and are subsequently refined for specific tasks. Pretrained models, e.g., BERT for natural language processing, have demonstrated impressive results and reduced the need for training from scratch [7]. There are breakthroughs in reinforcement learning that is a technique where agents learn through interactions with an environment, has achieved remarkable successes, e.g., AlphaGo's victory over human champions and the development of more capable robotic systems [8]. Generative adversarial networks, a class of neural networks pitting one network against another, have enabled the generation of highly realistic images, videos, and even text. They find applications in art, content creation, and data augmentation [9]. Overall, the years from 2020 to the present have witnessed rapid growth in machine learning research, driven by a combination of improved algorithms, increased computing power, availability of data, and a growing interdisciplinary community collaborating to push the boundaries of what is possible with machine learning.

This study will focus on some commonly used machine learning algorithms on signal compensation. These algorithms cover a range of tasks and applications in machine learning. Incorporating these techniques into signal compensation tasks can lead to improved accuracy, adaptability, and efficiency in compensating for various forms of signal degradation or interference. However, the specific implementation and success of these techniques would depend on the characteristics of the signals being compensated and the domain in which they are applied [10].

2. Commonly Used Algorithms in Machine Learning

Machine Learning utilizes various algorithms to address data-related challenges. Data scientists frequently underscore that a one-size-fits-all algorithm doesn't exist. The selection of an algorithm is contingent upon the particular issue at hand, the variables in play, the appropriate model, and other pertinent considerations [11]. Machine learning is categorized into distinct domains, including supervised learning, unsupervised learning, semi-supervised learning, reinforcement learning, multi-task learning, ensemble learning, neural network, and instance-based learning. This study will particularly focus on the comprehensive exploration of supervised learning, unsupervised learning, and ensemble learning.

2.1. Supervised Learning

Supervised learning is a category within machine learning wherein the algorithm learns from labeled training data. This entails the provision of input-output pairs, commonly referred to as examples. The purpose of supervised learning is to learn a mapping from inputs to outputs so that the algorithm can make accurate predictions or classifications on unseen data. Supervised learning has achieved significant success in practical real-world scenarios. Supervised learning finds application across diverse fields, encompassing text and web-related domains. In the realm of machine learning, it's also referred to as classification or inductive learning. This approach mirrors human learning from prior encounters to enhance our competence in real-world activities. As computers lack personal experiences, machine learning draws insights from historical data, gathered in the past to mirror past encounters in various practical applications [12]. Input data set that is consists of various attributes and features of data is categorized testing data and training data. The datasets used to train the model consists of input-output pairs, where each example is associated with its corresponding output label. This labeled data is used by the algorithm to learn the underlying patterns and relationships. A sketch is shown in Fig. 1.

During training, the algorithm feeds the input features into the model that is the algorithm's representation of the learned patterns from the training data, calculates the predicted outputs, and compares them to the actual labels Utilizing a loss function to measure the disparity between the model's predictions and the true output labels, aiming to minimize discrepancies and enhance

precision. Once the model is trained and fine-tuned, it's tested on a separate set of data that it has never seen before. This testing data provides a final assessment of the model's generalization ability. After training and testing, the model can be deployed to make predictions on new, unlabeled data, generating output labels that are desired outcomes or classifications that the algorithm aims to predict based on the input features. Overall, supervised learning involves the iterative process of feeding labeled data to a model, refining the model's parameters, and evaluating its performance to create an accurate predictor or classifier for new data.

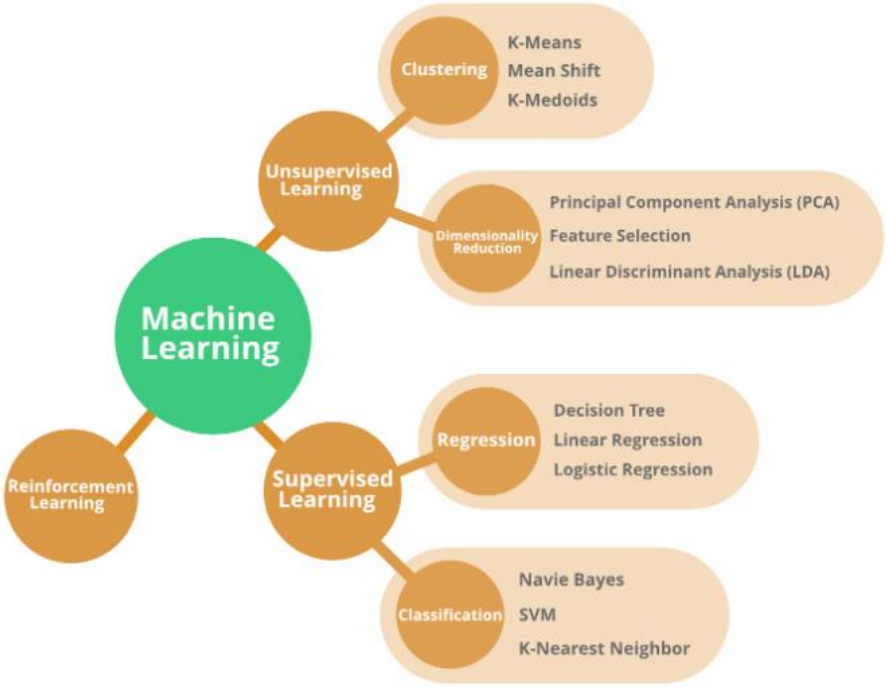

Figure 1. Structure of machine learning.

2.2. Unsupervised Learning

Unsupervised learning is a facet of machine learning where the algorithm is presented with datasets lacking explicit labels or predefined target outputs. Differing from supervised learning, which learns from labeled instances, unsupervised learning centers on unveiling patterns, relationships, and structures inherent in the data [13]. The central objective of unsupervised learning is to identify intrinsic structures within the data, which can involve tasks such as clustering, dimensional reduction, and anomaly detection. Since there are no provided labels, the algorithm must identify patterns and groupings based solely on the intrinsic properties of the data. Unsupervised learning prepares the input data for analysis by handling missing values, scaling or normalizing features, and performing any other necessary transformations to make sure the data is feasible for the chosen unsupervised learning task. Appropriate unsupervised learning technique based on the type of insights you want to extract from the data should be chosen.

Different techniques are suitable for different goals, such as clustering, dimensional reduction, or density estimation. By running the selected unsupervised learning algorithm on the preprocessed data, the algorithm strives to uncover patterns, relationships, or structures within the data without relying on labeled outputs. After interpreting the patterns, insights gained from the analysis can be applied to real-world problems. Unsupervised learning is a powerful approach for exploring and uncovering hidden patterns within data, and it can be especially useful when you're working with unstructured or complex datasets where explicit labels are not available.

2.3. Ensemble Learning

Ensemble learning is a method within machine learning that merges the predictions of numerous individual models to formulate a more potent, precise, and resilient predictive model [14]. The

concept underlying ensemble learning is that through amalgamating predictions from several models, the strengths of individual models can offset the deficiencies of others, resulting in enhanced performance and better generalization. Ensemble learning methods are particularly useful when dealing with complex and challenging problems, as well as when the datasets is noisy or limited. For getting accurate prediction model, a set of base models (also called weak learners or base learners) that will form the ensemble will be chosen. These base models can be of the same type or different types, as long as they produce diverse predictions.

After identifying the optimal hyper-parameters, retrain the base models on the complete training datasets to construct the final ensemble model. Finally, the ensemble model will be deployed to generate predictions on novel and unseen data by aggregating the predictions from its base models. It's important to note that the success of ensemble learning depends on the diversity and quality of the base models. If the base models are too similar, the ensemble might not perform well. In addition, while ensemble methods can improve performance, they can be computationally expensive, so their use should be considered in relation to the problem's complexity and available resources.

3. Application of machine learning algorithm on signal compensation

Machine learning has had a significant impact on signal compensation across various domains. Signal compensation aims to mitigate signal distortions, noise, or interference to enhance the quality and reliability of signals. It can allow for adaptive compensation strategies. Models can learn from real-time or dynamic data, adjusting compensation parameters based on changing signal conditions. This adaptability is particularly useful in scenarios where signal characteristics vary over time. Machine learning algorithms can also learn complex patterns from large datasets, enabling more accurate compensation for signal distortions. Traditional compensation methods may struggle with intricate non-linear relationships, but machine learning models like decision trees, K-means, and boosting can capture such complexities.

3.1. Decision Tree

A decision tree stands as a widely employed supervised machine learning algorithm catering to both classification and regression assignments. It is a model that represents decisions and their possible consequences in a structured manner. The algorithm operates through iterative segmentation of input data into subsets according to various attribute values, ultimately culminating in the forecast of a target variable or class label. The arrangement of a decision tree comprises nodes and edges, symbolizing choices and their outcomes, a sketch is shown in Fig. 2 [14].

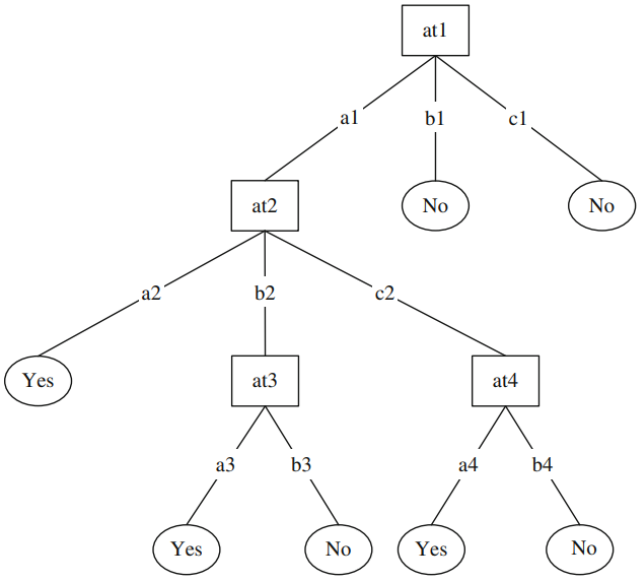

Figure 2. Structure of decision tree.

The root node is a fundamental component of a decision tree and holds a central role in guiding the entire tree's structure and decision-making process. The initial node, referred to as the root node, marks the commencement of the decision tree. It represents the initial decision or test that divides the entire datasets into two or more subsets, each of which will be further split as the tree expands. At the first node, a specific attribute is chosen for the first test. Based on the attribute's values and the associated test condition, the datasets are partitioned into subsets. Each subset corresponds to a different branch originating from the root node. The attribute selected at the root node is typically the one that provides the best separation between different classes or the highest reduction in variance for regression problems. This attribute choice aims to create a strong initial partition that can efficiently lead to classifying or predicting the target variable. The decision made at the root node has a significant impact on the structure of the entire decision tree. It lays the foundation for subsequent decisions made at internal nodes, which further refine the classification or prediction process. The depth of the root node is the shortest path from the root to a terminal leaf node. It represents the number of attribute tests required to make a prediction based on the tree's structure. The depth of the root node is often used as a measure of the decision tree's complexity. While the root node itself doesn't directly provide predictions or class labels, it determines the path that data will take through the tree. As the data traverses the tree, it encounters more attribute tests at internal nodes, ultimately leading to a prediction at a leaf node. The root node's significance lies in its ability to establish the beginning segmentation of the data and set the tone for subsequent decisions throughout the tree. Its attribute choice and test condition are pivotal in creating a decision tree that effectively captures patterns and provides accurate predictions or classifications for new data. Edges define the logical connections between nodes in a decision tree, guiding the decision-making process and leading to the final predictions or classifications at the leaf nodes. Their arrangement forms the distinctive branching structure that makes decision trees an intuitive and interpretable model for machine learning tasks.

The leaf node is a critical component of a decision tree, representing the end points of the tree's branches. A leaf node is the ultimate destination for data as it traverses through the decision tree. It signifies the conclusion of the decision-making process for a specific path through the tree. The context of classification tasks, a leaf node holds a predicted class label. This label corresponds to the class that the majority of training samples reaching that leaf node belong to. For regression tasks, the leaf node contains a predicted continuous value, typically calculated as the average of the target values of training samples in that leaf. Ideally, the samples within a leaf node are as homogeneous as possible in terms of the target variable or class label. This ensures that the prediction made at the leaf node is more accurate and reliable. The leaf node is often referred to as a terminal node because it doesn't have any further branching or decisions associated with it. It signifies the endpoint of a specific path in the decision tree. In classification, some decision tree algorithms provide probabilities associated with class predictions. A threshold can be set for the probability to decide which class label is assigned if the probabilities are close. Some decision tree variants, like weighted decision trees, consider the weight of training samples reaching a leaf node. This weight can affect how predictions are determined in the final outcome. The predictions made at leaf nodes contribute to the overall interpret ability of the decision tree model. It allows for clear understanding of which class or value is assigned under specific conditions. The leaf node is a fundamental endpoint in a decision tree that ultimately provides the predictions or outcomes of the model's decision process. It encapsulates the collective characteristics of the training data that has followed a particular path through the tree and serves as the basis for the final classification or regression results.

Edges also play an important element in connecting nodes and defining the structure of a decision tree. Edges are the links that connect nodes within the decision tree. They establish the logical flow of decisions, guiding the path that data takes as it moves through the tree. Each edge leaving an internal node corresponds to a specific outcome of the attribute test performed at that node. It represents the result of the decision made at the internal node. The outcome of an attribute test determines which branch the data will follow. The value of the tested attribute determines the specific edge the data will traverse along. Edges essentially convey the decision-making logic of the tree. By

following the edges based on the attribute test outcomes, the tree determines the class label or prediction for a given input data point. The hierarchical arrangement of nodes connected by edges forms a branching structure, resembling a tree. This hierarchy is central to the tree's ability to capture complex patterns in data. When a decision tree is graphically represented, edges are depicted as lines connecting nodes. This visual representation helps in understanding the logical flow of decisions and the paths taken by data. The length of an edge is not directly related to the complexity of the decision it represents. Some decisions might involve simple tests, leading to short edges, while others might involve more complex conditions and longer edges.

An application of decision tree on signal compensation is reducing time delay compensation. In wireless communication systems, signals can experience variable time delays due to factors such as signal propagation through different mediums or reflections. These delays can introduce errors and degrade the quality of received signals. Decision trees can be utilized to predict the time delay associated with received signals and subsequently apply compensation techniques. Relevant features that can help predict the time delay should be identified. These features could include signal strength, frequency, phase, modulation type, and any other characteristics of the transmitted and received signals. The known time delay values for each signal can be used as labels for training the decision tree. After training, the decision tree will learn the relationship between the signal features and the corresponding time delays. It can be also used to predict the time delay for newly received signals. The features of the received signal are input to the decision tree, and it provides a prediction of the time delay. The predicted time delay can be used to synchronize the received signal with the transmitted signal. Compensation techniques, such as adjusting the received signal's timing, can be applied to align the signals correctly. Finally, the effectiveness of the time delay compensation should be validated by assessing the accuracy of signal alignment and the reduction in errors. Decision trees provide insights into the factors that contribute to predicting time delays. This can aid in understanding the underlying causes of signal delays. Decision trees can also handle both categorical and numerical data, making them suitable for various signal characteristics. non-linear relationships between signal features and time delays can be captured by decision trees, which is essential in complex signal propagation scenarios. By utilizing decision trees for time delay compensation in wireless communication, it's possible to mitigate the adverse effects of signal delays, leading to improved data transmission quality and enhanced communication performance.

3.2. K-means

K-means, an algorithm for clustering within unsupervised machine learning, is utilized to divide a provided dataset into "K" clusters, relying on resemblances among data points. The algorithm seeks to identify cluster centers (centroids) that minimize the total of squared distances between data points and their corresponding cluster centers. A sketch is shown in Fig. 3 [15]. Every data point is allocated to the cluster with the closest centroid. Determining the suitable value of "K" for the K-means algorithm, often referred to as the number of clusters, is a crucial step. There are several techniques and methods can determine an optimal value for "K." Here are some common approaches. The elbow technique entails graphing the within-cluster sum of squares (WCSS) against varying "K" values. WCSS quantifies the squared distance between each data point and the centroid of the cluster it belongs to. With an increase in "K," there is a tendency for WCSS to decline, as each point moves closer to the centroid of its respective cluster. However, except a certain "K", the reduction in WCSS becomes less significant, forming an "elbow" point on the graph. This point is often considered a good choice for K [16]. Silhouette analysis assesses the degree of similarity between an object and its own cluster (cohesion) in contrast to other clusters (separation). A silhouette score spans from -1 to 1, with higher values suggesting effective alignment of data points with their designate clusters and weaker alignment with neighboring clusters. One can calculate silhouette scores for different "K" values and choose the one with the highest score [17]. The gap statistic compares the performance of the K-means clustering on the given datasets with its performance on randomly generated data. It helps identify if the observed clustering is better than what would be expected by chance. Larger gap

values indicate better clustering. Compare the gap statistic for different "K" values to select the optimal one [18]. The Davies-Bouldin index gauges the mean similarity between every cluster and its most akin cluster. A diminished index signifies improved differentiation among clusters. Then, one can calculate this index for different "K" values and choose the one with the lowest value [19]. One can use cross-validation to evaluate the performance of K-means for different "K" values. Measure the quality of clustering using metrics like WCSS, silhouette score, or others. One can choose the "K" that provides the best trade-off between clustering quality and model complexity [20].

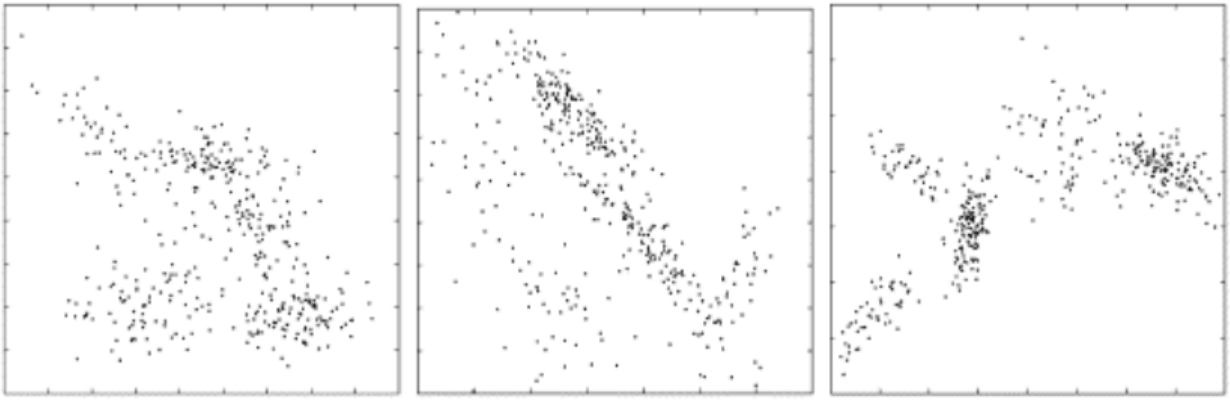

Figure 3. Some examples of k-means.

It is useful method for time compensation. There are datasets of timestamps from various devices or sensors. Due to differences in clock accuracy, there might be variations in the timestamps, leading to time discrepancies. The goal of K-means is to compensate for the time discrepancies in the timestamps and align them as accurately as possible. K-means clustering can be used to identify distinct clusters of timestamps that exhibit similar time discrepancies. Each cluster represents a group of devices or sensors with similar timing characteristics. Timestamps should be collected from different devices or sensors. Each timestamp should represent an observation. The time discrepancies will be calculated between each timestamp and a reference time. These discrepancies will be used as the features for clustering. The K-means clustering algorithm can be employed to categorize timestamps according to their temporal disparities. The number of clusters "K" will be chosen based on domain knowledge or techniques like the elbow method. After clustering, each timestamp should be assigned to its corresponding cluster. Calculated time offset will be applied to the timestamps within each cluster to compensate for the time discrepancies. Finally, the quality of time compensation should be assessed by measuring the alignment of timestamps across devices or sensors. K-means finds extensive application in various contexts, including image segmentation, customer segmentation, data compression, and numerous other scenarios. It's essential that the choice of the number "K" (the desired number of clusters) can impact the results. Selecting an appropriate value for "K" is often determined using techniques, e.g., the elbow method or silhouette analysis.

3.3. Boosting

Boosting is an ensemble machine learning technique with the objective of enhancing the effectiveness of feeble learners through a structured amalgamation of their predictions. Unlike bagging (Bootstrap Aggregating), where multiple independent models are trained in parallel and their predictions are averaged, boosting builds a sequence of models iteratively, where each new model focuses on correcting the mistakes made by the previous ones. The central idea of boosting is to give more weight to the examples that were misclassified by the previous models, thereby forcing the new model to focus on the difficult cases. As the boosting process continues, the new models pay increasing attention to the instances that the ensemble as a whole find challenging [21].

Each data point is given equal weight at the beginning. A weak learner is trained on the data with the initial weights. After training, the weights of inaccurately classified instances are elevated. This emphasizes the instances that the current model struggled with. A new weak learner is trained on the

data with updated weights. This new model focuses on the misclassified examples from the previous iterations. The predictions of the new model are combined with the predictions of the previous models using weighted averaging for regression tasks. These actions are reiterated for a predetermined count of iterations or until a cessation criterion is fulfilled. The ultimate forecast is derived from the compilation of predictions made by all the feeble learners, usually with each learner's contribution weighted based on its performance. Therefore, Boosting is a powerful technique that often outperforms individual models and other ensemble methods in a wide range of machine learning tasks. However, it's important to monitor for overfitting, as boosting can lead to memorizing the training data if not properly controlled.

Boosting can be applied to signal compensation problems, particularly when dealing with noisy or distorted signals. In communication systems, signals can be corrupted by noise during transmission, which can lead to errors in data reception. The goal of boosting is to compensate for this noise and recover the original signal, and the challenge is to accurately identify and mitigate the effects of noise in the received signal to improve data recovery and minimize errors. Boosting can be employed to create an ensemble of models that work together to compensate for the noise in the received signal. A simple model should be chosen as the weak learner. This could be a decision stump (a decision tree with only one level) or another model with low complexity. Multiple instances of the chosen weak learner should be trained sequentially. Each iteration focuses on predicting the difference between the original and received signals. In each iteration, calculate the differences between the original and received signals. Train the weak learner to predict these residuals. Samples with larger residuals will be assigned higher weight. This emphasizes the samples that have been more affected by noise in the training process. After combing the predictions of all weak learners in the ensemble, the boosted ensemble model will predict the residuals, which can be added back to the received signal to compensate for the noise and recover the original signal. Boosting focuses on samples that are more affected by noise, allowing the ensemble to prioritize the correction of challenging cases. By combining multiple weak learners, boosting can effectively capture complex relationships between the noisy and original signals. Each iteration of boosting contributes to gradually improving the signal compensation by addressing different aspects of noise. The ensemble nature of boosting helps in reducing the impact of outliers or extreme noise cases. Boosting can incorporate various weak learners, making it adaptable to different signal compensation scenarios. Applying boosting to signal compensation enables the creation of an ensemble model that collaboratively addresses the challenges posed by noise in communication systems. It improves the accuracy of signal recovery and enhances the overall reliability of data transmission.

4. Conclusion

To sum up, machine learning has revolutionized signal compensation by providing more accurate, adaptable, and automated solutions. It enables compensation strategies that were difficult to achieve using traditional methods, contributing to enhanced signal quality and reliability across various applications. It can optimize signal compensation for energy efficiency. Algorithms can balance signal quality and power consumption, adjusting compensation parameters to achieve the desired trade-off. Machine learning models can automatically identify relevant features in signals and enhance them while suppressing noise. This is particularly beneficial for tasks like image or audio signal processing. In scenarios with multiple sources of signal distortion or interference, machine learning can model these complexities and provide effective compensation solutions. In summary, the future of machine learning is likely to bring about advancements that improve accuracy, transparency, ethics, and applications across diverse industries, making AI a more integral part of our daily lives. However, addressing challenges related to data privacy, bias, and ethical considerations will be essential for the responsible development and deployment of machine learning technologies.

References

[1] Mahesh B. Machine learning algorithms-a review. International Journal of Science and Research (IJSR), 2020, 9 (1): 381 - 386.

[2] Schmitz G P J, Aldrich C, Gouws F S. Ann-DT: An algorithm for extraction of decision trees from Artificial Neural Networks. IEEE Transactions on Neural Networks, 1999, 10 (6): 1392 – 1401.

[3] Shih A, Choi A, Darwiche A. A symbolic approach to explaining bayesian network classifiers. arXiv preprint arXiv:1805.03364, 2018.

[4] Chou W, Juang B H. Pattern recognition in speech and Language Processing. CRC Press, 2003.

[5] Navarro C A, Hitschfeld-Kahler N, Mateu L. A survey on parallel computing and its applications in data-parallel problems using GPU architectures. Communications in Computational Physics, 2014, 15 (2), 285 – 329.

[6] Babu R G, Nedumaran A, Manikandan G, Selvameena R. Tensorflow: Machine learning using heterogeneous edge on distributed systems. Deep Learning in Visual Computing and Signal Processing, 2022: 71 – 90.

[7] Mallouh A, Qawaqneh Z, Barkana B D. Utilizing CNNS and transfer learning of pre-trained models for age range classification from unconstrained face images. Image and Vision Computing, 2019, 88: 41 – 51.

[8] Wiering M, Otterlo M. Reinforcement learning state-of-the-art. Springer Berlin, 2014.

[9] Gunasekaran A. Generative Adversarial Networks: A Brief History and Overview, 2022.

[10] Häger C, Pfister H D, Bütler R M, Liga G, Alvarado A. Model-based machine learning for Joint Digital Backpropagation and PMD Compensation. Optical Fiber Communication Conference (OFC) 2020: w3d.

[11] Liu B. Supervised Learning. In: Web Data Mining. Data-Centric Systems and Applications. Springer, Berlin, Heidelberg, 2011.

[12] Bousquet Q, von Luxburg U, Rätsch G. (Advanced Lectures on Machine Learning. Springer, Berlin, Heidelberg, 2004.

[13] Arbib M A. The Handbook of Brain Theory and Neural Networks. MIT Press, 2003.

[14] Kotsiantis S B. Decision trees: a recent overview. Artif Intell Rev, 2013, 39, 261 – 283.

[15] Aristidis L, Nikos V, Jakob J V. The global k-means clustering algorithm, Pattern Recognition, 2003, 36 (2): 451 - 461.

[16] Bholowalia P, Kumar A. EBK-means: A clustering technique based on elbow method and K-means in WSN. EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN, 2014.

[17] Lletí R, Ortiz M C, Sarabia L A, Sánchez M S. Selecting variables for k-means cluster analysis by using a genetic algorithm that optimises the silhouettes. Analytica Chimica Acta, 2004, 515 (1): 87-100.

[18] El-Mandouh A M, Abd-Elmegid L A, Mahmoud H A, et al. Optimized K-means clustering model based on gap statistic. International Journal of Advanced Computer Science and Applications, 2019, 10 (1).

[19] Jumadi Dehotman Sitompul B, Salim Sitompul O, Sihombing P. Enhancement clustering evaluation result of davies-bouldin index with determining initial centroid of k-means algorithm. Journal of Physics: Conference Series. IOP Publishing, 2019, 1235 (1): 012015.

[20] Allayear S M, Sarker K, Ara S J F. Prediction model for prevalence of type-2 diabetes complications with ANN approach combining with K-fold cross validation and K-means clustering. Advances in Information and Communication Networks: Proceedings of the 2018 Future of Information and Communication Conference (FICC). 2018, 886: 451.

[21] Schapire R E. The boosting approach to machine learning: An overview. Nonlinear estimation and classification, 2003: 149 - 171.