**Limitations of HMMs**

1. Limited Modeling Capabilities

- **First-Order Assumption** : HMMs assume that the probability of transitioning to the next state depends only on the current state. This is unrealistic for complex systems where future states depend on multiple previous states.

- **Discrete State Space** : traditional HMMs use a discrete set of hidden states, which may not be sufficient for modeling phenomena with continuous state spaces.

- **Stationarity Assumption** : HMMs assume that the transition and emission probabilities do not change over time while many real-world processes are non-stationary, meaning their statistical properties change over time.

2. Overfitting

- **Large Number of Hidden States :** Each state corresponds to a different aspect of the phenomenon being modeled (e.g., gestures in gesture recognition). If there are too many states, the model may overfit the training data by capturing noise or specific variations that are not relevant.

- **Limited Training Data :** HMMs require sufficient data to learn the underlying transition probabilities and emission probabilities accurately. With insufficient data, the model may fit the noise rather than the true underlying process.

3. Lack of Robustness

HMMs are limited in their robustness to noise and variability in the data. For example, in speech recognition, the acoustic signals generated by speech can be subjected to a variety of distortions and noise, which can make it difficult for the HMM to accurately estimate the underlying structure of the data. In some cases, these distortions and noise can cause the HMM to make incorrect decisions, which can result in poor performance.

4. Computaional Complexity

- **Parameter Estimation :** HMMs require estimating model parameters, such as transition probabilities and emission probabilities, from the training data. This estimation process can be time-consuming and computationally expensive, especially for large models or high-frequency data.

- **Likelihood Computation :** To evaluate the likelihood of observed data given the model, we need to compute the forward probabilities. This involves summing over all possible state sequences, which can be computationally intensive.

**Unsupervised Machine Learning**

**What is Unsupervised Learning?**

As the name suggests, unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things.

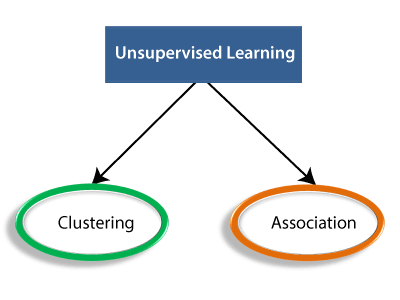
Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to **find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format**.

**Example:** Suppose the unsupervised learning algorithm is given an input dataset containing images of different types of cats and dogs. The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own. Unsupervised learning algorithm will perform this task by clustering the image dataset into the groups according to similarities between images.

**Why use Unsupervised Learning?**

* Unsupervised learning is helpful for finding useful insights from the data.
* Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.
* Unsupervised learning works on unlabeled and uncategorized data which make unsupervised learning more important.
* In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.

**Types of Unsupervised Learning Algorithm:**



* **Clustering**: Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.
* **Association**: An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is Market Basket Analysis.

**Working of Unsupervised Learning**



Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabeled input data is fed to the machine learning model in order to train it. Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as k-means clustering, Decision tree, etc.

Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and difference between the objects.

**Advantages of Unsupervised Learning**

* Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.
* Unsupervised learning is preferable as it is easy to get unlabeled data in comparison to labeled data.

**Disadvantages of Unsupervised Learning**

* Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
* The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.

**Federated Learning**

**What is federated learning?**

Federated learning (often referred to as collaborative learning) is a decentralized approach to training machine learning models. It doesn’t require an exchange of data from client devices to global servers. Instead, the raw data on edge devices is used to train the model locally, increasing data privacy. The final model is formed in a shared manner by aggregating the local updates.

*Here’s why FL is important*.

**Privacy**: In contrast to traditional methods where data is sent to a central server for training, federated learning allows for training to occur locally on the edge device, preventing potential data breaches.

**Data security**: Only the encrypted model updates are shared with the central server, assuring data security.

**Access to heterogeneous data**: FL guarantees access to data spread across multiple devices, locations, and organizations. It makes it possible to train models on sensitive data, such as financial or healthcare data while maintaining security and privacy. And thanks to greater data diversity, models can be made more generalizable.

**How does federated learning work?**

A generic baseline model is stored at the central server. The copies of this model are shared with the client devices, which then train the models based on the local data they generate. Over time, the models on individual devices become personalized and provide a better user experience.

In the next stage, the updates (model parameters) from the locally trained models are shared with the main model located at the central server using secure aggregation techniques. This model combines and averages different inputs to generate new learnings. Since the data is collected from diverse sources, there is greater scope for the model to become generalizable.

Once the central model has been re-trained on new parameters, it’s shared with the client devices again for the next iteration. With every cycle, the models gather a varied amount of information and improve further without creating privacy breaches.

**Types of Federated Learning**

* **Centralized Federated Learning**

Centralized federated learning requires a central server. It coordinates the selection of client devices in the beginning and gathers the model updates during training. The communication happens only between the central server and individual edge devices.

While this approach looks straightforward and generates accurate models, the central server poses a bottleneck problem : network failures can halt the complete process.

A diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of

Description automatically generated

* **Decentralized federated learning**

Decentralized federated learning does not require a central server to coordinate the learning. Instead, the model updates are shared only among the interconnected edge devices. The final model is obtained on an edge device by aggregating the local updates of the connected edge devices.

This approach prevents the possibility of a single-point failure; however, the model's accuracy is completely dependent on the network topology of the edge devices.

A diagram of a cell phone network

Description automatically generated

**Challenges and limitations of federated learning**

* **Communication efficiency**

Federated learning involves millions of devices in one network. The transfer of messages becomes slow due to several reasons: low bandwidth, lack of resources, or geographical location.

To keep the communication channels efficient, the total number of message passes and the size of a message in a single pass should be reduced. We can achieved it by using:

* local updating methods (to reduce the number of rounds)
* model compression schemes (to reduce the size of the message)
* decentralized training (to operate in low bandwidth)
* **Heterogeneity of Clients**

Clients in federated learning can have varying levels of computational power, memory, and storage. Some devices might be high-end smartphones, while others could be older, less powerful devices. This disparity affects the clients' ability to perform local computations and participate in the training process.

* **Non-Independent and Identically Distributed (Non-IID) Data**

Each client might have data that are biased or have different distributions. For example, some devices may have high-resolution image data, while others can only store low-resolution pictures, or languages might vary based on geographical location. This diversity makes it challenging to create a global model that performs well for all clients.

**Hierarchical Federated Learning in 5G Networks**

Devices or clients within a close proximity, such as those connected to the same edge server or base station, first perform local model aggregation. This reduces the communication overhead by summarizing local updates before sending them to a higher-level server.

Local aggregates are then sent to regional servers (edge or fog nodes) where further aggregation occurs. This hierarchy can continue through multiple levels (e.g., from regional to central servers).

**Benefits of Hierarchical Federated Learning in 5G Networks**

* **Bandwidth Optimization** : By aggregating updates at multiple levels, HFL reduces the amount of data transmitted over long distances, optimizing the use of 5G bandwidth
* **Scalability**: HFL supports a larger number of devices by distributing the aggregation workload across multiple hierarchical levels.