ANN UPDATED NOTES 27-12-24.pdf

by Head CSE

Submission date: 27-Dec-2024 10:05AM (UTC+0530)
Submission ID: 2558317218
File name: ANN_UPDATED_NOTES_27-12-24.pdf (13.9M)
Word count: 4257
Character count: 24516

Module 1

The Basics of Neural Learning

Introduction

ANNs are set up for a particular application, like pattern recognition or data classification, through a learning process. Learning in biological systems involves adjusting the synaptic connections that exist between the neurons.

Neural Computing is an information processing paradigm inspired by biological systems comprised of a large number of highly inter connected processing elements (neurons) working in unison to solve specific problems.

ANN Definition

A network of inter connected linked processing units known as nodes or neurons is referred to as a neural network or neural net.

Artificial Neural Networks (ANN) or Neural Networks are the cornerstone of deep learning and an essential component of Artificial Intelligence. An ANN or Neural Network is a computational architecture made up of nodes or neurons that mathematically simulates how a biological neural network finds and recognizes correlations in data.

Neural networks are essentially non-linear models for Machine Learning that container be applied to both supervised and unsupervised learning. Neural networks can also be thought of as a collection of algorithms designed to find patterns and are loosely based on the structure of the humanlike brain.

The ANN Basic Concept

A computer system called an ANN or Neural Network is made to mimic the way the humanlike brain interprets and processes data. It is the cornerstone of AI and resolves issues that by statistical or human standards would be impossible or challenging.

The main purpose of ANN is to replicate and model how the human brain operates. An ANN is built to simulate biological neurons using anatomical structure.

The way the human brain makes decisions is by observing or absorbing information through the five senses, storing it, comparing it to prior knowledge, and then making decisions based on that information.

The idea behind ANN is similar to that of a natural neural network. Making robots or systems comprehend and simulate how the human brain makes decisions and then acts on them is the aim of ANN's.

Neural Net

A Neural Net is a synthetical model of the human brain that attempts to simulate how it learns. Artificial Neural Networks (ANNs) are sometimes referred to as "Neural Networks" or simply Neural Nets (NNs). Traditionally, the term "Neuron Network" refers to a network of biological neurons in the nervous system that process and transmit information. An Artificial Neural Network is a group of interconnected Artificial Neurons that use a scientific discipline model or computational model for information processing based on a connection list approach to computation. The Artificial Neural Networks are composed of interconnecting artificial neurons that may share some properties with biological neural networks.

Neural Networks are a type of multiprocessor computer system that has simple processing elements, a high degree of inter connection, simple scalar messages, and adaptive interaction between elements. Neural Networks are based on the parallel architecture of biological brains, while machines are based on the processing or memory abstraction of human information processing.

History

Most people agree that McCulloch and Pitts (1943) created the first ANN. They put a lot of simplex processing units together, which could boost computing power overall. They made numerous suggestions, such as the hypothesis that a neuron has a threshold level and fires when it reaches it. It remains the core mechanism by which ANNs function. HEBB (1949) established the first ANN learning rule, which states that if two nerve cells are progressive simultaneously, their strength should be raised. The weights in McCulloch and Pitts' network were fixed.

In 1985–86, Neural Networks gained popularity again. Parker and LeCun, the researchers, found that back propagation, a learning approach for multi-layer networks, could understand the problems that were not linearly divisible.

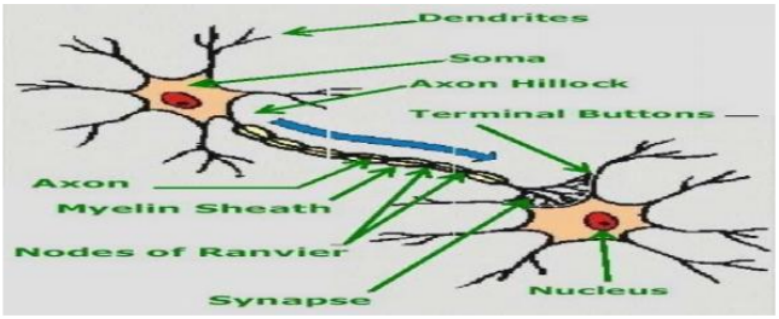


Fig: ANN MODEL

There are many Neuronic cells in the human brain more than a million that process information. Every cell functions similarly to a basic processor. Nerve fiber are branching fibers that extend to from the cell structure or Soma. The brain's capabilities are only made possible by the vast fundamental interaction between all nerve cells and their comparable processing.

An axone is a single nerve fiber that emballement message from the soma to the conjunction sites of other neurons (dendrites and somas), muscles, or glands. The soma, also known as the nerve cell body of a neuron, houses the nucleus and other structures, supports natural science processing, and produces neuro transmitters.

The axone hillock is where incoming information is summarized. Anaxon hillock and propagation down the axon are determined at any one time by the combined action of all neurons that conduct impulse to a particular neuron.

The fat-containing cells that make up the myelin sheath protect the axon from electrical activity. Signal transmission speed is accelerated by this insulation. Along the axon, there is a space between every myeline sheath cell. Since embonpoint acts as a good nonconductor, the myelin cover speed up the rate of transmission of an electric impulse along the axone.

Conjugation is the point of connection between two neuron cells or a neuron cell and a musculus or gland. Nodes of Ranvier are the gaps (roughly 1 m) between myelin sheath cells long axons Because fat inhibits the propagation of electricity, signs jump from one gap to the next.

These connections allow neurons to communicate electrochemically. A neuron's time period switch are the small convex shape at the end of an axone that release chemical substance known as neuro communicators.

Neural cell Data Flow

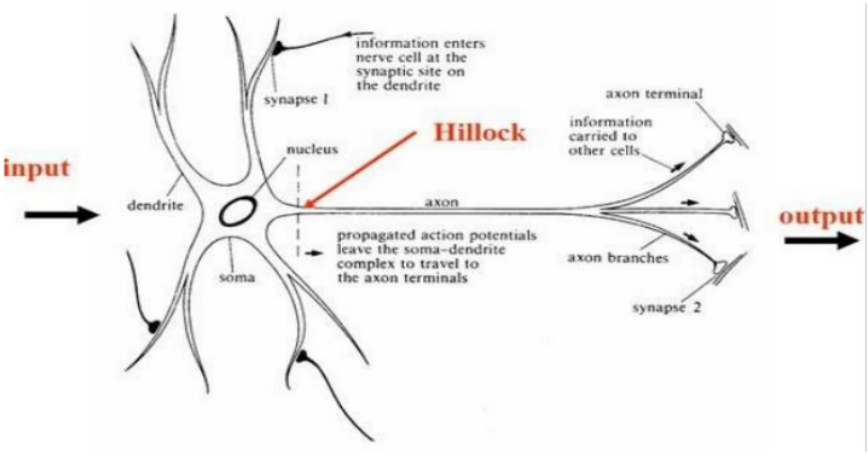


Fig: Neural cell Information Flow

Nerve fiber receive activation from other neurons Incoming activations are processed by Soma, which then transforms them into output activations. In order to activate other neurons, axons function as transmission lines. Signal transmission between axons and dendrites is made possible via synapses, which are junctions. Transmission occurs through the diffusion of substances known as neuro-transmitters.

Neural Networks Key components

a. Nodes or Neurons

The fundamental structure blocks of a neural network are neurons.

After processing inputs, each neuron applies weights and biases, computes, and then runs the result via an activation function.

b. Layers

Three different determination of layers make up the neural networks.

Input Layer: Raw data is received by the input signal layer, which past forwards it to the following layer.

Hidden Layers: Use weights, biases, and activation functions to carry out calculations and feature modifications.

Output Layer: The output layer generates the end product, such as the regression value or categorization.

c. Biases and Weights

Weights: Indicate how strongly neurons are connected to one another.

Biases: To improve learning let the model change the activation function.

d. Activation-Function

Mathematical function that initiate non linearity into the model allowing it to solve analyzable problems.

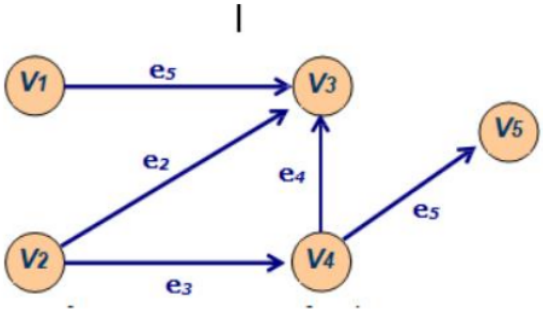
Individual types Use the **Sigmoid** for probabilities: $\sigma(x)=1/1+e-x1$.

Deep learning frequently uses **ReLU**: $f(x)=\max(0,x)$.

Tanh: The formula **tanh** (x)= ex+e+x/ex-e-x return belief ranging from -1 to 1.

Softmax: For classification issues involving several classes.

Neural Network Architectures



An oriented graph G, an ordered 2-tuple (V, E) with a set V of Vertices and a set E of Edges, can be used to represent an Artificial Neural Network (ANN), a data processing system made up of many simple, highly connected processing elements, such as artificial neurons, in a network structure.

The input or output Neurons may be represented by the Vertices cells & Edge Cells.

V = { x1, x2, x3, x4, x5}

E={y1,y2,y3,y4,y5}

MultiLayer Feed forwardNetwork

A single layer of weights makes up the Single Layer Feed forward Network, in which a sequence of weights connects the inputs and outputs directly. All inputs and outputs are connected by synaptic linkages that carry weights, but not the other way around.

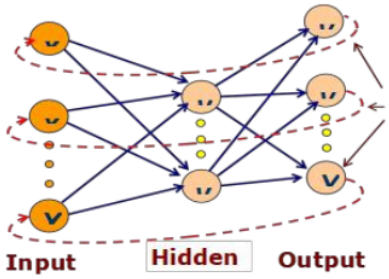
In this sense, it is seen as a feed forward network. Each neuron node computes the sum of the products of the weights and the inputs if the value is greater than a certain threshold (usually 0) the neuron fires and takes the active value (usually 1) if not it takes the deactivated value (usually -1).

Feed forward Multilayer Network

As the name refers, it has several levels. This type of networks architecture includes one or more intermediate levels known as hidden layers in addition to the input and output layers. Hidden neurons are the hidden layers computational units.

Recurrent Networks

Recurrent Networks have at least one feed-back loop, which sets them apart from feed forward design.

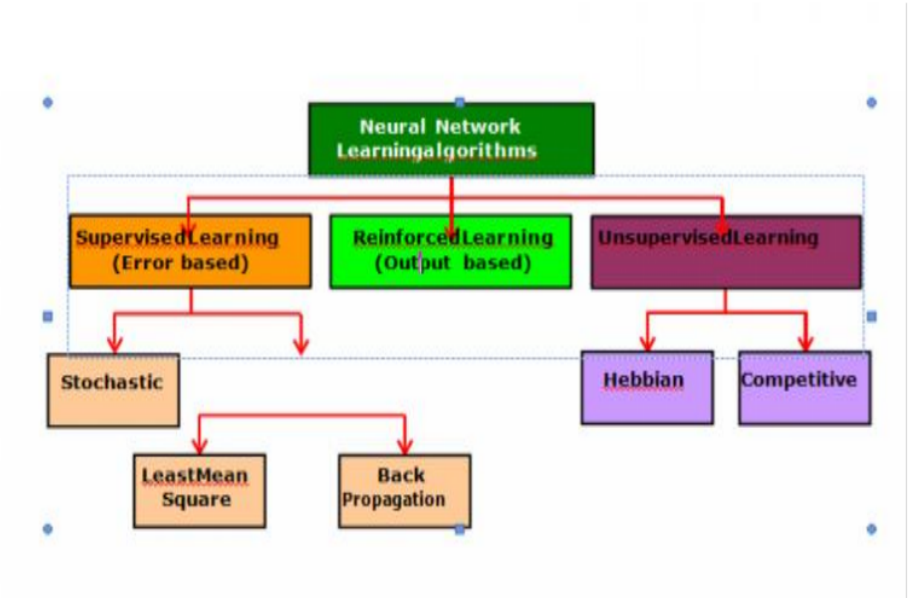


Neurons of Input Layer = Ii

Neurons of Hidden Layer = Hj

Neurons of Output Layer = Ok

Classification of ANN



Supervised Learning

In this approach, the algorithm learns from labeled data, where the training data includes input-output pairs. The model learns to map inputs to outputs, making predictions or classifications based on new, unseen data.

Unsupervised Learning

Here, the algorithm learns from unlabeled data to find hidden patterns or intrinsic structures within the data. Clustering and dimensionality reduction are common tasks in unsupervised learning.

Reinforcement Learning

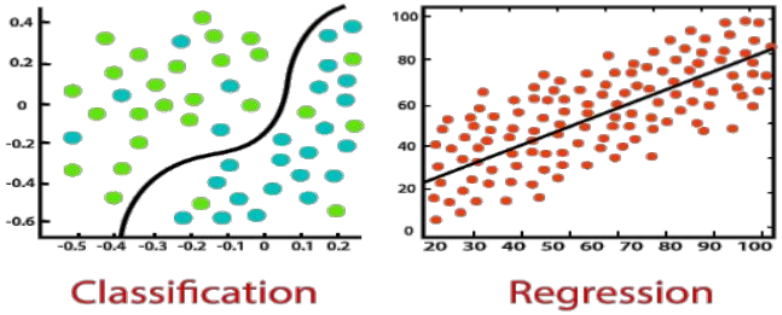
This involves training agents to make sequences of decisions by rewarding or punishing them based on their actions.

Reinforcement learning is often used in scenarios where the model interacts with an environment and learns optimal strategies through trial and error.

Regression vs Classification in Machine Learning

Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labeled datasets. But the difference between both is how they are used for different machine learning problems.

The main difference between Regression and Classification algorithms that Regression algorithms are used to predict the continuous values such as price, salary, age, etc. and Classification algorithms are used to predict or Classify the discrete values such as Male or Female, True or False, Spam or Not Spam, etc.



Classification

Classification is a process of finding a function which helps in dividing the dataset into classes based on different parameters. In Classification, a computer program is trained on the training dataset and based on that training, it categorizes the data into different classes.

Types of Machine learning Classification Algorithms

Classification Algorithms can be further divided into the following types:

1. Logistic Regression
2. K-Nearest Neighbours
3. Support Vector Machines
4. Kernel SVM
5. Naive Bayes
6. Decision Tree Classification
7. Random Forest Classification

Regression

Regression is a process of finding the correlations between dependent and independent variables. It helps in predicting the continuous variables such as prediction of Market Trends, prediction of House prices, etc.

The task of the Regression algorithm is to find the mapping function to map the input variable(x) to the continuous output variable(y).

Example: Suppose we want to do weather forecasting, so for this, we will use the Regression algorithm. In weather prediction, the model is trained on the past data, and once the training is completed, it can easily predict the weather for future days.

Types of Regression Algorithm

1. Simple Linear Regression
2. Multiple Linear Regression
3. Polynomial Regression
4. Support Vector Regression
5. Decision Tree Regression
6. Random Forest Regression

Difference between Regression and Classification

Regression: The output variable must be of continuous nature or real value.

Classification: The output variable must be a discrete value.

The task of the **Regression algorithm** is to map the input value (x) with the continuous output variable(y).

The task of the **Classification algorithm** is to map the input value(x) with the discrete output variable(y).

Regression Algorithms are used with continuous data.

Classification Algorithms are used with discrete data.

Regression: we try to find the best fit line, which can predict the output more accurately.

Classification: we try to find the decision boundary, which can divide the dataset into different classes.

Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc.

Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc.

The **Regression Algorithm** can be further divided into Linear and Non-linear Regression.

The **Classification algorithms** can be divided into Binary Classifier and Multi-class Classifier.

Data Preprocessing

Data Pre processing is the process of cleaning and preparing the raw data to enable feature engineering. After getting large volumes of data from sources like databases, object stores, data lakes, engineers prepare them so data scientists can create features.

This includes basic cleaning, crunching and joining different sets of raw data. In an operational environment this Pre processing would run as an ETL job for batch processing, or it could be part of a streaming processing for live data.

Data Preprocessing

Data Pre processing is the process of cleaning and preparing the raw data to enable feature engineering. After getting large volumes of data from sources like databases, object stores, data lakes, engineers prepare them so data scientists can create features.

This includes basic cleaning, crunching and joining different sets of raw data. In an operational environment this Pre processing would run as an ETL job for batch processing, or it could be part of a streaming processing for live data.

Data Preprocessing Learning Steps

Normalization: Normalization is the process of scaling numeric features to a standard range, typically between 0 and 1. This ensures that all features contribute equally to the model, preventing one dominant feature from overshadowing others.

Encoding: Categorical data, such as gender or country names, needs to be converted into numerical format for machine learning algorithms. Encoding techniques like one-hot encoding or label encoding transform categorical variables into a format that algorithms can understand.

Handling Missing Data: Dealing with missing data is essential for robust model performance. Strategies include removing rows with missing values, imputing missing values with statistical measures, or using advanced techniques like machine learning-based imputation.

Feature Engineering

Feature engineering is the creation of features from raw data.

Feature engineering includes:

1. Determining required features for ML mode.
2. Analysis for understanding statistics, distribution, implementing one hot encoding and imputation, and more. Tools like Python and Python libraries are used.
3. Preparing features for ML model consumption.
4. Building the models.
5. Testing if the features achieve what is needed.

Repeating the preparation and testing process, by running experiments with different features, adding, removing and changing features. During the process, the data scientist might find out data is missing from the sources. The data scientist will request Pre processing again from the data engineer.

Fig: Process of Feature Engineering

ML pipeline Process

Creation of Derived Features:

Feature engineering involves creating new features that enhance the predictive power of the model. For example, extracting the day of the week from a date or creating interaction terms between existing features can provide valuable information.

Dimensionality Reduction: High-dimensional datasets may suffer from the curse of dimensionality, leading to increased computational complexity and potential over fitting. Techniques like Principal Component Analysis (PCA) help reduce dimensionality while preserving essential information.

Handling Outliers: Handling Outliers can misrepresent model training, and addressing them is crucial. Techniques such as trimming, win sorizing, or transforming features can mitigate the effect of outliers on framework execution.

Feature extraction

The feature extraction algorithms transform the data onto a new feature space. When it is important to derive useful information from the data, hence creating new feature subspace doesn't affect the model. Used to improve the predictive performance of the models.

Two categories of Feature extraction

Linear

It assumes that the data falls on a linear subspace or classes of data can be distinguished linearly

Non-linear

It assumes that the pattern of data is more complex and exists on a non-linear sub-manifold.

Unsupervised Feature Extraction

They mostly concentrate on the variation and distribution of data.

PCA:

Linear unsupervised method

The aim of PCA is to find orthogonal directions which represent the data with the least error. PCA tries to maximize this variance to find the most variant ortho normal directions of data. The desired directions are the eigenvectors of the covariance matrix of data.

Kernel Principle Component analysis

KPCA finds the non-linear subspace of data which is useful if the data pattern is not linear. The kernel PCA uses kernel method which maps data to a higher dimensional space . Kernel PCA relies on the blessing of dimensionality by using kernels. i.e., it assumes in higher dimensions, the representation or discrimination of data is easier.

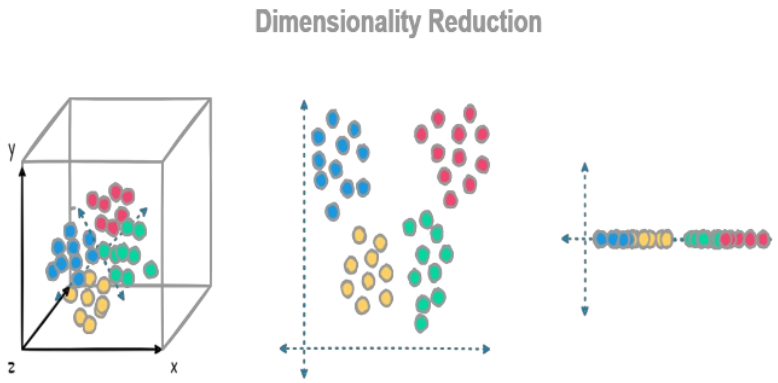
There are so many other unsupervised feature extraction techniques like:

1. Dual Principal Component Analysis
2. Multidimensional Scaling
3. ISO-MAP
4. Locally Linear Embedding
5. Laplacian Eigenmap
6. Maximum variance unfolding
7. Auto encoders and Neural Networks
8. T-distributed stochastic neighbor embedding

Curse of Dimensionality

The Curse of Dimensionality in Machine Learning arises when working with high dimensional data, leading to increased computational complexity, over fitting, and spurious correlations.

Techniques like dimensionality reduction, feature selection, and careful model design are essential for mitigating its effects and improving algorithm performance. Navigating this challenge is crucial for unlocking the potential of high dimensional datasets and ensuring robust machine learning solutions.



The Curse of Dimensionality refers to the phenomenon where the efficiency and effectiveness of algorithms deteriorate as the dimensionality of the data increases exponentially.

In high dimensional spaces, data points become sparse, making it challenging to discern meaningful patterns or relationships due to the vast amount of data required to adequately sample the space.

The Curse of Dimensionality significantly impacts machine learning algorithms in various ways. It leads to increased computational complexity, longer training times, and higher resource requirements. Moreover, it escalates the risk of over fitting and spurious correlations, hindering the algorithms' ability to generalize well to unseen data.

Overcome the Curse of Dimensionality

To overcome the curse of dimensionality, you can consider the following strategies.

Dimensionality Reduction Techniques

Feature Selection: Identify and select the most relevant features from the original dataset while discarding irrelevant or redundant ones. This reduces the dimensionality of the data, simplifying the model and improving its efficiency.

Feature Extraction: Transform the original high-dimensional data into a lower-dimensional space by creating new features that capture the essential information. Techniques such as Principal Component Analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE) are commonly used for feature extraction.

Data Preprocessing

Normalization: Scale the features to a similar range to prevent certain features from dominating others, especially in distance-based algorithms.

Handling Missing Values: Address missing data appropriately through imputation or deletion to ensure robustness in the model training process.

Training the classifiers

Training Before Dimensionality Reduction: Train a Random Forest classifier (clf_before) on the original scaled features (X_train_scaled) without dimensionality reduction.

Evaluation Before Dimensionality Reduction: Make predictions (y_pred_before) on the test set (X_test_scaled) using the classifier trained before dimensionality reduction, and calculate the accuracy (accuracy_before) of the model.

Training After Dimensionality Reduction: Train a new Random Forest classifier (clf_after) on the reduced feature set (X_train_pca) after dimensionality reduction.

Evaluation After Dimensionality Reduction: Make predictions (y_pred_after) on the test set (X_test_pca) using the classifier trained after dimensionality reduction, and calculate the accuracy (accuracy_after) of the model.

Polynomial Curve Fitting

Any pattern recognition or machine learning task can be primarily divided into two categories: Supervised and Unsupervised Learning. In a supervised machine learning problem, we have the input and corresponding desired output. For any supervised learning problem, the aim of the pattern recognition algorithm is to come up with an algorithm or model which can predict the output given the input.

Based on the output, the supervised learning problem can be divided into two categories: Classification (when we have a finite number of discrete output) and Regression (if the desired output consists of one or more continuous variables).

For an unsupervised learning problem, we don't have the desired output for the input variables. The goal of pattern recognition is: to discover groups of similar examples within the data (clustering), to determine the distribution of data within the input space (density estimation), to project the data from a high-dimensional space to low-dimensional space etc..

For simplicity, we can generate the information for this task from the mathematical function $\text{Sin}(2\pi x)$ with random Gaussian interference enclosed in the target v variable, i.e, for any input x, target $t=\text{Sin}(2\pi x)+\epsilon t=\text{Sin}(2\pi x)+\epsilon$. Let the training set consists of N samples with inputs as $A=(a1,a2,\ldots,aN)t$ and the corresponding target variables as $t=(t1,t2,\ldots,tN)Tt=(t1,t2,\ldots,tN)T$. Let the polynomial function used for the prediction, whose order is M, is:

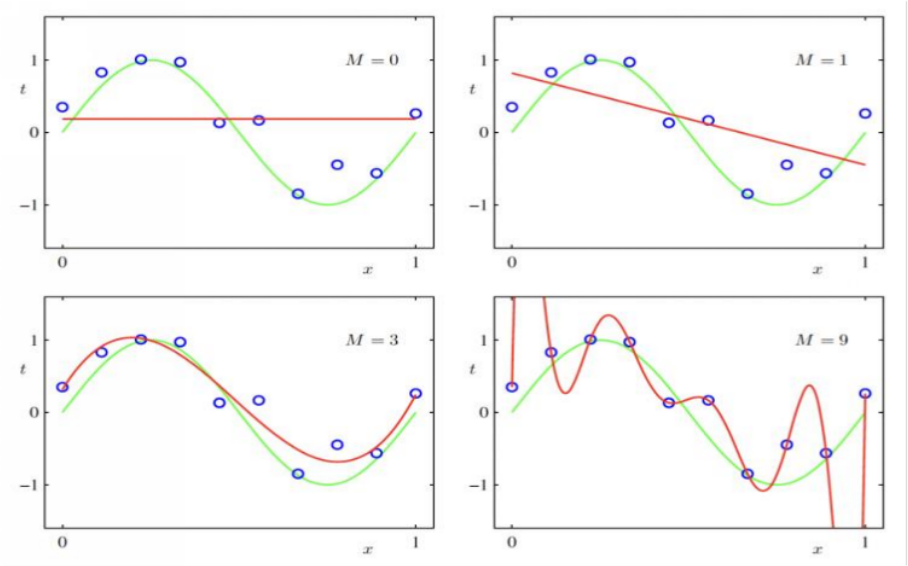
$y(x,w)=w0+w1x+w2x2+\ldots\ldots+wMxM=\sum j=0Mwjxj$

This polynomial function is linear with respect to the coefficients w. The goal of the pattern recognition task is to minimize the error in predicting t. Or we can say that we have to minimize some error function which should encode how much we deviated from the actual value while doing the prediction. One of the common choice of error function is:

E(w)=12∑n=1N[y(xn,w)−tn]2E(w)=12∑n=1N[y(xn,w)−tn]2

The erroneousness function is quadratic in w and hence taking it's derivative w.r.t w and equating it to 0 gives us a unique solution w∗ for the problem.

One of the important parameter in deciding how well the solution will perform on the unseen data is the order of the polynomial function M. As shown in the below figure, if we keep on increasing M, we will get a errorless fit on the preparation data getting the training error E(w∗)=0E(w∗)=0 (called as over fitting) but the prediction on unseen data will be flawed. The best fit polynomial seems to be the one which has on order M=3.



	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

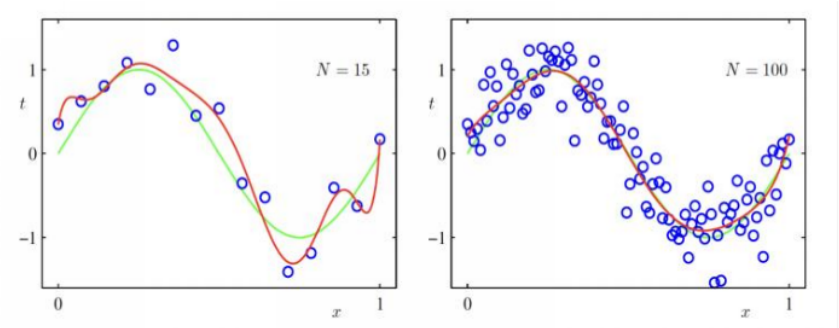
Based on these coefficients, one of the techniques which can be used to compensate for the problem of over fitting is condition which concern adding a reward term to the error mathematical function which courageousness the coefficients from getting larger in magnitude. The modified error function is given as:

$$E\tilde{}(w)=12\sum n=1N(y(xn,w)-tn)2+\lambda 2 \,//\, w \,//\, 2E\sim(w)=12\sum n=1N(y(xn,w)-tn)2+\lambda 2|w|2$$

where $// w // 2=wTw=w20+w21+\ldots+w2M|w|2=wTw=w02+w12+\ldots+wM2.$

This technique is also called as shrinkage method and a quadratic regularizer is called as ridge regression.

Another way to decrease over fitting or to use the complex models for prediction is by growing the sample sized of the preparation data. The same order M=9 polynomial is fit on N=15 and N=100 data points and result is shown in the left and the right figure below. It can be seen that the exploding the number of data points reduces the problem of over fitting.



Model Complexity in Machine Learning

Model complexity refers is a measure of how well a model can capture the underlying patterns in the data. In the context of machine learning, model complexity is often associated with the number of parameters in a model and its ability to fit both the training data and generalize to new, unseen data.

There are **two main aspects** of **model complexity**:

Simple Models: Simple models have few parameters, making them less flexible therefore they struggle to capture the complexity of the underlying patterns in the data leading to under fitting, where the model performs poorly on the training data as well as on unseen data.

Complex Models: Complex models have a larger number of parameters, allowing them to represent more intricate relationships in the data. While complex models may perform well on the training data, model tends to over fitting.

Modelling complexity can be influenced by several factors:

Number of Features: The more attributes or features your model scrutinizes, the higher its complexity is likely to be. Too many features can potentially magnify noise and result in over fitting.

Model Algorithm: The nature of the algorithm used influences the complexity of the model. For instance, decision trees are considerably simpler than neural networks.

Hyper parameters: Settings such as the learning rate, number of hidden layers, and regularization parameters can influence the complexity of a machine learning model.

Model Complexity Importance

Finding the optimal model complexity is important because:

1.Bias-Variance Trade off: Model complexity is closely related to the bias-variance trade off. Simple models may have high bias (systematic errors), while complex models may have high variance (sensitivity to small fluctuations in the training data). Finding the right level of complexity involves managing this trade off to achieve good predictive performance.

2.Computational Resources: Complex models often require more computational resources for training and inference. The choice of model complexity may be influenced by practical considerations such as available computing power and time constraints.

3.Interpretability: Simple models are often more interpretable, making it easier to understand and explain their decision-making processes. In some cases, interpretability is crucial, especially in sensitive applications where decisions impact individuals' lives.

Avoid Model Complexity and Over fitting

Addressing model complexity and over fitting is critical to achieving robust machine learning models. Here are some strategies:

Regularization: Regularization techniques introduce penalties for complexity in the loss function of the model which discourages learning overly complex model parameters, discouraging over fitting. L1 and L2 regularization are common methods to control the magnitude of coefficients, preventing the model from becoming overly complex.

Cross-validation: Cross-Validation is a technique that Assess model generalization and provides a realistic measure of how well the model is likely to perform on unseen data, helping to assess its level of complexity and over fitting.

Reducing Features: By minimizing the number of input features, we could lower the complexity, and thus, prevent over fitting.

Use of Ensemble Models: Combining predictions from multiple diverse models can often lead to better performance and reduced risk of over fitting compared to relying on a single model. This is because individual models may have unique strengths and weaknesses, and averaging their predictions can lead to a more robust and generalizable result.

Early Stopping: By monitoring the validation error during training, we can stop the training process when the validation error starts to increase, even if the training error continues to decrease. This prevents the model from learning irrelevant patterns in the training data that could lead to over fitting.

Split the dataset into training and testing Data: Splitting your dataset is crucial because it ensures the model doesn't simply memorize the training data and can generalize to unseen examples.

Multivariate Non-linear Functions in Neural Networks

1 Linearity

A neural network is only non-linear if you squash the output signal from the nodes with a non-linear activation function. A complete neural network (with non-linear activation functions) is an arbitrary function approximator.

Bonus: It should be noted that if you are using linear activation functions in multiple consecutive layers, you could just as well have pruned them down to a single layer due to them being linear. (The weights would be changed to more extreme values). Creating a network with multiple layers using linear activation functions would not be able to model more complicated functions than a network with a single layer.

2 Activation signal

Interpreting the squashed output signal could very well be interpreted as the strength of this signal (biologically speaking). Thought it might be incorrect to interpret the output strength as an equivalent of confidence as in fuzzy logic.

3 Non-linear activation functions

The input signals along with their respective weights are a linear combination. The non-linearity comes from your selection of activation functions. Remember that a linear function is drawn as a line - sigmoid, tanh, ReLU and so on may not be drawn with a single straight line.

Need of non-linear activation functions

Most functions and classification tasks are probably best described by non-linear functions. If we decided to use linear activation functions we would end up with a much coarser approximation on a complex function.

Ex: Universal approximators

1. Multivariate Non-linear Functions

Multivariate non-linear functions involve multiple variables and exhibit non-linear relationships between their inputs and outputs.

In neural networks, these functions model complex relationships between features in the data that cannot be captured by simple linear equations.

2. Role in Neural Networks

Neural networks use non-linear activation functions to transform linear combinations of inputs into non-linear mappings.

This non-linearity allows the network to approximate complex decision boundaries and solve problems like image recognition, speech processing, and natural language tasks.

3. Mathematical Formulation

Given input features x1 ,x2 ,…,xn , weights w1,w2,…,wnw_1, w_2, \ldots, w_nw1 ,w2 ,…,wn , and bias b: z=w1x1+w2x2+…+wnxn+bz = w_1x_1 + w_2x_2 + \ldots + w_nx_n + bz=w1 x1 +w2 x2 +…+wn xn +b

The non-linear transformation f(z)f(z)f(z) is applied to produce: y=f(z)y = f(z)y=f(z)

4. Common Non-linear Activation Functions

Sigmoid: For probabilities (e.g., binary classification). f(z)=1/1+e−z1

Tanh: For zero-centered outputs.
f(z)=ez+e−z/ez−e−z

ReLU: For introducing sparsity.f(z)=max(0,z)

Softmax: For multi-class classification. f(zi)=∑j=1N ezj ezi

5. Importance of Non-linearity

Without non-linearity:

Neural networks become equivalent to a linear regression model, regardless of the number of layers.

They cannot capture the intricate patterns in data required for tasks like image or speech recognition.

With non-linearity:

Networks can approximate any continuous function, as stated by the Universal Approximation method.

Bayes Theorem

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

P(A | B)=P(B | A)P(A)P(B)P(A | B)=P(B)P(B | A)P(A)

where A and B are events and **P(B) ≠ 0**

Basically, we are trying to find probability of event A, given the event B is true.

Event B is also termed as evidence.

P(A) is the priori of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).

P(B) is Marginal Probability: Probability of Evidence.

P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.

P(B|A) is Likelihood probability i.e the likelihood that a hypothesis will come true based on the evidence.

Now, with regards to our dataset, we can apply Bayes' theorem in following way:

P(y | X)=P(X | y)P(y)P(X)P(y | X)=P(X)P(X | y)P(y)

where, y is class variable and X is a dependent feature vector (of size n) where:

X=(x1,x2,x3,…..,xn)X=(x1 ,x2 ,x3 ,…..,xn)

Just to clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of dataset)

X = (Rainy, Hot, High, False)y = No

Naive Bayes Assumption

The fundamental Naive Bayes assumption is that each feature makes an:

Feature independence: The features of the data are conditionally independent of each other, given the class label.

Continuous features are normally distributed: If a feature is continuous, then it is assumed to be normally distributed within each class.

Discrete features have multinomial distributions: If a feature is discrete, then it is assumed to have a multinomial distribution within each class.

Features are equally important: All features are assumed to contribute equally to the prediction of the class label.

No missing data: The data should not contain any missing values.

Naive Bayes Classifiers

A Naive Bayes classifiers, a family of algorithms based on Bayes' Theorem. Despite the "naive" assumption of feature independence, these classifiers are widely utilized for their simplicity and efficiency in machine learning. The article delves into theory, implementation, and applications, shedding light on their practical utility despite oversimplified assumptions.

Naive Bayes

The "Naive" part of the name indicates the simplifying assumption made by the Naive Bayes classifier. The classifier assumes that the features used to describe an observation are conditionally independent, given the class label. The "Bayes" part of the name refers to Reverend Thomas Bayes, an 18th-century statistician and theologian who formulated Bayes' theorem.

Consider a fictional dataset that describes the weather conditions for playing a game of golf. Given the weather conditions, each tuple classifies the conditions as fit("Yes") or unfit("No") for playing golf.Here is a tabular representation of our dataset.

Naive Bayes Classifier Advantages

1. Easy to implement and computationally efficient.
2. Effective in cases with a large number of features.
3. Performs well even with limited training data.
4. It performs well in the presence of categorical features.
5. For numerical features data is assumed to come from normal distributions

Naive Bayes Classifier Disadvantages

1. Assumes that features are independent, which may not always hold in real-world data.
2. Can be influenced by irrelevant attributes.
3. May assign zero probability to unseen events, leading to poor generalization.
4. Applications of Naive Bayes Classifier
5. Spam Email Filtering: Classifies emails as spam or non-spam based on features.
6. Text Classification: Used in sentiment analysis, document categorization, and topic classification.
7. Medical Diagnosis: Helps in predicting the likelihood of a disease based on symptoms.
8. Credit Scoring: Evaluates creditworthiness of individuals for loan approval.
9. Weather Prediction: Classifies weather conditions based on various factors.

Decision boundary

In a statistical-classification problem with two classes, a decision boundary or decision surface is a hyper surface that partitions the underlying vector space into two sets, one for each class. The classifier will classify all the points on one side of the decision boundary as belonging to one class and all those on the other side as belonging to the other class.

A Decision extent is a hyper surface in machine learning that delineates the boundaries of classes. When the model's prediction shifts from combined class to another and the dimension space is represented by this area.

A decision boundary is the region of a problem space in which the output label of a classifier is ambiguous. If the decision surface is a hyperplane, then the classification problem is linear, and the classes are linearly separable.

Decision boundaries are not always clear cut. That is, the transition from one class in the feature space to another is not discontinuous, but gradual. This effect is common in fuzzy logic based classification algorithms, where membership in one class or another is ambiguous.

Decision boundaries can be approximations of optimal stopping boundaries. The decision boundary is the set of points of that hyperplane that pass through zero. For example, the angle between a vector and points in a set must be zero for points that are on or close to the decision boundary.

In the case of backpropagation based artificial neural networks or perceptrons, the type of decision boundary that the network can learn is determined by the number of hidden layers the network has. If it has no hidden layers, then it can only learn linear problems. If it has one hidden layer, then it can learn any continuous function on compact subsets of Rn as shown by the universal approximation theorem, thus it can have an arbitrary decision boundary.

In particular, support vector machines find a hyperplane that separates the feature space into two classes with the maximum margin. If the problem is not originally linearly separable, the kernel trick can be used to turn it into a linearly separable one, by increasing the number of dimensions. Thus a general hyper surface in a small dimension space is turned into a hyperplane in a space with much larger dimensions.

Neural networks try to learn the decision boundary which minimizes the empirical error, while support vector machines try to learn the decision boundary which maximizes the empirical margin between the decision boundary and data points.

Types of Decision Boundaries

The complexity of the model and the characteristics used determine the kind of decision boundary learned by a machine learning method. Common decision-learning boundaries in machine learning include the following:

Linear

A linear decision boundary is a line that demarcates one feature space class from another.

Non-linear

A non-linear decision border is a curve or surface that delineates a set of categories. Learning non-linear decision boundaries is possible in non collinear models like Decision trees, SVM and ANN.

Piecewise Linear

Linear segments are joined together to produce a piece wise linear curve, which is the piece wise linear decision boundary. Piece wise linear decision boundaries may be learned by some Decision trees & Random forests.

Clustering

The boundaries between groups of information points in a attribute area are called “clustering decision boundaries.” K-means and DBSCAN are only two examples of clustering algorithms whose decision limits may be learned.

Probabilistic

A data point's likelihood of belonging to one group or another is represented by a border called a probabilistic decision boundary. Probabilistic models may be trained to learn probabilistic decision boundaries, including Naive Bayes and Gaussian Mixture Models.

Risk minimization

Risk Minimization refers to the process of minimizing the expected risk in learning tasks. It involves the analysis and minimization of both empirical risk and functional risk, with the goal of finding the optimal solution.

We can interpret learning as the outcome of the minimization of the expected risk. In the previous section, we have analyzed different kinds of loss function, that are used to construct the expected and the empirical risk. In this section, we discuss the minimization of the risk with the main purpose of understanding the role of the chosen loss in the optimal solution.

In general, we would like to minimize the expected risk E(f). This can be carried out by an elegant analysis based on variational calculus. Here, we will concentrate on the minimization of the empirical risk, which is what is used in the real-word. Interestingly, the study on the empirical risk will also indicate the structure of the minimum of the functional risk.

We begin discussing the case of regression. Let us consider the class of loss functions. We discuss the cases p=0,1,2, and $+\infty$, which are more commonly adopted. We will also assume that the marginal probability density p1 is strictly positive: p1>0. The process of learning is converted into the problem of minimizing.

$E=E|Y-f(X)|pp=1p\int X\times Y|y-f(x)|pdP(x,y)$.

ERM - Empirical Risk Minimization

The Empirical Risk Minimization (ERM) principle is a learning paradigm which consists in selecting the model with minimal average error over the training set. This so-called training error can be seen as an estimate of the risk (due to the law of large numbers), hence the alternative name of empirical risk.

By minimizing the empirical risk, we hope to obtain a model with a low value of the risk. The larger the training set size is, the closer to the true risk the empirical risk is.

If we were to apply the ERM principle without more care, we would end up learning by heart, which we know is bad. This issue is more generally related to the over fitting phenomenon, which can be avoided by restricting the space of possible models when searching for the one with minimal error. The most severe and yet common restriction is encountered in the contexts of linear classification or linear regression. Another approach consists in controlling the complexity of the model by regularization.

There are five elementary methods of Risk management:

1. "Avoidance"
2. "Retention"
3. "Spreading"
4. "Loss Prevention and Reduction"
5. "Transfer"

Avoidance: Many times it is not possible to completely avoid risk but the possibility should not be overlooked. For example, at the height of a thunderstorm, Physical Plant may not release vehicles for travel until the weather begins to clear, thus avoiding the risk of auto accidents during severe weather. Some buildings on campus have had repeated water problems in some areas. By not allowing storage of records or supplies in those areas, some water damage claims may be avoided.

Retention: At times, based on the likely frequency and severity of the risks presented, retaining the risk or a portion of the risk may be cost-effective even though other methods of handling the risk are available. For example, the University retains the risk of loss to fences, signs, gates and light poles because of the difficulty of enumerating and evaluating all of these types of structures. When losses occur, the cost of repairs is absorbed by the campus maintenance budget, except for those situations involving the negligence of a third party. Although insurance is available, the University retains the risk of loss to most University personal property.

Spreading: It is possible to spread the risk of loss to property and persons. Duplication of records and documents and then storing the duplicate copies in a different location is an example of spreading risk. A small fire in a single room can destroy the entire records of a department's operations. Placing people in a large number of buildings instead of a single facility will help spread the risk of potential loss of life or injury.

Loss Prevention and Reduction: When risk cannot be avoided, the effect of loss can often be minimized in terms of frequency and severity. For example, Risk Management encourages the use of security devices on certain audio visual equipment to reduce the risk of theft. The University requires the purchase of health insurance by students who are studying abroad, so that they might avoid the risk of financial difficulty, should they incur medical expenses in another country.

Transfer: In some cases risk can be transferred to others, usually by contract. When outside organizations use University facilities for public events, they must provide evidence of insurance and name the University as an additional insured under their policy, thereby transferring the risk of the event from the University to the facility user. The purchase of insurance is also referred to as a risk transfer since the policy actually shifts the financial risk of loss, contractually, from the insured entity to the insurance company. Insurance should be the last option and used only after all other techniques have been evaluated.

Contracts: Often vendors and service providers will attempt through a contract to release themselves from all liability for their actions relating to the contract. These are often referred to as "hold harmless or indemnification" clauses. Due to the complexity of interpreting these provisions, the President has delegated contracting authority for the University solely to staff in Contracts & Procurement. The Office of University Risk Management reviews contracts and agreements as requested by Contracts & Procurement to identify and assess risks, evaluate insurance standards, and review hold harmless and indemnification provisions. The Chancellor's Office requires that the University obtain in most instances not only a Certificate of Insurance, but also an Endorsement. Collecting these documents is often the most time consuming aspect of the contracting process.

Density estimation

In statistics, probability density estimation or simply density estimation is the construction of an estimate, based on observed data, of an unobservable underlying probability density function. The unobservable density function is thought of as the density according to which a large population is distributed; the data are usually thought of as a random sample from that population.

Example

We will consider records of the incidence of diabetes.

The following is quoted verbatim from the data set description:

A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes mellitus according to World Health Organization criteria. The data were collected by the US National Institute of Diabetes and Digestive and Kidney Diseases. We used the 532 complete records.

In this example, we construct three density estimates for "glu" (plasma glucose concentration), one conditional on the presence of diabetes, the second conditional on the absence of diabetes, and the third not conditional on diabetes. The conditional density estimates are then used to construct the probability of diabetes conditional on "glu".

The "glu" data were obtained from the MASS package of the R programming language. Within R, ?Pima.tr and Pima.te give a fuller account of the data.

The mean of "glu" in the diabetes cases is 143.1 and the standard deviation is 31.26. The mean of "glu" in the non-diabetes cases is 110.0 and the standard deviation is 24.29. From this we see that, in this data set, diabetes cases are associated with greater levels of "glu". This will be made clearer by plots of the estimated density functions.

The first figure shows density estimates of p(glu | diabetes=1), p(glu | diabetes=0), and p(glu). The density estimates are kernel density estimates using a Gaussian kernel. That is, a Gaussian density function is placed at each data point, and the sum of the density functions is computed over the range of the data.

From the density of "glu" conditional on diabetes, we can obtain the probability of diabetes conditional on "glu" via Bayes' rule. For brevity, "diabetes" is abbreviated "db." in this formula.

$$p(\text{diabetes}=1|\text{glu})=p(\text{glu}|\text{db.}=1)p(\text{db.}=1)p(\text{glu}|\text{db.}=1)p(\text{db.}=1)+p(\text{glu}|\text{db.}=0)p(\text{db.}=0)$$

The second figure shows the estimated posterior probability p(diabetes=1 | glu). From these data, it appears that an increased level of "glu" is associated with diabetes.

Application and purpose

A very natural use of density estimates is in the informal investigation of the properties of a given set of data. Density estimates can give a valuable indication of such features as skewness and multimodality in the data. In some cases they will yield conclusions that may then be regarded as self-evidently true, while in others all they will do is to point the way to further analysis and/or data collection.

An important aspect of statistics is often the presentation of data back to the client in order to provide explanation and illustration of conclusions that may possibly have been obtained by other means. Density estimates are ideal for this purpose, for the simple reason that they are fairly easily comprehensible to non-mathematicians.

More examples illustrating the use of density estimates for exploratory and presentational purposes, including the important case of bivariate data.

Density estimation is also frequently used in anomaly detection or novelty detection, if an observation lies in a very low-density region, it is likely to be an anomaly or a novelty.

In hydrology the histogram and estimated density function of rainfall and river discharge data, analyzed with a probability distribution, are used to gain insight in their behaviour and frequency of occurrence. An example is shown in the blue figure.

Parametric Methods

Parametric methods are statistical techniques that rely on specific assumptions about the underlying distribution of the population being studied. These methods typically assume that the data follows a known Probability distribution, such as the normal distribution, and estimate the parameters of this distribution using the available data.

The basic idea behind the Parametric method is that there is a set of fixed parameters that are used to determine a probability model that is used in Machine Learning as well. Parametric methods are those methods for which we priory know that the population is normal, or if not then we can easily approximate it using a Normal Distribution which is possible by invoking the Central Limit Theorem.

Parameters for exploitation the statistical distribution are as follows:

A. Mean

B. Standard Deviation

Eventually, the classification of a method to be parametric completely depends on the presumptions that are made about a population.

Parametric Methods Assumptions

Parametric methods require several assumptions about the data.

Normality: The data follows a normal (Gaussian) distribution.

Homogeneity of variance: The variance of the population is the same across all groups.

Independence: Observations are independent of each other.

Parametric Methods

Statistical Tests:

t-test: Tests for the difference between the means of two independent groups.

ANOVA: Tests for the difference between the means of three or more groups.

F-test: Compares the variances of two groups.

Chi-square test: Tests for relationships between categorical variables.

Correlation analysis: Measures the strength and direction of the linear relationship between two continuous variables.

Machine Learning Models

Linear regression: Predicts a continuous outcome based on a linear relationship with one or more independent variables.

Logistic regression: Predicts a binary outcome (e.g., yes/no) based on a set of independent variables.

Naive Bayes: Classifies data points based on Bayes' theorem and assuming independence between features.

Hidden Markov Models: Models sequential data with hidden states and observable outputs.

Parametric Methods Advantages

More powerful: When the assumptions are met, parametric tests are generally more powerful than non-parametric tests, meaning they are more likely to detect a real effect when it exists.

More efficient: Parametric tests require smaller sample sizes than non-parametric tests to achieve the same level of power.

Provide estimates of population parameters: Parametric methods provide estimates of the population mean, variance, and other parameters, which can be used for further analysis.

Parametric Methods Disadvantages

Sensitive to assumptions: If the assumptions of normality, homogeneity of variance, and independence are not met, parametric tests can be invalid and produce misleading results.

Limited flexibility: Parametric methods are limited to the specific probability distribution they are based on.

May not capture complex relationships: Parametric methods are not well-suited for capturing complex non-linear relationships between variables.

Parametric Methods Applications

Parametric methods are widely used in various fields, including

1. **Bio-statistics:** Comparing the effectiveness of different treatments.
2. **Social sciences:** Investigating relationships between variables.
3. **Finance:** Estimating risk and return of investments.
4. **Engineering:** Analyzing the performance of systems.

Maximum Likelihood Method

In statistics, Maximum Likelihood Estimation (MLE) is a method of estimating the parameters of an assumed probability distribution, given some observed data. This is achieved by maximizing a likelihood function so that, under the assumed statistical model, the observed data is most probable. The point in the parameter space that maximizes the likelihood function is called the maximum likelihood estimate. The logic of maximum likelihood is both intuitive and flexible, and as such the method has become a dominant means of statistical Inference Principles.

The Maximum Likelihood Method is the most popular technique for deriving estimators. It is based on the likelihood function which, for an observed sample x, is defined as the probability (or density) of x expressed as a function of θ; in symbols $L(\theta)=\prod i=1nf(xi;\theta)$.

This function provides a measure of plausibility of each possible value of θ on the basis of the observed data. Then, the method at issue consists of estimating θ through the value of θ which maximizes L(θ) since this corresponds to the parameter value for which the observed sample is most likely. The estimate found in this way, that is,

$\theta\hat{}=\theta\hat{}(x)$such that$L(\theta\hat{})=sup\theta\in\Theta L(\theta)$

Maximum-Likelihood Approach

The Maximum Likelihood Approach is, far and away, the preferred approach to correcting for non-response bias, and it is the one advocated by Little and Rubin. The maximum-likelihood approach begins by writing down a probability distribution that defines the likelihood of the observed sample as a function of population and distribution parameters θ. If x1 and x2 represent responses to two different survey questions by a single individual, the likelihood associated with a complete response may be expressed as f(x1, x2; θ), where f is the joint probability density function of x1 and x2.

For individuals who only report x1, the likelihood associated with x1 is $\int_{-\infty}^{\infty} f(x1, x2; \theta) dx2$, which can, under the assumption of joint normality, be simplified to a more convenient form. In this way, a likelihood function is specified that includes terms corresponding to each observation, whether completely or only partially observed. The likelihood objective is then maximized with respect to θ, which produces estimates of the desired characteristics, enjoying all the well-known properties of maximum-likelihood estimation.

Bayesian inference

Bayesian inference is a method of statistical inference in which Bayes' theorem is used to calculate a probability of a hypothesis, given prior evidence, and update it as more information becomes available. Fundamentally, Bayesian inference uses a prior distribution to estimate posterior probabilities. Bayesian inference is an important technique in statistics, and especially in mathematical statistics. Bayesian updating is particularly important in the dynamic analysis of a sequence of data. Bayesian inference has found application in a wide range of activities, including science, engineering, philosophy, medicine, sport, and law. In the philosophy of decision theory, Bayesian inference is closely related to subjective probability, often called "Bayesian probability".

Bayes rule Introduction

Bayesian inference derives the posterior probability as a consequence of two antecedents: a prior probability and a "likelihood function" derived from a statistical model for the observed data. Bayesian inference computes the posterior probability according to Bayes' theorem.