# Setting the context … By turning a page of History

*"If you don't know history, then you don't know anything. You are a leaf that doesn't know it is part of a tree. "* - Michael Crichton

## 1.1. Introduction

History of Humans and their evolution goes back to 70,000 years ago. In this course of time, our ancestors have witnessed major changes in the Earth eco system. Some organism got extinct; some new breeds were discovered. People migrated to new places, settled new colonies. Humans achieved major mathematical and scientific breakthroughs as their ever-evolving requirements grew, they kept on innovating and researching. Science has always been the biproduct of requirements of Human society. That is how, most of the major inventions of modern Human era came into existence, as scientists tried to solve various problems of their specific domains, driven with the goal of making a salient contribution to the lives of human society. This evolution of human cognitive function has been in pace with the course of time and progress of society, until the last century.

Leonardo Da Vinci (1452-1519) was convinced that humans would be able to fly like birds. He had this insight some 200 years before the actual aero-plane model was conceived. Benjamin Franklin would not very surprised if he would discover large locomotives and machines operating on electrical circuits.

However, I can bet they would be more than surprised, even fascinated, and overwhelmed, if they got to experience what Computers can do today. The advent of Digital age has grown into many folds exponentially. Today, we can order groceries at our home, book flight tickets around the globe, oh … wait, we can even experience a place without even visiting it, thanks to Augmented Reality and Meta verse. To cut it short, if you are a 90s kid, you are the bridge that has seen the crossover of Sci-fi films into the reality of Daily Lives of average human.

So how did computers evolve so quickly and manage to do all these stuffs? It was in only 1944 when Howard H. Aiken (1900-1973) built the Mark I electromechanical computer, with the assistance of IBM at Harvard. Military code breaking capabilities in world war II led to various computational projects. Some scientists started building computers to solve mathematical equations by 1950s. However, it was in 1962, when Computer Science as a discipline came into existence at Purdue University. This led to assembling of various theoretical branches of Computer applicability under one roof. Decade of 1960s saw an unprecedented development in the field of computer science. Operating systems made major advancements. New programming languages such as BASIC were invented. Intel designed the first microprocessor (computer on a chip) in 1969-1971. 1970s saw major advances in the field of data management. Operating System like Linux and supercomputers like CRAY-1 were designed. Computational ability of computers grew significantly. A major breakthrough came in 1981 when first portable computer Osborne I was marketed by IBM, however, personal computers became a revolution thanks to Steve Wozniak and Steve Jobs, founders of Apple Computer. And since then, there is no turning back, and by 1990s with boom of internet, the landscape of computer applicability changed for good. Computers transitioned from computational machines to daily usage machines, adapted by businesses, writers, salesperson, students, etc. Today computers are not just medium of storing online documents or automating a business flow, but they have become a mode of communication, thanks to internet.

Computer Science as a discipline grew many folds. New programming languages came into picture, which helped develop break through applications for enterprise business. Their applicability relied on programming tasks, storing information and their retrieval (database) and making sense of stored information (machine learning). Let us briefly look at the history of these three aspects, how they evolved in parallel and yet kept converging.

## 1.2. What is Programming?

***Programming** involves the development of software and algorithms to instruct computers. Programming is the process of creating computer programs or software by writing sets of instructions that a computer can execute. It involves designing, coding, testing, and maintaining computer programs to solve specific problems or perform desired tasks.*

At its core, programming is about giving precise instructions to a computer to perform a series of operations. These instructions are written using programming languages, which serve as a medium of communication between humans and computers. Programming languages vary in syntax and capabilities, but they all provide a set of rules and structures to express computations and manipulate data. Programmers use various tools and integrated development environments (IDEs) to write, edit, test, and debug their code. They use programming concepts such as variables, loops, conditionals, functions, and data structures to create efficient and maintainable programs.

The applications of programming are vast and diverse. It is used to develop software applications, websites, mobile apps, video games, artificial intelligence systems, databases, and much more. In today's interconnected world, programming has become an essential skill, empowering individuals, and businesses to leverage the power of computers to automate tasks, process data, and create innovative solutions.

## 1.3. History of Programming

### 1.3.1. World’s first Programmer: Ada Lovelace

Growth of Computer Science in last century has been rapid fast with mathematics adding wings to its flight. In the early days of computing, the focus was primarily on developing coding languages and algorithms to perform calculations and solve mathematical problems. Ada Lovelace's collaboration with Charles Babbage on the Analytical Engine in the 19th century laid the foundation for modern programming. In 1843, Lovelace wrote an algorithm for the Analytical Engine, which is considered the first computer program. She recognized the potential of numbers and their ability to represent more than just numerical values, envisioning that machines could be programmed to perform complex calculations and manipulate symbols. Lovelace's algorithm for computing Bernoulli numbers showcased the concept of step-by-step instructions that could be executed by a machine, serving as a blueprint for modern programming. Her visionary ideas about the capabilities of computing machines and the significance of programming were ahead of their time.

Although the Analytical Engine was never built during Lovelace's lifetime, her work and insights paved the way for future advancements in programming. Her contributions to the field earned her the title of the world's first programmer.

Since Ada Lovelace's era, programming languages have evolved significantly. They have become more powerful, expressive, and specialized, allowing humans to interact with computers more efficiently and effectively. Today, there is a wide range of programming languages available, each designed to address specific needs and cater to different paradigms of programming.

The development of programming languages has not only focused on mathematical computations but has expanded to encompass various domains such as web development, data analysis, artificial intelligence, and more. Modern programming languages offer a rich set of features, libraries, and frameworks that empower developers to create sophisticated software applications, from simple scripts to complex systems.

### 1.3.2. How Humans pass command to Computers?

**Assembly language** *is a low-level programming language that bridges the gap between machine code (binary instructions directly understood by a computer's hardware) and higher-level programming languages.*

It uses mnemonic instructions that represent specific machine operations and memory locations. It came to prominence in 1949. In assembly language, each mnemonic instruction corresponds to a specific machine instruction. These instructions are then converted into machine code using an assembler, a specialized program that translates assembly language into machine code. Assembly language provides direct access to the underlying hardware of a computer, allowing programmers to write code that can take full advantage of the computer's capabilities. It is often used in situations where performance is critical or when direct hardware control is required, such as in embedded systems, device drivers, and operating system kernels.

Alick Glennie's Autocode is considered by some to be the first compiled computer programming language. Autocode was developed in the early 1950s at the University of Manchester in England. Autocode was designed to simplify programming for the Manchester Mark 1 computer, an early computer system. It allowed programmers to write high-level instructions that could be translated directly into machine code. The key innovation of Autocode was its ability to be compiled. **Compilation** is the process of translating a program written in a high-level language into machine code that can be executed directly by the computer's hardware. Autocode was one of the earliest languages to introduce this concept. Autocode eliminated the need for manual translation of instructions into machine code, making programming more efficient and less error prone. It allowed programmers to express their instructions in a higher-level language, which was then transformed into machine code by the compiler.

The development of Autocode marked a significant advancement in programming languages, as it demonstrated the potential for translating high-level instructions into machine code automatically. This laid the foundation for future compiled languages and paved the way for the development of more sophisticated programming languages.

The programming languages, such as FORTRAN (1957, John Backus) and COBOL (1959, Dr. Grace Murray Hopper), were developed to facilitate scientific and business computations respectively. These early languages introduced the concepts of control flow, variables, and subroutines, laying the foundation for subsequent programming languages. FORTRAN, in particular, gained widespread adoption and was instrumental in advancing the idea of high-level languages that could be compiled.

In the 1960s and 1970s, procedural programming languages, such as ALGOL (1960) and C (1972), gained prominence. Procedural programming focused on breaking down complex problems into smaller, manageable procedures or functions. It introduced concepts like loops, conditionals, and modularization, improving code organization and reusability. Procedural programming languages remained popular for several decades and are still widely used today.

The 1980s witnessed a significant paradigm shift with the emergence of object-oriented programming. Languages like Simula (1967) and Smalltalk (1972) laid the groundwork for OOP, but it was the introduction of C++ (1983) and later Java (1995, originally intended to be used with hand-held devices) that popularized the paradigm. OOP emphasized the organization of code around objects, encapsulating data, and behavior together. This approach provided modularity, reusability, and the ability to model real-world entities effectively.

Functional programming, rooted in mathematical concepts, gained traction in the 1970s and 1980s. Languages like Lisp (1958) and ML (1973) introduced functional programming concepts such as higher-order functions, immutability, and recursion. Functional programming focused on composing functions and minimizing mutable state, leading to more concise and maintainable code. Modern functional programming languages like Haskell (1990) and Elixir (2011) continue to be influential.

Domain-specific languages (DSLs) were designed to address specific problem domains, enabling developers to express solutions in a language tailored to the problem at hand. DSLs like HTML (1993) for web development and MATLAB (1984) for mathematical computations have gained widespread adoption.

As computers became more powerful and multi-core architectures became prevalent, concurrent, and parallel programming gained importance. Concurrent programming focused on handling multiple tasks that execute independently but potentially interact with each other, while parallel programming aimed to leverage multiple processing units for increased performance. Languages like Erlang (1986) and Go (2009) were designed to provide built-in support for concurrency and parallelism. Martin Odersky created Scala as a programing language that combines aspects of functional programming.

In recent years, programming has witnessed a convergence of paradigms and the rise of new trends. Languages like Python (1991), developed by Guido Van Rossum as a simplified computer language that is easy to read, have embraced multiple programming paradigms, offering flexibility and expressiveness. Functional programming concepts have influenced mainstream languages like JavaScript, while OOP principles are employed.

Modern computer programming languages have evolved from earlier languages, with many older languages still in use or serving as the basis for new languages. The newer languages strive to simplify the programming process for developers. Programming skills helped developers to use their creativity and develop useful software products for enterprises. And these products interacted with data of enterprises, which are stored using Databases.

## 1.4. Story of Databases

Humans have been storing information (gathering data) since ages. However, as computers became more accessible and businesses started using them for practical purposes, the need for managing and storing large amounts of data became apparent. Initially, data was considered secondary to the primary computational tasks. It was often stored in files or even on punched cards, and the emphasis was on processing and performing calculations. But as businesses started relying on computers to handle their operations and store valuable information, the importance of managing data efficiently grew. This led to the emergence of databases as a structured and organized way to store, retrieve, and manage data.

***A database is a self-describing collection of integrated records***. A record is a representation of some physical or conceptual object. A database is self-describing in the context that it contains a description of its own structure. This description is called **metadata -** *data about the data*. The database is integrated in that it includes the relationships among data items, as well as including the data items themselves (Kroenke & Auer, 2007).

The integration aspect of a database refers to the inclusion of relationships among data items in addition to the data items themselves. This means ***that a database not only holds the individual data items but also captures the connections and associations between different data elements.***

By incorporating metadata and maintaining relationships among data items, databases provide a structured and organized approach to storing and managing information, making it easier to search, update, and retrieve information as needed. They allowed for the creation of relationships between different sets of data, enabling more complex and sophisticated data management.

Over time, databases evolved to include more advanced features, such as support for multiple users accessing the data simultaneously, data integrity constraints, and data security mechanisms. Relational databases, which organize data into tables and use structured query language (SQL) for managing data, became particularly popular and widely adopted.

With the advent of the internet and the explosive growth of digital information, databases became even more crucial. They underpin numerous applications and services we rely on today, from e-commerce platforms and social media sites to banking systems and healthcare records.

In recent years, we've also seen the rise of non-relational databases, such as NoSQL databases, which provide flexible data models and scalability options to handle the demands of modern applications.

### 1.4.1. History of DATABASES

During the 1960s and mid-1960s, the popularity and cost-effectiveness of computers led to a new direction in the field of databases. The term “Database” emerged during this time, coinciding with the availability of direct-access storage devices like disks and drums. These storage technologies allowed for shared interactive use, contrasting with the previous tape-based systems that relied on batch processing.

#### 1.4.1.1. Primitive Digital Databases

During the 1960s, hierarchical databases emerged as one of the earliest database management systems (DBMS). IBM's Information Management System (IMS) was a notable example of a hierarchical database. It organized data in a tree-like structure, with parent-child relationships. Soon after, network databases, such as CODASYL (Conference on Data Systems Languages), were introduced, enabling more complex relationships between data elements.

The introduction of the relational model revolutionized the database landscape. In 1970, Edgar Codd proposed the concept of a relational database, which represented data as sets of tables with rows and columns. The Structured Query Language (SQL) was developed as a standardized language to interact with relational databases. Oracle, IBM's DB2, and Microsoft SQL Server were some of the early relational database management systems (RDBMS) that gained popularity.

Object-oriented databases (OODBMS) emerged in the 1980s, driven by the need to store and manipulate complex data structures used in object-oriented programming. OODBMS represented data as objects, allowing for the storage of both data and associated behaviors. However, despite their potential, OODBMS adoption remained limited due to challenges in standardization and compatibility.

To bridge the gap between relational databases and object-oriented programming, relational-object mapping frameworks and tools were developed. These frameworks, such as Java Database Connectivity (JDBC) and Object-Relational Mapping (ORM) tools like Hibernate, allowed developers to map objects to relational database tables and perform operations on them.

#### 1.4.1.2. XML

In 1997, a significant development occurred with the introduction of the Extensible Markup Language (XML). XML is a markup language that establishes a set of rules for encoding documents in a format that is readable by both humans and machines. The XML 1.0 Specification, produced by the World Wide Web Consortium (W3C), along with other related specifications, defined XML as a gratis open standard. The design principles of XML prioritize simplicity, generality, and usability, particularly over the Internet. It is a text-based data format that provides extensive support, through Unicode, for languages worldwide.

While XML's design initially focused on documents, it has become widely utilized for representing diverse data structures, including web services. Numerous application programming interfaces (APIs) have been developed to enable software developers to process XML data effectively. Additionally, various XML schema systems exist to facilitate the definition of XML-based languages, enhancing the interoperability and structure of XML data.

#### 1.4.1.3. Relational DATABASES

While databases have traditionally stored and retrieved data, correlations and decisions based on the data have often required human interaction or external programs. However, advancements in database technology have aimed to address this limitation.

In the past, databases primarily focused on enforcing data integrity by controlling the types of data that could be inserted into specific fields. For instance, if a field was designated for storing date and time values, any other type of data would trigger an error message. Over time, databases have become more advanced and introduced additional features such as triggers, cascading updates, and deletes. These features prevent inconsistencies between tables and help maintain data integrity during updates.

Databases have also incorporated simplified procedural languages that include embedded SQL, along with looping and control structures. Examples of these languages include Sybase/SQL Servers' Transact-SQL, Oracle's PL/SQL, and PostgreSQL. These languages provide a means for implementing more complex business logic within the database itself, reducing the reliance on external programs for data manipulation and decision-making.

#### 1.4.1.4. NoSQL and Graph data

Carlo Strozzi's use of the term "NoSQL" in 1998 referred to his lightweight, open-source relational database that deviated from the standard SQL interface. However, it was Eric Evans, a Rackspace employee, who reintroduced the term "NoSQL" later on.

The term "NoSQL" aimed to label the emergence of a growing number of non-relational, distributed data stores that often did not prioritize providing ACID guarantees. ACID stands for Atomicity, Consistency, Isolation, and Durability, which are key attributes of traditional relational database systems like Sybase, IBM DB2, MySQL, Microsoft SQL Server, PostgreSQL, Oracle RDBMS, and Informix.

With the explosion of internet and social media, new data storage and processing needs emerged. NoSQL (Not Only SQL) databases emerged as alternatives to traditional relational databases, offering scalability, flexibility, and faster performance for handling large volumes of unstructured or semi-structured data. **Types of NoSQL databases include key-value stores, document stores, column stores, and graph databases.** Graph databases gained prominence for handling highly connected data, such as social networks and recommendation systems. Graph databases use graph structures to represent and query relationships between data elements efficiently. These databases gained prominence alongside major internet companies like Google, Amazon, Twitter, and Facebook, which faced unique challenges in handling vast amounts of data that traditional relational databases struggled to cope with.

#### 1.4.1.5. Big DATA

As the volume of data continues to grow exponentially, databases are facing new challenges in handling and analyzing large datasets. Systems with terabytes of data, such as those used in scientific projects like the genome project, geological research, national security, and space exploration, require novel approaches for data management and analysis. Techniques like data mining, data warehousing, and data marts have emerged as common practices for extracting valuable insights from vast datasets. Additionally, the rise of big data necessitated the development of technologies like Apache Hadoop and Apache Spark, which provided distributed processing capabilities for handling massive datasets.

#### 1.4.1.6. Database as Service

The advent of cloud computing brought about cloud databases, which offered scalable and on-demand data storage and processing capabilities. Database-as-a-Service (DBaaS) providers, such as Amazon

RDS and Google Cloud Spanner, offered managed database solutions, eliminating the need for organizations to handle database administration and infrastructure management.

In recent times, databases are undergoing further advancements and incorporating more complex logic. One notable feature that is expanding is the concept of separating the physical location from the abstract concept of the database itself, as originally defined by Edgar F. Codd. This feature allows a database to be distributed across multiple locations and queried as a unified unit. Such instances are referred to as distributed or federated databases.

#### 1.4.1.7. Distributed Database

With the introduction of distributed databases, different portions of a database can be physically located in separate geographical locations. For example, one part of a database can reside in New York while another part is in Mumbai. Queries that involve data from multiple locations can be executed concurrently at both locations, providing a seamless and unified view of the data.

The concept of distributed databases has become feasible due to advancements in network speed and reliability. The increased speed of networks enables efficient communication and data transfer between distributed locations, allowing for real-time or near-real-time access to data regardless of its physical location.

By leveraging distributed databases, organizations can achieve improved scalability, fault tolerance, and load balancing across multiple locations. This approach facilitates efficient data processing and analysis, particularly in scenarios where data is geographically dispersed or where high availability and redundancy are essential.

As data storage becomes increasingly complex, various types of specialized data pose unique challenges. Spatial data, for example, requires specific functions and structures to effectively store and query geographical information. This type of data is commonly used in geographic information systems (GIS) and requires spatial indexing techniques to optimize retrieval and analysis.

### 1.4.2. Continuous advancements in the field of Data Storage

Image data and scanned-in data present another set of challenges. Storing and managing large volumes of image files or scanned documents require efficient storage formats and indexing methods to support quick retrieval. Additionally, the processing of image data often involves techniques such as image recognition and computer vision, which require advanced algorithms and computational resources.

In the field of medicine, complex data types like gene sequences are crucial for research and analysis. Storing and processing genetic information demands specialized database structures and algorithms tailored to handle DNA sequences. Furthermore, medical devices that record physical data, such as patient monitoring devices, generate vast amounts of data that need to be efficiently processed and stored for further analysis.

Advancements in data storage and processing capabilities have been essential in meeting these challenges. The development of high-performance storage systems, including solid-state drives (SSDs) and distributed storage solutions, has improved data access speeds and overall performance. Additionally, advancements in computational power and cloud computing enable the processing of large-scale data sets, such as DNA sequences, to derive meaningful patterns and insights.

## 1.5. Machine Learning and AI

Ever since, Humans developed mechanism to store data, we have always tried to extract information from the data. The classification of similar objects into groups is an important human activity. In everyday life, this is part of the learning process: A child learns to distinguish between cats and dogs, between tables and chairs, between men and women, by means of continuously improving subconscious classification schemes.

Beyond classification, another important aspect of understanding data is identifying patterns within it. Humans have an innate tendency to recognize patterns and derive meaning from them. This ability allows us to identify trends, make predictions, and gain insights into complex systems. Recognizing patterns in data has significant implications across various domains, from scientific research to business analysis.

The concept of computers learning and improving automatically has been a long-standing fascination since their invention. Artificial Intelligence is the theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages. **ML is a subfield of Artificial Intelligence (AI) that focuses on the development of algorithms and models that enable computers to learn and make intelligent decisions based on data.**

The process of pattern discovery in ML involves training models on labeled data, where the patterns and relationships between input features and output labels are explicitly provided. Through iterative learning processes, ML models analyze the data, identify relevant patterns, and adjust their internal parameters to improve performance. This learning process allows the models to generalize their knowledge and make accurate predictions on new, unseen data.

ML algorithms employ various techniques to identify patterns, such as statistical analysis, clustering, regression, and neural networks. These techniques enable the models to identify complex relationships and dependencies within the data. ML models can uncover patterns that may not be easily discernible to humans due to the high dimensionality or non-linear nature of the data.

Pattern recognition and understanding in ML have found numerous applications across various domains. For instance, in image and speech recognition, ML models can learn to identify objects, faces, or speech patterns by extracting and analyzing relevant features. In natural language processing (NLP), ML models can discover patterns in text data to perform tasks such as sentiment analysis, language translation, or question-answering. By recognizing patterns in language, AI systems can comprehend and generate human-like responses.

In business and finance, ML algorithms can analyze large datasets to identify market trends, make stock predictions, or detect fraudulent transactions. In healthcare, ML models can analyze patient data to predict disease outcomes, identify risk factors, or assist in medical diagnosis. ML is also extensively used in recommendation systems, where patterns in user preferences and behavior are analyzed to provide personalized recommendations.

The field of machine learning continues to evolve, with advancements in algorithms, models, and computational power. As more data becomes available and computational capabilities increase, the potential applications of machine learning expand across diverse domains, ranging from image and speech recognition to natural language processing and autonomous driving. The continuous development and deployment of machine learning techniques have the potential to drive significant progress in the field of artificial intelligence.

The potential impact of programming computers to learn from experience is immense. Imagine the possibilities of computers learning from medical records to identify effective treatments for new diseases, houses learning to optimize energy costs based on occupants' usage patterns, or personal software assistants learning user interests to deliver highly relevant information. If we could successfully unlock the ability for computers to learn, it would revolutionize computer usage, enabling new levels of customization and competence.

### 1.5.1. History of ML

#### 1.5.1.1. First Neural Network

In the year 1943, Logician Walter Pitts and neuroscientist Warren McCulloch published world's first mathematical model, McCulloch-Pitts neuron, of a neural network to create algorithms that mimic human thought process. The McCulloch-Pitts neuron was inspired by the biological neurons found in the human brain. It introduced the concept of binary threshold units that received inputs and produced outputs based on predefined activation thresholds. The neuron's output would only activate if the summed inputs crossed a specific threshold value. This simple computational model paved the way for the development of more complex neural networks.

Neural networks, inspired by the McCulloch-Pitts model, became an essential component of many machine learning algorithms. Neural networks consist of interconnected nodes or artificial neurons that can learn from data, recognize patterns, and make predictions.

#### 1.5.1.2. World War and Alan Turing

Alan Turing played a significant role in World War II through his work at the Government Code and Cypher School (GC&CS) at Bletchley Park, the British code-breaking center. Turing and his colleagues were tasked with breaking the German Enigma machine's encrypted messages, which were considered to be unbreakable at the time. Turing made significant contributions to the development of machines and techniques for code breaking.

The **Turing Test**, proposed by Alan Turing in 1950, is a significant milestone in the field of artificial intelligence. It serves as a benchmark for evaluating the ability of a machine to exhibit intelligent behavior indistinguishable from that of a human. The test involves a human evaluator interacting with a machine and a human through a computer interface. The evaluator's task is to converse with both the machine and the human, asking them questions and receiving responses. If the evaluator cannot consistently distinguish between the machine and the human based on their answers, the machine is considered to have passed the Turing Test.

While the Turing Test has been influential in stimulating research and discussions around artificial intelligence, it has also faced criticism. Some argue that the test is subjective and merely measures the machine's ability to imitate human behavior rather than demonstrating true intelligence.

#### 1.5.1.3. A Game of Checkers

In 1952, Arthur Samuel developed a computer program that played checkers at a championship level, marking an important milestone in the application of artificial intelligence techniques to gaming. Samuel's program employed various techniques, including the use of pattern recognition and a self-learning algorithm. It was one of the first instances of a computer program that could improve its performance through experience. The program would play numerous games against itself, gradually refining its strategy based on the outcomes and learning from its mistakes.

One of the key techniques used in Samuel's program was **alpha-beta pruning**, which is an algorithmic approach for minimizing the number of game positions that need to be evaluated. This technique allows for more efficient and effective decision-making by exploring only the most promising branches of the game tree.

Additionally, Samuel developed the **minimax algorithm**, a fundamental concept in game theory and artificial intelligence. The minimax algorithm is used to determine the best possible move for a player in a game by considering the potential outcomes and minimizing the maximum possible loss.

#### 1.5.1.4. Perceptron: A Major Breakthrough

In 1957, Frank Rosenblatt, a psychologist and computer scientist, introduced the perceptron, which was a groundbreaking development in the realm of artificial neural networks. The perceptron is a type of algorithm that simulates the functioning of a biological neuron. It was inspired by the structure and functionality of the human brain. The perceptron consists of input nodes, weights assigned to each input, an activation function, and an output node. It takes in input data, processes it through the activation function, and produces a binary output.

The key innovation of the perceptron was its ability to learn from training data through a process called supervised learning. It adjusts the weights assigned to each input based on the observed outcomes, with the aim of improving the accuracy of its predictions. This process allows the perceptron to iteratively update its parameters until it achieves a desired level of accuracy.

Rosenblatt's perceptron algorithm gained attention for its ability to classify patterns and make predictions, and it played a significant role in the development of artificial neural networks and machine learning. However, it should be noted that perceptron has limitations in handling complex problems that are not linearly separable. This limitation led to the perception that perceptron was not capable of addressing more complex real-world problems.

#### 1.5.1.5. Nearest Neighbor

In 1967, the article "Nearest neighbor pattern classification" by Cover and Hart introduced the concept of the nearest neighbor algorithm for pattern classification. This algorithm is widely used in machine learning and serves as a foundation for various classification tasks.

The nearest neighbor algorithm is an instance-based learning method that classifies an input object by comparing it to its nearest neighbors in the training dataset. The basic idea is to assign the same class label to the input object as its closest neighbors in the feature space. In other words, the algorithm determines the class membership of an object based on the classes of its neighboring instances. To classify a new object using the nearest neighbor algorithm, the distances between the new object and all the instances in the training set are computed. The object is then assigned the class label that is most common among its k nearest neighbors, where k is a user-defined parameter representing the number of neighbors to consider. The distance metric used can vary, with popular choices being Euclidean distance or Manhattan distance.

#### 1.5.1.6. Backpropagation: The Core of Deep Learning

In 1974, Paul Werbos introduced the concept of backpropagation in his doctoral dissertation titled "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences." Backpropagation is a fundamental algorithm for training artificial neural networks. Backpropagation is a gradient-based optimization algorithm that enables neural networks to learn from labeled training data. It involves two main steps: forward propagation and backward propagation. During forward propagation, the input data is fed through the network, and the outputs are computed layer by layer. Then, during backward propagation, the error between the predicted outputs and the true labels is calculated and propagated backward through the network. The weights of the network are adjusted based on the computed error gradients, aiming to minimize the prediction errors.

The key idea behind backpropagation is to iteratively update the weights of the neural network using gradient descent. By calculating the gradients of the error with respect to the weights, the algorithm determines how much each weight contributes to the overall error and adjusts them accordingly. This iterative process continues until the network converges to a point where the error is minimized, and the network can make accurate predictions on unseen data.

#### 1.5.1.7. Power of Computation in 21st century

With time, computing power became more affordable and rise of Machine Learning was exponential.

#### 1.5.1.8. 1997: A Machine Defeats a Man in Chess

In 1997, IBM's Deep Blue defeated world chess champion Garry Kasparov in a six-game match. This event showcased the potential of computers to outperform humans in complex tasks requiring strategic thinking and decision-making.

#### 1.5.1.9. 2002: Software Library Torch

The development of Torch, a scientific computing framework with wide ML capabilities, in 2002, marked a significant milestone. Torch provided a powerful and flexible environment for researchers and practitioners to develop and deploy ML models.

#### 1.5.1.10. 2006: Geoffrey Hinton, the father of Deep Learning

Geoffrey Hinton, a prominent figure in the field of ML, made significant contributions to the advancement of deep learning. In 2006, Hinton's research on deep neural networks and the introduction of the "deep belief network" demonstrated the potential of deep learning models to learn hierarchical representations of data.

#### 1.5.1.11. 2011: Google Brain

The creation of Google Brain, a deep learning research project, in 2011, was a pivotal moment for ML. Google Brain employed large-scale neural networks to train models on massive datasets using distributed computing infrastructure.

#### 1.5.1.12. 2014: DeepFace

DeepFace, developed by researchers at Facebook in 2014, demonstrated remarkable progress in facial recognition technology. The deep learning-based model achieved near-human-level accuracy in identifying faces in photographs.

#### 1.5.1.13. 2017: ImageNet Challenge

The ImageNet Challenge is an annual competition that focuses on object detection and image classification tasks using large-scale visual recognition datasets. The challenge involves training models on a massive dataset called ImageNet, which contains millions of labeled images belonging to thousands of different categories. The goal is to develop models that can accurately classify and identify objects within images. In 2017, the ImageNet Challenge saw a remarkable performance from participating teams, with 29 out of 38 teams achieving an impressive 95% accuracy with their computer vision models.

Traditionally, computer vision tasks such as image recognition were approached using handcrafted features and traditional machine learning algorithms. However, the introduction of deep learning and convolutional neural networks (CNNs) revolutionized the field. These deep learning models are capable of automatically learning hierarchical representations from raw image data, leading to improved accuracy and performance.

#### 1.5.1.14. 2020 & beyond

At the time of writing of this book, Generative AI is taking giant leaps in the field of computer science. **Generative AI**, also known as **Generative Adversarial Networks (GANs)**, is a subset of artificial intelligence (AI) that focuses on generating new content, such as images, text, music, or even videos, that is not directly sourced from existing data. It involves training models to understand and mimic the patterns and characteristics of the input data and then generate new samples that resemble the original data.

In Generative AI, two key components work together: the generator and the discriminator. The generator's role is to create new content based on random noise or a given input. The discriminator, on the other hand, tries to distinguish between the generated content and real examples from the original dataset. Through an iterative process, both components continuously improve their performance. Many players like Azure OpenAI and Google Bard have come to the picture with their Large Language Models, providing API to connect and train data, even finetune custom models.

## 1.6. Summary

In this chapter, we discussed about three major applications of Computer science: **Programming**, **Data** and **Machine Learning**. We observed their interconnectivity. Apart from computational abilities, Programs are used to develop software applications which are used by the customer like us. Software Applications retain data of customers to give them a custom user experience. This analysis of behavioral aspect of a customer is extracted from data using Machine Learning, a subset of Artificial intelligence. With time, amount of data being captured and generated has grown significantly, but so has grown the ability of computers to extract information from these dataset, thanks to powerful computational powers and advancements in the field of Data Science and ML. Information extracted is further used by Software Companies to enhance experience of their customers on their platforms. However, the use case of Machine Learning is not just limited to enhancing customer experience, but goes way beyond towards solving complex equations, pattern recognition, speech, and text generation, etc.

Now that we have the context, we shall be discussing Programming, Databases and Machine Learning in separate sections and at the right juncture we shall be discussing their potential convergence in the upcoming chapters.