Machine Learning Basics

1. The first general purpose computer designed by Charles Babbage in the 1800s.
2. In fact, once computers started evolving with the invention of the Analytical Engine by Babbage and the first computer program, which was written by Ada Lovelace in 1842
3. Why do we need the machine learning?

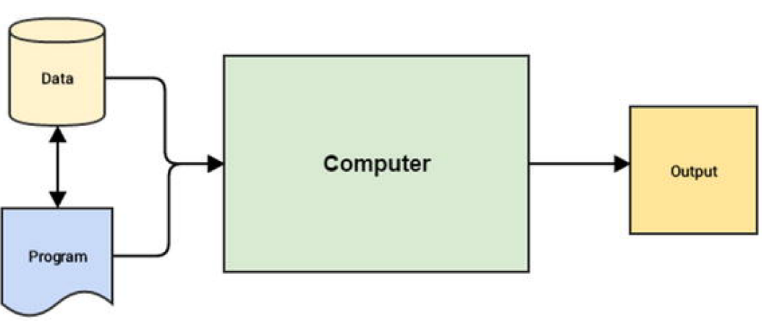
Solution: The answer can be summed up to a simple sentence, “To make data-driven decisions at scale”

1. The art and science of leveraging your data to get actionable insights and make better decisions is known as making data-driven decisions.
2. Of course, this is easier said than done because rarely can we directly use raw data to make any insightful decisions.
3. Efficiency and scale

🡪While getting insights and making decisions driven by data are of paramount importance, it also needs to be done with efficiency and at scale.

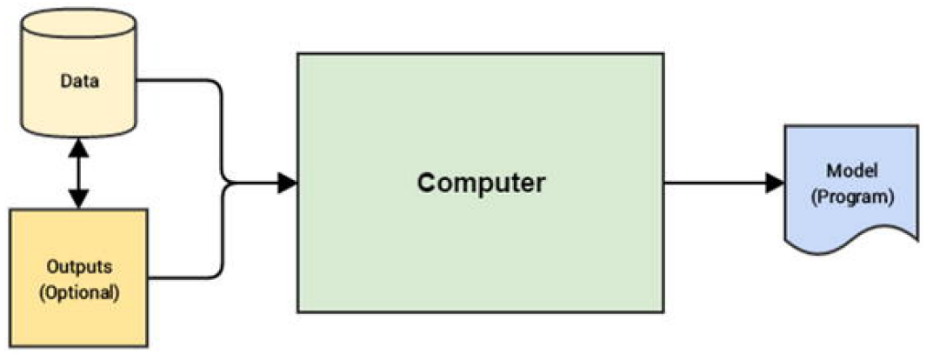
🡪The key idea of using techniques from Machine Learning or artificial intelligence is to automate processes or tasks by learning specific patterns from the data.

🡪Traditional programming paradigms basically involve the user or programmer to write a set of instructions or operations using code that makes the computer perform specific computations on data to give the desired results. Figure below depicts a typical workflow for traditional programming paradigms.



1. Why Machine Learning?

🡪 Considering what you have learned so far, while the traditional programming paradigm is quite good and human intelligence and domain expertise is definitely an important factor in making datadriven decisions, we need Machine Learning to make faster and better decisions. The Machine Learning paradigm tries to take into account data and expected outputs or results if any and uses the computer to build the program, which is also known as a model. This program or model can then be used in the future to make necessary decisions and give expected outputs from new inputs. Figure below shows how the Machine Learning paradigm is similar yet different from traditional programming paradigms.



🡪Figure above reinforces the fact that in the Machine Learning paradigm, the machine, in this context the computer, tries to use input data and expected outputs to try to learn inherent patterns in the data that would ultimately help in building a model analogous to a computer program, which would help in making data-driven decisions in the future (predict or tell us the output) for new input data points by using the learned knowledge from previous data points (its knowledge or experience).

🡪 You might start to

see the benefit in this. We would not need hand-coded rules, complex

flowcharts, case and if-then conditions, and other criteria that are typically

used to build any decision making system or a decision support system. The

basic idea is to use Machine Learning to make insightful decisions.

🡪 The beauty of Machine Learning is that it is never domain

constrained and you can use techniques to solve problems spanning

multiple domains, businesses, and industries.

1. Understanding machine learning

🡪 Machine Learning came

into prominence perhaps in the 1990s when researchers and scientists started giving it more prominence as a sub-field of Artificial Intelligence (AI) such that techniques borrow concepts from AI, probability, and statistics, which perform far better compared to using fixed rule-based models requiring a lot of manual time and effort.

🡪 You could say that it started off in the late 1700s and the early 1800s when the first works of research were published which basically talked about the Bayes’ Theorem. In fact Thomas Bayes’ major work, “An Essay Towards Solving a Problem in the Doctrine

of Chances,” was published in 1763. Besides this, a lot of research and discovery was done during this time in the field of probability and mathematics. This paved the way for more ground breaking research and inventions in the 20th Century, which included Markov Chains by Andrey Markov in the early 1900s, proposition of a learning system by Alan Turing, and the invention of the very famous perceptron by Frank Rosenblatt in the 1950s. Many of you might know that neural networks had several highs and lows since the 1950s and they finally came back to prominence in the 1980s with the discovery of backpropagation (thanks to Rumelhart, Hinton, and Williams!) and several other inventions, including

Hopfield networks, neocognition, convolutional and recurrent neural networks, and Q-learning. Of course, rapid strides of evolution started taking place in Machine Learning too since the 1990s with the discovery of random forests, support vector machines, long short-term memory networks (LSTMs), and development and release of frameworks in both machine and

Deep Learning including torch, theano, tensorflow, scikitlearn,

and so on. We also saw the rise of intelligent systems including

IBM Watson, DeepFace, and AlphaGo.

\*IBM Waston is IBM’s portfolio of business-ready tools, applications and solutions, designed to reduce the costs and hurdles of AI adoption while optimizing outcomes and responsible use of AI.

\*DeepFace is a deep learning facial recognition system created by a research group at Facebook. It identifies human faces in digital images

\*AlphaGo is most challenging classical game for artificial intelligence because of its complexity. Despite decades of work, the strongest Go computer programs could only play at the level of human amateurs

1. Why make machines learn

🡪We as humans and domain experts already have enough knowledge about the world and our respective domains, which can be objective, subjective, and sometimes even intuitive. With the availability of large volumes of historical data, we can leverage the Machine Learning paradigm to make machines perform specific tasks by gaining enough experience by observing patterns in data over a period of time and then use this experience in solving tasks in the future with minimal manual intervention. The core idea remains to make machines solve tasks that can be easily defined intuitively and almost involuntarily but extremely hard to define formally.

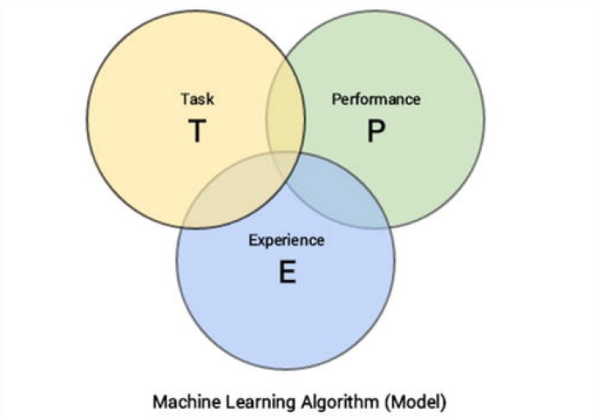
🡺Formal Definition

🡪The best way to define Machine Learning would be to start from the basics of Machine Learning as defined by renowned professor Tom Mitchell in 1997.

🡪The idea of Machine Learning is that there will be some learning algorithm that will help the machine learn from data. Professor Mitchell defined it as follows:

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

🡪While this definition might seem daunting at first, I ask you go read through it a couple of times slowly focusing on the three parameters —T, P, and E—which are the main components of any learning algorithm, as depicted in Figure



🡪We can simplify the definition as follows. Machine Learning is a field that consists of learning algorithms that:

1. Improve their performance P
2. At executing some task T
3. Over time with experience E

🡺Defining the task, T

🡪Machine Learning based tasks are difficult to solve by conventional and traditional programming approaches.

🡪The typical tasks that could be classified as Machine

Learning tasks, the following list describes some popular tasks.

1. Classification or categorization : This typically encompasses the list of problems or tasks where the machine has to take in data points or samples and assign a specific class or category to each sample. A simple example would be classifying animal images into dogs, cats, and zebras.
2. Regression : These types of tasks usually involve performing a prediction such that a real numerical value is the output instead of a class or category for an input data point. The best way to understand a regression task would be to take the case of a real-world problem of predicting housing prices considering the plot area, number of floors, bathrooms, bedrooms, and kitchen as input attributes for each data point.
3. Anomaly detection : These tasks involve the machine going over event logs, transaction logs, and other data points such that it can find anomalous or unusual patterns or events that are different from the normal behaviour. Examples for this include trying to find denial of service attacks from logs, indications of fraud, and so on.
4. Structured annotation : This usually involves performing some analysis on input data points and adding structured metadata as annotations to the original data that depict extra information and relationships among the data elements. Simple examples would be annotating text with their parts of speech, named entities, grammar, and sentiment. Annotations can also be done for images like assigning specific categories to image pixels, annotate specific areas of images based on their type, location, and so on.
5. Translation : Automated machine translation tasks are typically of the nature such that if you have input data samples belonging to a specific language, you translate it into output having another desired language. Natural language based translation is definitely a huge area dealing with a lot of text data.
6. Clustering or grouping : Clusters or groups are usually formed from input data samples by making the machine learn or observe inherent latent patterns, relationships and similarities among the input data points themselves. Usually there is a lack of pre-labelled or preannotated data for these tasks hence they form a part of unsupervised Machine Learning. Examples would be grouping similar products, events and entities.
7. Transcriptions : These tasks usually entail various representations of data that are usually continuous and unstructured and converting them into more structured and discrete data elements. Examples includes peech to text, optical character recognition, images to text, and so on.

🡪This should give you a good idea of typical tasks that are often solved using Machine Learning, but this list is definitely not an exhaustive one as the limits of tasks are indeed endless and more are being discovered with

extensive research over time

🡺Defining the Experience, E

🡪The process of consuming a dataset that consists of data samples or data points such that a learning algorithm or model learns inherent patterns is defined as the experience, E which is gained by the learning algorithm.

🡪Any experience that the algorithm gains is from data samples or data points and this can be at any point of time.

🡪The idea of a model or algorithm gaining experience usually occurs as an iterative process, also known as training the model.

🡪You could think of the model to be an entity just like a human being which gains knowledge or experience through data points by observing and learning more and more about various attributes, relationships and patterns present in the data.

🡪Of course, there are various forms and ways of learning and gaining experience including supervised, unsupervised, and reinforcement learning

🡺Defining the performance, P

🡪Let’s say we have a Machine Learning algorithm that is supposed to perform a task, T, and is gaining experience, E, with data points over a period of time. But how do we know if it’s performing well or behaving the way it is supposed to behave? This is where the performance, P, of the model comes into the picture.

🡪The performance, P, is usually a quantitative measure or metric that’s used to see how well the algorithm or model is

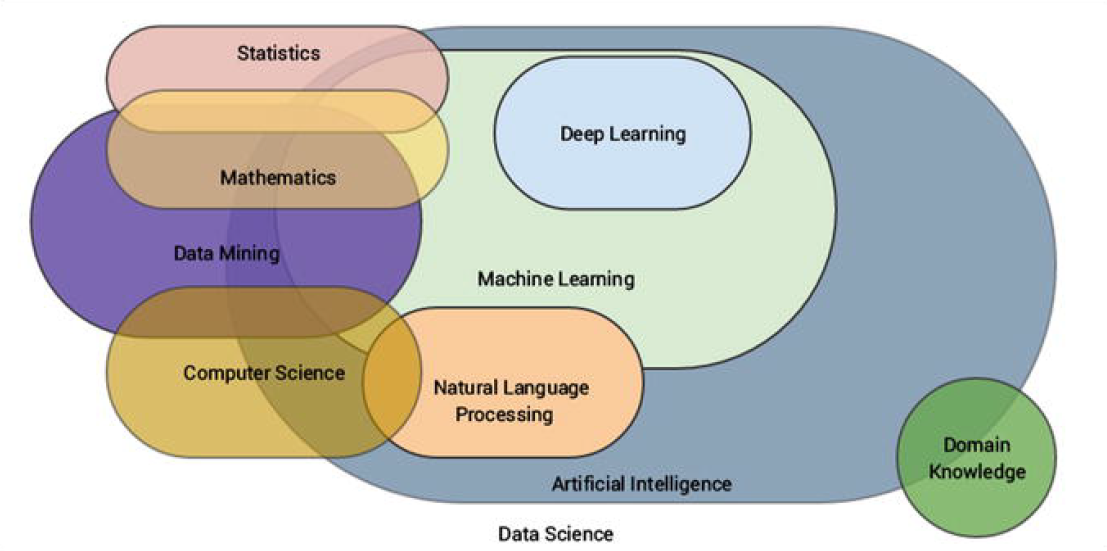
performing the task, T, with experience, E.

🡪Typical performance measures include accuracy, precision, recall, F1 score, sensitivity, specificity, error rate, misclassification rate, and many more. Performance measures are usually evaluated on training data samples (used by the algorithm to gain experience, E) as well as data samples which it has not seen or learned from before, which are usually known as validation and test data samples.

🡪While this is true in most scenarios, you should always remember that sometimes it is difficult to choose performance measures that will accurately be able to give us an idea of how well the algorithm is performing based on the actual behaviour or outcome which is expected from it.

1. A Multi-Disciplinary Field

🡪Figure below should give you a good idea with regard to the major fields that overlap with Machine Learning based on concepts, methodologies, ideas, and techniques.



🡪The major domains or fields associated with Machine Learning include the following, as depicted in Figure. We will discuss each of these fields in upcoming sections.

1. Artificial Intelligence
2. Natural Language Processing
3. Data Mining
4. Mathematics
5. Statistics
6. Computer Science
7. Deep Learning
8. Data Science

🡪Ideas of pattern recognition and basic data mining methodologies like **knowledge discovery of databases (KDD)** came into existence when relational databases were very prominent.

🡪KDD is a whole process by itself that includes data acquisition, storage, warehousing, processing, and analysis.

🡺Deep Learning is a subfield of Machine Learning itself which deals more with techniques related to representational learning such that it improves with more and more data by gaining more experience. It follows a layered and hierarchical approach such that it tries to represent the given input attributes and its current surroundings, using a nested layered hierarchy of concept representations such that, **each complex layer is built from another layer of simpler concepts**. **Neural networks are something which is heavily utilized by Deep Learning.**

**🡺Computer Science**

🡪The field of computer science (CS) can be defined as the study of the science of understanding computers. This involves study, research, development, engineering, and experimentation of areas dealing with understanding, designing, building, and using computers. This also involves

extensive design and development of algorithms and programs that can be used to make the computer perform computations and tasks as desired. There are mainly two major areas or fields under computer science, as follows.

* Theoretical Computer Science
* Applied or Practical Computer Science

🡪The main essence of computer science includes formal languages, automata and theory of computation, algorithms, data structures, computer design and architecture, programming languages, and software engineering principles.

\*An automata is **an abstract self-propelled computing device that follows a predetermined sequence of operations automatically**. An automaton with a finite number of states is called a Finite Automaton (FA) or Finite-State Machine (FSM).

**🡺Theoretical Computer Science**

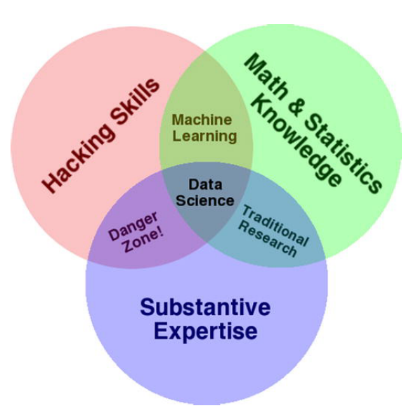
🡪Theoretical computer science is the study of theory and logic that tries to explain the principles and processes behind computation. This involves understanding the theory of computation which talks about how computation can be used efficiently to solve problems. Theory of computation includes the study of formal languages, automata, and understanding complexities involved in computations and algorithms. Information and coding theory is another major field under theoretical CS that has given us domains like signal processing, cryptography, and data compression. Principles of programming languages and their analysis is another important aspect that talks about features, design, analysis, and implementations of various programming languages and how compilers and interpreters work in understanding these languages. Last but never the least, data structures and algorithms are the two fundamental pillars of theoretical CS used extensively in computational programs and functions.

**🡺Practical Computer Science**

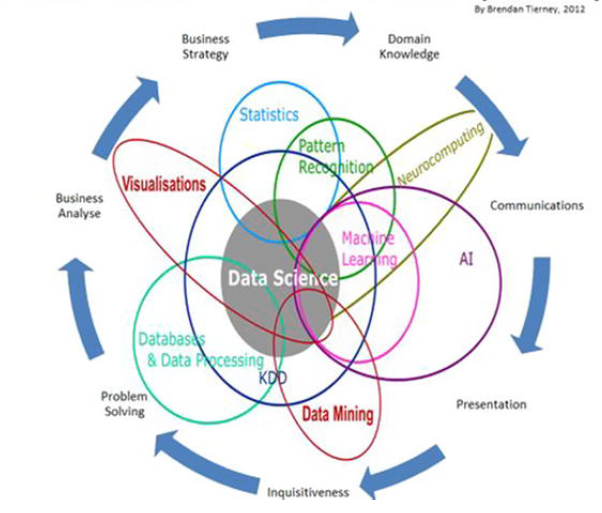
🡪Practical computer science also known as applied computer science is more about tools, methodologies, and processes that deal with applying concepts and principles from computer science in the real world to solve practical day-to-day problems. This includes emerging sub-fields like artificial intelligence, Machine Learning, computer vision, Deep Learning, natural language processing, data mining, and robotics and they try to solve complex real-world problems based on multiple constraints and parameters and try to emulate tasks that require considerable human intelligence and experience. Besides these, we also have well-established fields, including computer architecture, operating systems, digital logic and design, distributed computing, computer networks, security, databases, and software engineering.

**🡺Data Science**

🡪The field of Data Science is a very diverse, inter-disciplinary field which encompasses multiple fields. Data Science basically deals with principles, methodologies, processes, tools, and techniques to gather knowledge or information from data (structured as well as unstructured). Data Science is more of a compilation of processes, techniques, and methodologies to foster a data-driven decision based culture. In fact Drew Conway’s “Data Science Venn Diagram,” depicted in Figure below, shows the core components and essence of Data Science, which in fact went viral and became insanely popular!



🡪Besides this, we also have Brendan Tierney, who talks about the true nature of Data Science being a mult-disciplinary field with his own depiction, as shown in Figure



**🡺Important Concepts Of Mathematics**

**🡺Scalar** : A scalar usually denotes a single number as opposed to a collection of numbers. A simple example might be x = 5 or x ∈ R, where x is the scalar element pointing to a single number or a real-valued single number.

**🡺Vector** : A vector is defined as a structure that holds an array of numbers which are arranged in order. This basically means the order or sequence of numbers in the collection is important. Vectors can be mathematically denoted as x = [x

1, x 2, …, x n ], which basically tells us that x is a one-dimensional vector having n elements in the array. Each element can be referred to using an array index determining its position in the vector.

**🡺Matrix** : A matrix is a two-dimensional structure that basically holds numbers. It’s also often referred to as a 2D array . Each element can be referred to using a row and column index as compared to a single vector index in case of vectors. Mathematically, you can depict a matrix as

such that M is a 3 x 3 matrix having three rows and three columns and each element is denoted by m rc such that r denotes the row index and c denotes

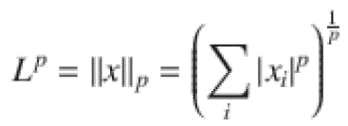
the column index.

**🡺Tensor** : You can think of a tensor as a generic array. Tensors are basically arrays with a variable number of axes. An element in a three-dimensional tensor T can be denoted by T x,y,z where x, y, z denote the three axes for specifying

element T.

**🡺Norm** : The norm is a measure that is used to compute the size of a vector often also defined as the measure of distance from the origin to the point denoted by the vector. Mathematically, the pth norm of a vector is denoted as

follows.



Such that p ≥ 1 and p ∈ R. Popular norms in Machine Learning include the L1 norm used extensively in Lasso regression models and the L2 norm, also known as the Euclidean norm, used in ridge regression models.

**🡺Eigen Decomposition** : This is basically a matrix decomposition process such that we decompose or break down a matrix into a set of eigen vectors and eigen values. The eigen decomposition of a matrix can be mathematically denoted by M = V diag(λ) V ^(-1) such that the matrix M has a total of n linearly independent eigen vectors represented as {v (1), v (2), …, v (n)} and their corresponding eigen values can be represented as {λ 1, λ 2, …, λ n }. The matrix V consists of one eigen vector per column of the matrix i.e., V = [v (1), v (2), …, v (n)] and the vector λ consists of all the eigen values together i.e., λ = [λ 1, λ 2, …, λ n ].

🡪An eigen vector of the matrix is defined as a non-zero vector such that on multiplying the matrix by the eigen vector, the result only changes the scale of the eigen vector itself, i.e., the result is a scalar multiplied by the eigen vector. This scalar is known as the eigen value corresponding to the eigen vector. Mathematically this can be denoted by Mv = λv where M is our matrix, v is the eigen vector and λ is the corresponding eigen value.

🡺**Singular Value Decomposition** : The process of singular value decomposition, also known as SVD, is another matrix decomposition or factorization process such that we are able to break down a matrix to obtain singular vectors and singular values. Any real matrix will always be decomposed by SVD even if eigen decomposition may not be applicable in some cases. Mathematically, SVD can be defined as follows. Considering a matrix M having dimensions m x n

such that m denotes total rows and n denotes total columns, the SVD of the matrix can be represented with the following equation.



🡪This gives us the following main components of the decomposition equation.

* U m x m is an m x m unitary matrix where each column represents a left singular vector
* S m x n is an m x n matrix with positive numbers on the diagonal, which can also be represented as a vector of the singular values
* V ^(T)n x n is an n x n unitary matrix where each row represents a right singular vector

🡪In some representations, the rows and columns might be interchanged but the end result should be the same, i.e., U and V are always orthogonal.

🡺**Random Variable** : Used frequently in probability and uncertainty measurement, a random variable is basically a variable that can take on various values at random. These variables can be of discrete or continuous type in general.

**🡺Probability Distribution** : A probability distribution is a distribution or arrangement that depicts the likelihood of a random variable or variables to take on each of its probable states. There are usually two main types of distributions based on the variable being discrete or continuous.

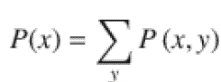
**🡺Probability Mass Function** : A probability mass function , also known as PMF, is a probability distribution over discrete random variables. Popular examples include the

Poisson and binomial distributions.

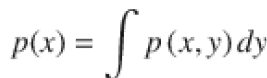
**🡺Probability Density Function** : A probability density function , also known as PDF, is a probability distribution over continuous random variables. Popular examples include the normal, uniform, and student’s T distributions.

**🡺Marginal Probability** : The marginal probability rule is used when we already have the probability distribution for a set of random variables and we want to compute the

probability distribution for a subset of these random variables. For discrete random variables, we can define marginal probability as follows.

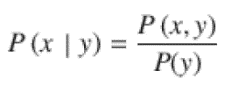


🡪For continuous random variables, we can define it using the integration operation as follows.



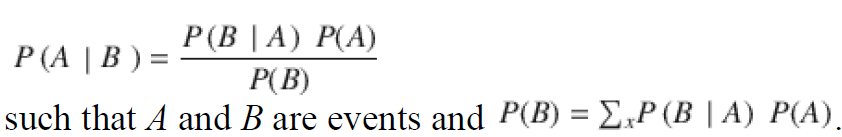
**🡺Conditional Probability** : The conditional probability rule is used when we want to determine the probability that an event is going to take place, such that another event has

already taken place. This is mathematically represented as follows.



🡪This tells us the conditional probability of x, given that y has already taken place.

**🡺Bayes Theorem** : This is another rule or theorem which is useful when we know the probability of an event of interest P(A), the conditional probability for another event based on our event of interest P(B | A) and we want to determine the conditional probability of our event of interest given the other event has taken place P(A | B). This can be defined mathematically using the following expression.



**🡺Statistics** : The field of statistics can be defined as a specialized branch of mathematics that consists of frameworks and methodologies to collect, organize, analyze, interpret, and present data. Generally this falls more under applied mathematics and borrows concepts from linear algebra, distributions, probability theory, and inferential methodologies. There are two major areas

under statistics that are mentioned as follows.

* Descriptive Statistics
* Inferential Statistics

🡪 The core component of any statistical process is data. Hence typically data collection is done first, which could be in global terms, often called a population or a more restricted subset due to various constraints often knows as a sample. Samples are usually collected manually, from surveys, experiments, data stores, and observational studies. From this data, various analyses are carried out using statistical methods.

🡪 Descriptive statistics is used to understand basic characteristics of the data using various aggregation and summarization measures to describe and understand the data better. These could be standard measures like mean,

median, mode, skewness, kurtosis, standard deviation, variance, and so on.

🡪 Libraries and frameworks like pandas, scipy, and numpy in general help us compute descriptive statistics and summarize data easily in Python.

🡪 Inferential statistics are used when we want to test hypothesis, draw inferences, and conclusions about various characteristics of our data sample or population. Frameworks and techniques like hypothesis testing,

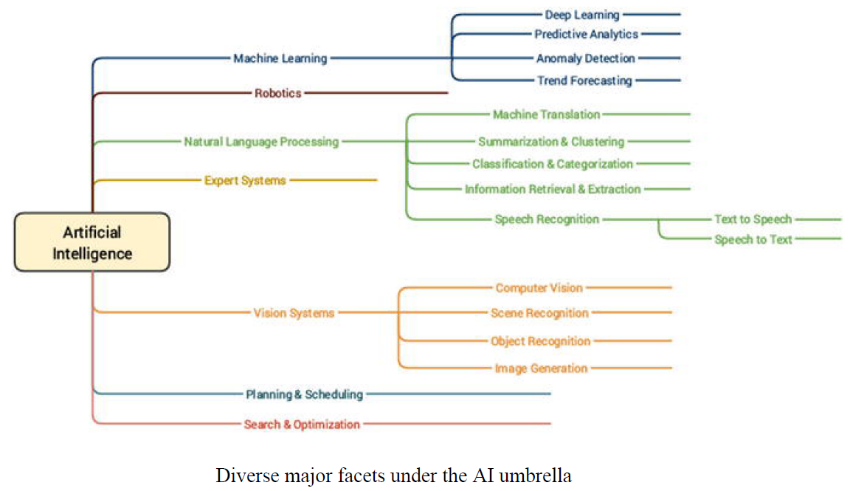
correlation, and regression analysis, forecasting, and predictions are typically used for any form of inferential statistics.

**🡺Data Mining**

🡪 The field of data mining involves processes, methodologies, tools and techniques to discover and extract patterns, knowledge, insights and valuable information from non-trivial datasets. Datasets are defined as nontrivial when they are substantially huge usually available from databases and data warehouses. Once again, data mining itself is a multi-disciplinary field, incorporating concepts and techniques from mathematics, statistics, computer science, databases, Machine Learning and Data Science. The term is a misnomer in general since the “mining” refers to the mining of actual insights or information from the data and not data itself! In the whole process of KDD or Knowledge Discovery in Databases, data mining is the step where all the analysis takes place.

🡪 In general, both KDD as well as data mining are closely linked with Machine Learning since they are all concerned with analyzing data to extract useful patterns and insights. Hence methodologies, concepts, techniques, and processes are shared among them. The standard process for data mining followed in the industry is known as the CRISP-DM model, Which we will see in the up-coming sections.

**🡺Artificial Intelligence**



🡪Artificial Intelligence (AI) is the superset consisting of Machine learning as one of its specialized areas. The basic idea of AI is the study and development of intelligence as exhibited by machines based on their perception of their environment, input parameters and attributes and their response such that they can perform desired tasks based on expectations. AI itself is a truly massive field which is itself inter-disciplinary. It draws on concepts from mathematics, statistics, computer science, cognitive sciences, linguistics, neuroscience, and many more. Machine Learning is more concerned with algorithms and techniques that can be used to understand data, build representations, and perform tasks such as predictions. Another major sub-field under AI related to Machine Learning is natural language processing (NLP) which borrows concepts heavily from computational linguistics and computer science. Text Analytics is a prominent field today among analysts and data scientists to extract, process and understand natural human language. Combine NLP with AI and Machine Learning and you get chatbots, machine translators, and virtual personal assistants, which are indeed the future of innovation and technology!

🡪Some of the main objectives of AI include emulation of cognitive functions also known as cognitive learning, semantics, and knowledge representation, learning, reasoning, problem solving, planning, and natural language processing.

**🡺Natural Language Processing**

🡪The field of Natural Language Processing (NLP) is a multi-disciplinary field combining concepts from computational linguistics, computer science and artificial intelligence.

🡪NLP involves the ability to make machines

process, understand, and interact with natural human languages. The major objective of applications or systems built using NLP is to enable interactions between machines and natural languages that have evolved over time. Major challenges in this aspect include knowledge and semantics

representation, natural language understanding, generation, and processing. Some of the major applications of NLP are mentioned as follows.

* Machine translation
* Speech recognition
* Question answering systems
* Context recognition and resolution
* Text summarization
* Text categorization
* Information extraction
* Sentiment and emotion analysis
* Topic segmentation

🡪Using techniques from NLP and text analytics, you can work on text data to process, annotate, classify, cluster, summarize, extract semantics, determine sentiment, and much more!

**🡺Deep Learning**

🡪The field of Deep Learning, as depicted earlier, is a sub-field of Machine Learning that has recently come into much prominence. Its main objective is to get Machine Learning research closer to its true goal of “making machines intelligent”. Deep Learning is often termed as a rebranded fancy term for neural networks. This is true to some extent but there is definitely more to Deep Learning than just basic neural networks. Deep Learning based algorithms involves the use of concepts from representation learning where various representations of the data are learned in different layers that also aid in automated feature extraction from the data.

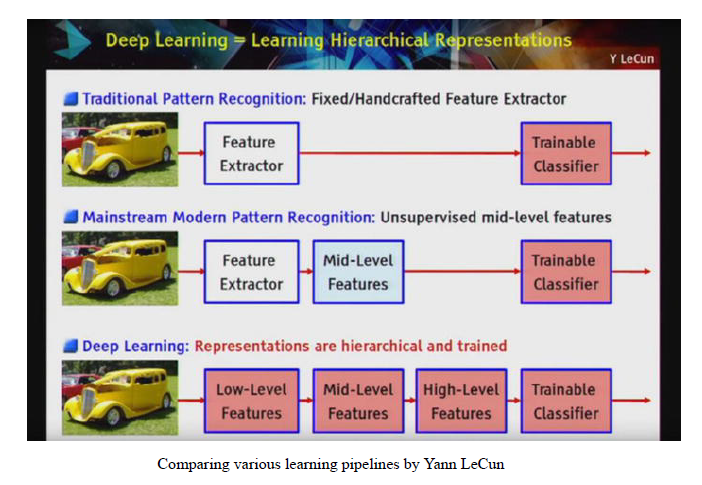
🡪In simple terms, a Deep Learning based approach tries to build machine intelligence by representing data as a layered hierarchy of concepts, where each layer of concepts is built from other simpler layers. This layered architecture itself is

one of the core components of any Deep Learning algorithm.

🡪In any basic **supervised Machine Learning technique**, we basically try to learn a mapping between our data samples and our output and then try to predict output for newer data samples but **Representational learning** tries to understand the representations in the data itself besides learning mapping from inputs to outputs.

🡪There have been several noticeable trends and characteristics related to Deep Learning that we have noticed over the past decade. They are summarized as follows.

* Deep Learning algorithms are based on distributed representational learning and they start performing better with more data over time.
* Deep Learning could be said to be a rebranding of neural networks, but there is a lot into it compared to traditional neural networks.
* Better software frameworks like tensorflow, theano, caffe, mxnet, and keras, coupled with superior hardware have made it possible to build extremely complex, multi-layered Deep Learning models with huge sizes.
* Deep Learning has multiple advantages related to automated feature extraction as well as performing supervised learning operations, which have helped data scientists and engineers solve increasingly complex problems over time.

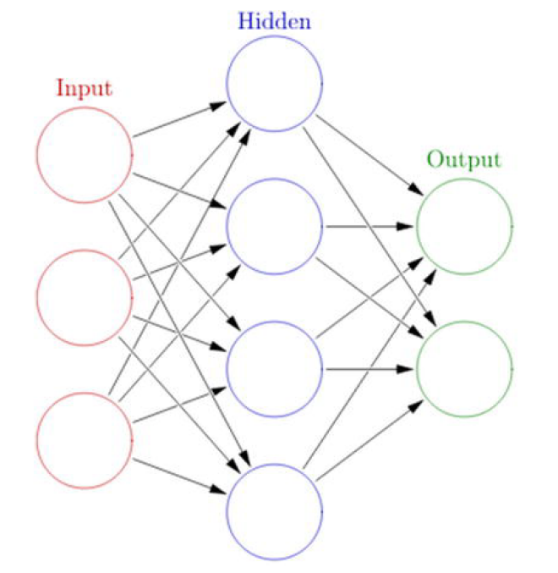


🡪 You can clearly see how Deep Learning methods involve a hierarchical layer representation of features and concept from the raw data as compared to other Machine Learning methods. We conclude this section with a brief coverage of some essential concepts pertaining to Deep Learning.

1. Artificial Neural Networks

🡪 An Artificial Neural Network (ANN) is a computational model and

architecture that simulates biological neurons and the way they function in our brain. Typically, an ANN has layers of interconnected nodes. The nodes and their inter-connections are analogous to the network of neurons in our brain. A typical ANN has an input layer, an output layer, and at least one hidden layer between the input and output with inter-connections, as depicted in Figure below:



🡪 Any basic ANN will always have multiple layers of nodes, specific

connection patterns and links between the layers, connection weights and activation functions for the nodes/neurons that convert weighted inputs to outputs. The process of learning for the network typically involves a cost function and the objective is to optimize the cost function (typically minimize the cost). The weights keep getting updated in the process of learning.

1. Backpropagation

🡪 The backpropagation algorithm is a popular technique to train ANNs and it led to a resurgence in the popularity of neural networks in the 1980s. The algorithm typically has two main stages—propagation and weight updates. They are described briefly as follows.

* 1. Propagation
     1. The input data sample vectors are propagated forward through the neural network to generate the output values from the output layer.
     2. Compare the generated output vector with the actual/desired output vector for that input data vector.
     3. Compute difference in error at the output units.
     4. Backpropagate error values to generate deltas at each node/neuron.
  2. Weight Update

1. Compute weight gradients by multiplying the output delta (error) and input activation.
2. Use learning rate to determine percentage of the gradient to be subtracted from original weight and update the weight of the nodes.

🡪 These two stages are repeated multiple times with multiple iterations/epochs until we get satisfactory results. Typically backpropagation is used along with optimization algorithms or functions

like stochastic gradient descent.

1. Multilayer Perceptrons

🡪 A multilayer perceptron , also known as MLP, is a fully connected, feedforward artificial neural network with at least three layers (input, output, and at least one hidden layer) where each layer is fully connected to the adjacent layer. Each neuron usually is a non-linear functional processing unit. Backpropagation is typically used to train MLPs and even deep neural nets are MLPs when they have multiple hidden layers. Typically used for supervised Machine Learning tasks like classification.

1. Convolutional Neural Networks

🡪 A convolutional neural network, also known as convnet or CNN, is a

variant of the artificial neural network, which specializes in emulating

functionality and behavior of our visual cortex. CNNs typically consist of the following three components.

* Multiple convolutional layers, which consist of multiple filters that are convolved across the height and width of the input data (e.g., image raw pixels) by basically computing a dot product to give a two dimensional activation map. On stacking all the maps across all the filters, we end up getting the final output from a convolutional layer.
* Pooling layers , which are basically layers that perform non-linear down sampling to reduce the input size and number of parameters from the convolutional layer output to generalize the model more, prevent overfitting and reduce computation time. Filters go through the heights and width of the input and reduce it by taking an aggregate like sum, average, or max. Typical pooling components are average or max pooling.
* Fully connected MLPs to perform tasks such as image classification and object recognition.

🡪 A typical CNN architecture with all the components is depicted as

follows in Figure below, which is a LeNet CNN model (Source:

deeplearning.net)

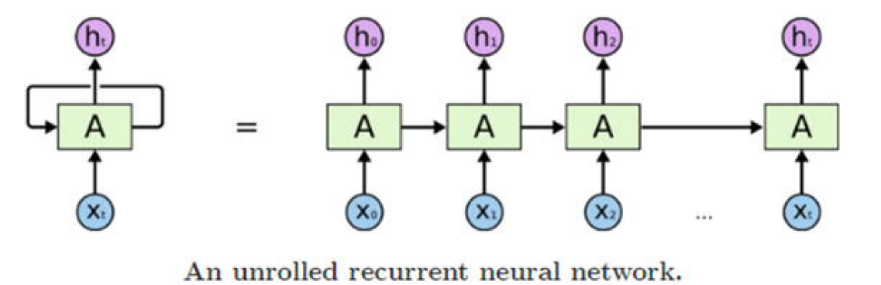


1. Recurrent Neural Networks

🡪 A recurrent neural network, also known as RNN, is a special type of an

artificial neural network that allows persisting information based on past

knowledge by using a special type of looped architecture. They are used a lot in areas related to data with sequences like predicting the next word of a sentence. These looped networks are called recurrent because they perform the same operations and computation for each and every element in a sequence of input data. RNNs have memory that helps in capturing information from past sequences. Figure below shows the typical structure of a RNN and how it works by unrolling the network based on input sequence length to be fed at any point in time.

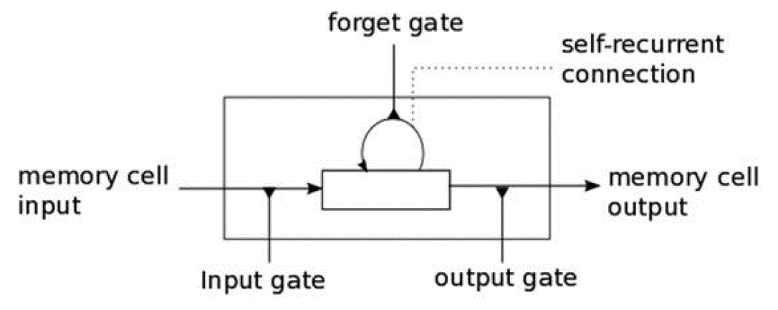


🡪Figure above clearly depicts how the unrolled network will accept

sequences of length t in each pass of the input data and operate on the same.

1. Long Short-Term Memory Networks

🡪 RNNs are good in working on sequence based data but as the sequences start increasing, they start losing historical context over time in the sequence and hence outputs are not always what is desired. This is where Long Short-Term Memory Networks, popularly known as LSTMs, come into the picture! Introduced by Hochreiter & Schmidhuber in 1997, LSTMs can remember information from really long sequence based data and prevent issues like the vanishing gradient problem, which typically occurs in ANNs trained with backpropagation. LSTMs usually consist of three or four gates, including input, output, and a special forget gate. Figure below shows a high-level pictorial representation of a single LSTM cell.



🡪 The input gate usually can allow or deny incoming signals or inputs to

alter the memory cell state. The output gate usually propagates the value to other neurons as needed. The forget gate controls the memory cell’s self recurrent connection to remember or forget previous states as necessary. Multiple LSTM cells are usually stacked in any Deep Learning network to solve real-world problems like sequence prediction.

1. Autoencoders

🡪 An autoencoder is a specialized Artificial Neural Network that is primarily used for performing unsupervised Machine Learning tasks. Its main objective is to learn data representations, approximations, and encodings. Autoencoders can be used for building generative models, performing dimensionality reduction, and detecting anomalies.

1. Machine Learning Methods

🡪 Machine Learning has multiple algorithms, techniques, and methodologies that can be used to build models to solve real-world problems using data. This section tries to classify these Machine Learning methods under some broad categories to give some sense to the overall landscape of Machine Learning methods that are ultimately used to perform specific Machine Learning tasks we discussed in a previous section. Typically the same Machine Learning methods can be classified in multiple ways under multiple umbrellas. Following are some of the major broad areas of Machine Learning methods.

* 1. Methods based on the amount of human supervision in the learning process
     1. Supervised Learning
     2. Unsupervised Learning
     3. Semi-Supervised Learning
     4. Reinforcement Learning
  2. Methods based on the ability to learn from incremental data sample
     1. Batch Learning
     2. Online Learning
  3. Methods based on their approach to generalization from data samples
     1. Instance Based Learning
     2. Model Based Learning

🡪 We briefly cover the various types of learning methods in the following sections to build a good foundation with regard to Machine Learning methods and the type of tasks they usually solve. This should give you enough knowledge to start understanding which methods should be applied in what scenarios when we tackle various real-world use cases and problems.

1. Supervised Learning

🡪 Supervised learning methods or algorithms include learning algorithms that take in data samples (known as training data) and associated outputs (known as labels or responses) with each data sample during the model training process. The main objective is to learn a mapping or association between input data samples x and their corresponding outputs y based on multiple training data instances. This learned knowledge can then be used in the future to predict an output y ′ for any new input data sample x′ which was previously unknown or unseen during the model training process. These methods are termed as supervised because the model learns on data samples where the desired output responses/labels are already known beforehand in the training phase.

🡪 Supervised learning basically tries to model the relationship between the inputs and their corresponding outputs from the training data so that we would be able to predict output responses for new data inputs based on the knowledge it gained earlier with regard to relationships and mappings between the inputs and their target outputs. This is precisely why supervised learning methods are extensively used in predictive analytics where the main objective is to predict some response for some input data that’s typically fed into a trained supervised ML model. **Supervised learning methods are of two major classes based on the type of ML tasks they aim to solve**.

* Classification
* Regression

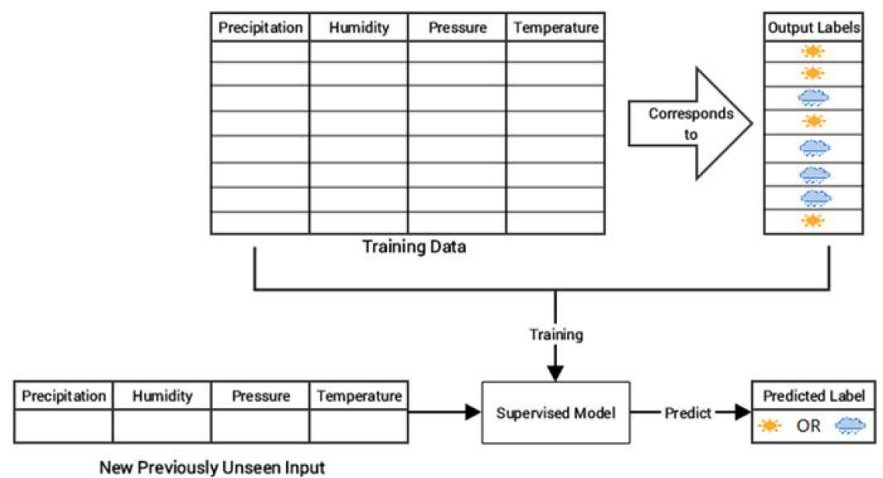
🡪 Let’s look at these two Machine Learning tasks and observe the subset of supervised learning methods that are best suited for tackling these tasks.

1. Classification

🡪 The classification based tasks are a sub-field under supervised Machine Learning, where the key objective is to predict output labels or responses that are categorical in nature for input data based on what the model has learned in the training phase. Output labels here are also known as classes or class labels are these are categorical in nature meaning they are unordered and discrete values. Thus, each output response belongs to a specific discrete class or category.

🡪 Suppose we take a real-world example of predicting the weather. Let’s keep it simple and say we are trying to predict if the weather is sunny or rainy based on multiple input data samples consisting of attributes or features like humidity, temperature, pressure, and precipitation. Since the prediction can be either sunny or rainy, there are a total of two distinct classes in total; hence this problem can also be termed as a binary classification problem. Figure below depicts the binary weather classification

task of predicting weather as either sunny or rainy based on training the supervised model on input data samples having feature vectors, (precipitation, humidity, pressure, and temperature) for each data sample/observation and their corresponding class labels as either sunny or rainy.



🡪 A task where the total number of distinct classes is more than two becomes a multi-class classification problem where each prediction response can be any one of the probable classes from this set. A simple example would be trying to predict numeric digits from scanned handwritten images. In this case it becomes a 10-class classification problem because the output class label for any image can be any digit from 0 - 9. In both the cases, the output class is a scalar value pointing to one specific class. Multi-label classification tasks are such that based on any input data sample, the output response is usually a vector having one or

more than one output class label. A simple real-world problem would be trying to predict the category of a news article that could have multiple output classes like news, finance, politics, and so on.

🡪Popular classification algorithms include logistic regression, support vector machines, neural networks, ensembles like random forests and gradient boosting, K-nearest neighbors, decision trees, and many more.

1. Regression

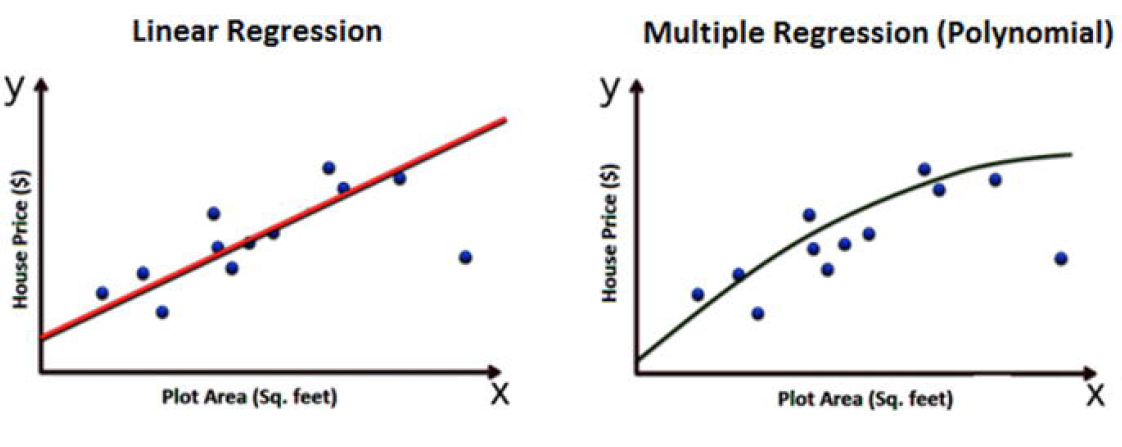
🡪 Machine Learning tasks where the main objective is value estimation can be termed as regression tasks. Regression based methods are trained on input data samples having output responses that are continuous numeric values unlike classification, where we have discrete categories or classes.

🡪 Regression models make use of input data attributes or features (also called explanatory or independent variables) and their corresponding continuous numeric output values (also called as response, dependent, or outcome variable) to learn specific relationships and associations between the inputs and their corresponding outputs. With this knowledge, it can predict output

responses for new, unseen data instances similar to classification but with continuous numeric outputs.

🡪 One of the most common real-world examples of regression is

prediction of house prices . You can build a simple regression model to predict house prices based on data pertaining to land plot areas in square feet. Figure below shows two possible regression models based on different methods to predict house prices based on plot area.



🡪 The basic idea here is that we try to determine if there is any

relationship or association between the data feature plot area and the outcome variable, which is the house price and is what we want to predict.

🡪 Thus once we learn this trend or relationship depicted in Figure above, we can predict house prices in the future for any given plot of land. If you have noticed the figure closely, we depicted two types of models on purpose to show that there can be multiple ways to build a model on your training data. The main objective is to minimize errors during training and validating the model so that it generalized well, does not overfit or get biased only to the training data and performs well in future predictions.

🡪 **Simple linear regression** models try to model **relationships on data with one feature or explanatory variable x and a single response variable y where the objective is to predict y**. Methods like ordinary least squares (OLS) are typically used to get the best linear fit during model training.

🡪 **Multiple regression** is also known as **multivariable regression**. These methods try to model data where we **have one response output variable y in each observation but multiple explanatory variables in the form of a vector X instead of a single explanatory variable**. The idea is to predict y based on the different features present in X. A real-world example would be extending our house prediction model to build a more sophisticated model where we predict the house price based on multiple features instead of just plot area in each data sample. The features could be represented in a vector as plot area, number of bedrooms, number of bathrooms, total floors, furnished, or unfurnished. Based on all these attributes, the model tries to learn the relationship between each feature vector and its corresponding house price so that it can predict them in the future.

🡪 **Polynomial regression** is a special case of multiple regression where the response variable **y is modeled as an nth degree polynomial of the input feature x**. Basically it is multiple regression, where each feature in the input feature vector is a multiple of x. The model on the right in Figure above to predict house prices is a polynomial model of degree 2.

🡪 **Non-linear regression** methods try to model **relationships between input features and outputs based on a combination of non-linear functions** applied on the input features and necessary model parameters.

🡪 **Lasso regression** is a special form of regression that **performs normal regression and generalizes the model well by performing regularization as well as feature or variable selection**. **Lasso stands for least absolute shrinkage and selection operator**. The **L1 norm** is typically used as the regularization term in lasso regression.

🡪 **Ridge regression** is another special form of regression that **performs normal regression and generalizes the model by performing regularization to prevent overfitting the model.** Typically the **L2 norm** is used as the regularization term in ridge regression.

🡪 Generalized linear models are generic frameworks that can be used to model data predicting different types of output responses, including continuous, discrete, and ordinal data. Algorithms like logistic regression are used for categorical data and ordered probit regression for ordinal data.

1. Unsupervised Learning

🡪 Supervised learning methods usually require some training data where the outcomes which we are trying to predict are already available in the form of discrete labels or continuous values. However, often we do not have the liberty or advantage of having pre-labeled training data and we still want to extract useful insights or patterns from our data. In this scenario, unsupervised learning methods are extremely powerful. These methods are

called unsupervised because the model or algorithm tries to learn inherent latent structures, patterns and relationships from given data without any help or supervision like providing annotations in the form of labeled outputs or outcomes.

🡪 Unsupervised learning is more concerned with trying to extract

meaningful insights or information from data rather than trying to predict some outcome based on previously available supervised training data. There is more uncertainty in the results of unsupervised learning but you can also gain a lot of information from these models that was previously unavailable to view just by looking at the raw data. Often unsupervised learning could be one of the tasks involved in building a huge intelligence system. For example, we could use unsupervised learning to get possible

outcome labels for tweet sentiments by using the knowledge of the English vocabulary and then train a supervised model on similar data points and their outcomes which we obtained previously through unsupervised learning. There is no hard and fast rule with regard to using just one specific technique. You can always combine multiple methods as long as they are relevant in solving the problem. Unsupervised learning methods can be

categorized under the following broad areas of ML tasks relevant to unsupervised learning.

* + Clustering
  + Dimensionality Reduction
  + Anomaly Detection
  + Association Rule-Mining

🡪We explore this tasks briefly in the following sections to get a good feel of how unsupervised learning methods are used in the real world.

1. Clustering

🡪Clustering methods are Machine Learning methods that try to find patterns of similarity and relationships among data samples in our dataset and then cluster these samples into various groups, such that each group or cluster of

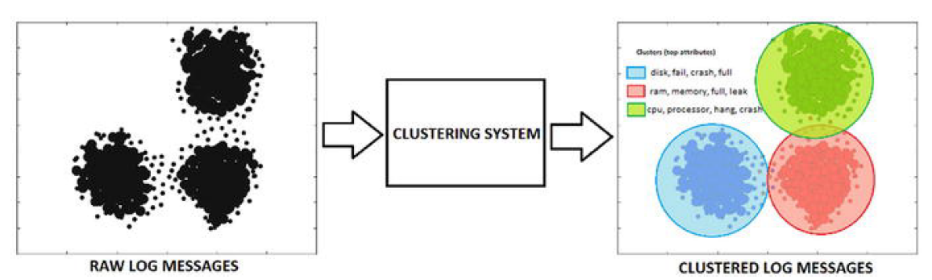
data samples has some similarity, based on the inherent attributes or features. These methods are completely unsupervised because they try to cluster data by looking at the data features without any prior training, supervision, or knowledge about data attributes, associations, and

relationships.

🡪Consider a real-world problem of running multiple servers in a data centre and trying to analyse logs for typical issues or errors. Our main task is to determine the various kinds of log messages that usually occur frequently each week. In simple words, we want to group log messages into

various clusters based on some inherent characteristics. A simple approach would be to extract features from the log messages, which would be in textual format and apply clustering on the same and group similar log messages together based on similarity in content.

🡪Figure below shows how clustering would solve this problem. Basically we have raw log messages to start with. Our clustering system would employ feature extraction to extract features from text like word occurrences, phrase occurrences, and so on. Finally, a clustering algorithm like K-means or hierarchical clustering would be employed to group or cluster messages based on similarity of their inherent features.



🡪It is quite clear from Figure above that our systems have three distinct clusters of log messages where the firs cluster depicts disk issues, the second cluster is about memory issues, and the third cluster is about processor issues. Top feature words that helped in distinguishing the clusters and grouping similar data samples (logs) together are also depicted in the figure. Of course, sometimes some features might be present across multiple data samples hence there can be slight overlap of clusters too since this is unsupervised learning. However, the main objective is always to create clusters such that elements of each cluster are near each other and far apart from elements of other clusters.

🡪There are various types of clustering methods that can be classified under the following major approaches.

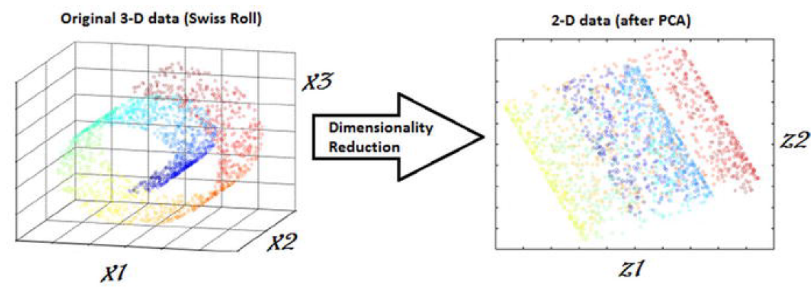
* Centroid based methods such as k-means and k-medoids
* Hierarchical clustering methods such as agglomerative and divisive(Ward’s, affinity propagation)
* Distribution based clustering methods such as Gaussian mixture models
* Density based methods such as dbscan and optics.

🡪 Besides this, we have several methods that recently came into the clustering landscape, like birch and clarans.

1. Dimensionality Reduction

🡪 Once we start extracting attributes or features from raw data samples, sometimes our feature space gets bloated up with a humongous number of features. This poses multiple challenges including analyzing and visualizing data with thousands or millions of features, which makes the feature space extremely complex posing problems with regard to training models, memory, and space constraints. In fact this is referred to as the “curse of dimensionality ”. 🡪Unsupervised methods can also be used in these

scenarios, where we reduce the number of features or attributes for each data sample. These methods reduce the number of feature variables by extracting or selecting a set of principal or representative features. There are multiple popular algorithms available for dimensionality reduction like Principal Component Analysis (PCA) , nearest neighbours, and discriminant analysis. Figure below shows the output of a typical feature reduction process applied to a Swiss Roll 3D structure having three dimensions to obtain a two-dimensional feature space for each data sample using PCA.



🡪 From Figure above, it is quite clear that each data sample originally had three features or dimensions, namely D(x1, x2, x3) and after applying PCA , we reduce each data sample from our dataset into two dimensions , namely D’(z1, z2). Dimensionality reduction techniques can be classified in two major approaches as follows.

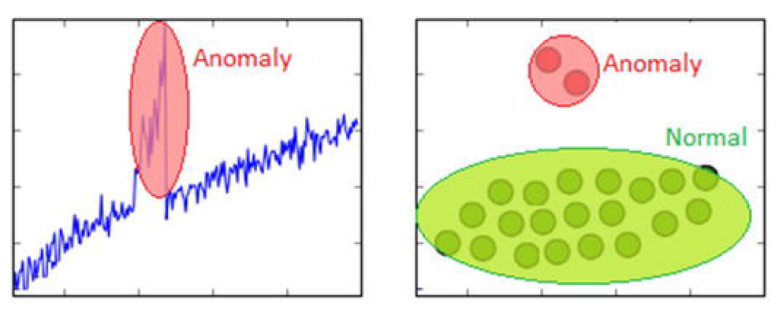
* + **Feature Selection Methods** : Specific features are selected for each data sample from the original list of features and other features are discarded. No new features are generated in this process.
  + **Feature Extraction Methods** : We engineer or extract new features from the original list of features in the data. Thus the reduced subset of features will contain newly generated features that were not part of the original feature set. PCA falls under this category.

1. Anomaly Detection

🡪 The process of anomaly detection is also termed as **outlier detection**, where we are interested in finding out occurrences of rare events or observations that typically do not occur normally based on historical data samples.

🡪 Sometimes anomalies occur infrequently and are thus rare events, and in other instances, anomalies might not be rare but might occur in very short bursts over time, thus have specific patterns. Unsupervised learning methods can be used for anomaly detection such that we train the algorithm on the training dataset having normal, non-anomalous data samples. Once it learns the necessary data representations, patterns, and relations among attributes in normal samples, for any new data sample, it would be able to identify it as anomalous or a normal data point by using its learned knowledge.

🡪 Figure below depicts some typical anomaly detection based scenarios where you could apply supervised methods like one-class SVM and unsupervised methods like clustering, K-nearest neighbours, autoencoders, and so on to detect anomalies based on data and its features.



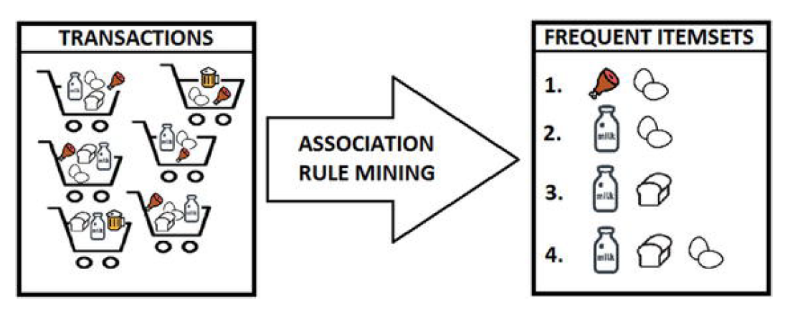
🡪 Anomaly detection based methods are extremely popular in real-world scenarios like detection of security attacks or breaches, credit card fraud, manufacturing anomalies, network issues, and many more.

1. Association Rule-Mining

🡪 Typically association rule-mining is a data mining method use to examine and analyse large transactional datasets to find patterns and rules of interest. These patterns represent interesting relationships and associations, among various items across transactions. Association rule-mining is also often termed as market basket analysis, which is used to analyse customer shopping patterns.

🡪 Association rules help in detecting and predicting transactional patterns based on the knowledge it gains from training transactions. Using this technique, we can answer questions like what items do people tend to buy together, thereby indicating frequent item sets. We can also associate or correlate products and items, i.e., insights like people

who buy beer also tend to buy chicken wings at a pub. Figure below shows how a typical association rule-mining method should work ideally on a transactional dataset.



🡪 From Figure above, you can clearly see that based on different customer transactions over a period of time, we have obtained the items that are closely associated and customers tend to buy them together. Some of these

frequent item sets are depicted like {meat, eggs}, {milk, eggs} and so on. The criterion of determining good quality association rules or frequent item sets is usually done using metrics like support, confidence, and lift.

🡪 This is an unsupervised method, because we have no idea what the frequent item sets are or which items are more strongly associated with which items beforehand. Only after applying algorithms like the apriori algorithm or FP-growth, can we detect and predict products or items

associated closely with each other and find conditional probabilistic dependencies.

1. Semi-Supervised Learning

🡪 **The semi-supervised learning methods typically fall between supervised and unsupervised learning methods**. These methods usually use a lot of training data that’s unlabeled (forming the unsupervised learning component) and a small amount of pre-labelled and annotated data (forming the supervised learning component). Multiple techniques are available in the form of generative methods, graph based methods, and heuristic based

methods.

🡪 A simple approach would be building a supervised model based on labeled data, which is limited, and then applying the same to large amounts of unlabeled data to get more labeled samples, train the model on them and repeat the process. Another approach would be to use unsupervised algorithms to cluster similar data samples, use human-in-the-loop efforts to manually annotate or label these groups, and then use a combination of this

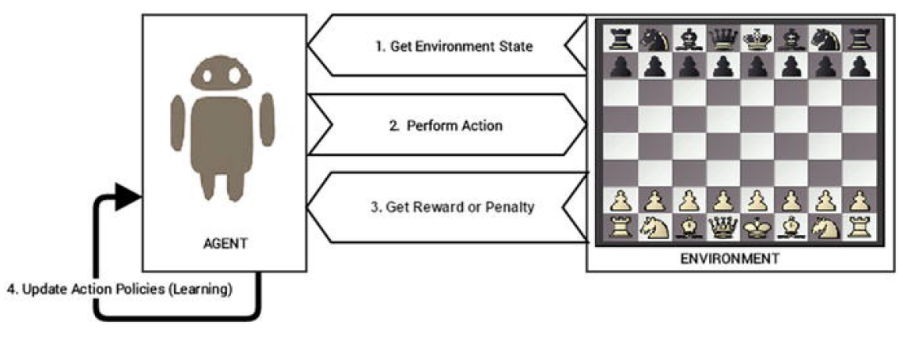
information in the future. This approach is used in many image tagging systems.

1. Reinforcement Learning

🡪 The reinforcement learning methods are a bit different from conventional supervised or unsupervised methods. **In this context, we have an agent that we want to train over a period of time to interact with a specific environment and improve its performance over a period of time with regard to the type of actions it performs on the environment**. **Typically the agent starts with a set of strategies or policies for interacting with the environment. On observing the environment, it takes a particular action based on a rule or policy and by observing the current state of the environment**. **Based on the action, the agent gets a reward, which could be beneficial or detrimental in the form of a penalty. It updates its current policies and strategies if needed and this iterative process continues till it learns enough about its environment to get the desired rewards**. The main steps of a reinforcement learning method are mentioned as follows.

1. Prepare agent with set of initial policies and strategy
2. Observe environment and current state
3. Select optimal policy and perform task
4. Get corresponding reward(or penalty)
5. Update Policies if needed
6. Repeat steps 2-5 iteratively until agent learns most optimal policies

🡪 Consider a real-world problem of trying to make a robot or a machine learn to play chess. In this case the agent would be the robot and the environment and states would be the chessboard and the positions of the chess pieces. A suitable reinforcement learning methodology is depicted in Figure below.



🡪 The main steps involved for making the robot learn to play chess is pictorially depicted in Figure above. This is based on the steps discussed earlier for any reinforcement learning method. In fact, Google’s DeepMind built the AlphaGo AI with components of reinforcement learning to train the system to play the game of Go.

1. Batch Learning

🡪 **Batch learning methods are also popularly known as offline learning methods** . These are Machine Learning methods that are used in end-to-end Machine Learning systems where the model is trained using all the available training data in one go. Once training is done and the model completes the process of learning, on getting a satisfactory performance, it is deployed into production where it predicts outputs for new data samples.

🡪However, the model doesn’t keep learning over a period of time

continuously with the new data. **Once the training is complete the model stops learning. Thus, since the model trains with data in one single batch and it is usually a one-time procedure, this is known as batch or offline learning**.

🡪 We can always train the model on new data but then we would have to add new data samples along with the older historical training data and again re-build the model using this new batch of data. If most of the model building workflow has already been implemented, retraining a model would not involve a lot of effort; however, with the data size getting bigger with each new data sample, the retraining process will start consuming more processor, memory, and disk resources over a period of time. These are some points to be considered when you are building models that would be running from systems having limited capacity.

1. Online Learning

🡪 Online learning methods work in a different way as compared to batch learning methods. **The training data is usually fed in multiple incremental batches to the algorithm**. These data batches are also known as mini-batches in ML terminology. **However, the training process does not end there unlike batch learning methods. It keeps on learning over a period of time based on new data samples which are sent to it for prediction. Basically it predicts and learns in the process with new data on the fly without have to re-run the whole model** on previous data samples.

🡪 There are several advantages to online learning—it is suitable in real world scenarios where the model might need to keep learning and retraining on new data samples as they arrive. Problems like device failure or anomaly prediction and stock market forecasting are two relevant scenarios. Besides this, since the data is fed to the model in incremental mini-batches, you can build these models on commodity hardware without worrying

about memory or disk constraints since unlike batch learning methods, you do not need to load the full dataset in memory before training the model. Besides this, once the model trains on datasets, you can remove them since we do not need the same data again as the model learns incrementally and remembers what it has learned in the past.

🡪 One of the major caveats in online learning methods is the fact **that bad data samples can affect the model performance adversely**. **All ML methods work on the principle of “Garbage In Garbage Out”**. **Hence if you supply bad data samples to a well-trained model, it can start learning relationships and patterns that have no real significance and this ends up affecting the overall model performance**. Since online learning methods keep learning based on new data samples, **you should ensure proper checks are in place to notify you in case suddenly the model performance drops**. Also suitable model parameters like learning rate should be selected with care to ensure the model doesn’t overfit or get biased based on specific data samples.

1. Instance Based Learning

🡪 There are various ways to build Machine Learning models using methods that try to generalize based on input data. Instance based learning involves ML systems and methods that use the raw data points themselves to figure out outcomes for newer, previously unseen data samples instead of building an explicit model on training data and then testing it out.

🡪A simple example would be a K-nearest neighbor algorithm. Assuming k = 3, we have our initial training data. The ML method knows the representation of the data from the features, including its dimensions, position of each data point, and so on. For any new data point, it will use a similarity measure (like cosine or Euclidean distance) and find the three nearest input data points to this new data point. Once that is decided, we simply take a majority of the outcomes for those three training points and

predict or assign it as the outcome label/response for this new data point.

🡪Thus, instance based learning works by looking at the input data points and using a similarity metric to generalize and predict for new data points.

1. Model Based Learning

🡪 The model based learning methods are a more traditional ML approach toward generalizing based on training data. **Typically an iterative process takes place where the input data is used to extract features and models are built based on various model parameters (known as hyperparameters)**.

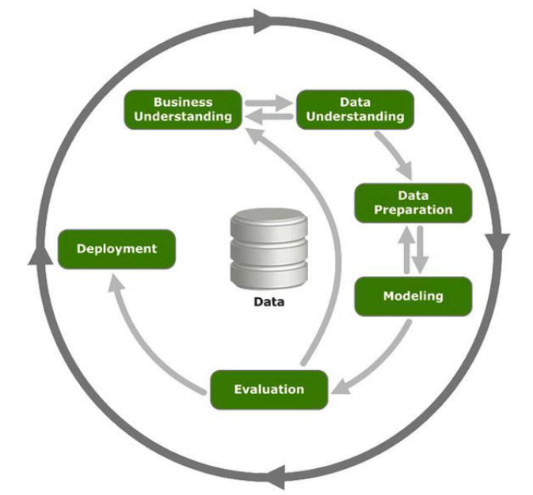
🡪 These hyperparameters are optimized based on various model validation techniques to select the model that generalizes best on the training data and some amount of validation and test data (split from the initial dataset). Finally, the best model is used to make predictions or decisions as and when needed.

1. The CRISP-DM Process Model

🡪 The CRISP-DM model stands for CRoss Industry Standard Process for Data Mining . More popularly known by the acronym itself, CRISP-DM is a tried, tested, and robust industry standard process model followed for data mining and analytics projects. CRISP-DM clearly depicts necessary steps, processes, and workflows for executing any project right from formalizing

business requirements to testing and deploying a solution to transform data into insights. Data Science, Data Mining, and Machine Learning are all about trying to run multiple iterative processes to extract insights and information from data. Hence we can say that analyzing data is truly both an art as well as a science, because it is not always about running algorithms without reason; a lot of the major effort involves in understanding the business, the actual value of the efforts being invested, and proper methods to articulate end results and insights.

🡪 The CRISP-DM model tells us that for building an end-to-end solution for any analytics project or system, there are a total of six major steps or phases, some of them being iterative. Just like we have a software development lifecycle with several major phases or steps for a software development project, we have a data mining or analysis lifecycle in this scenario. Figure below depicts the data mining lifecycle with the CRISP-DM model.



🡪 Figure above clearly shows there are a total of six major phases in the data mining lifecycle and the direction to proceed is depicted with arrows. This model is not a rigid imposition but rather a framework to ensure you are on the right track when going through the lifecycle of any analytics project. In some scenarios like anomaly detection or trend analysis, you might be more interested in data understanding, exploration, and visualization rather than intensive modeling. Each of the six phases is described in detail as follows.

1. Business Understanding

🡪 This is the initial phase before kick starting any project in full flow. However this is one of the most important phases in the lifecycle! The main objective here starts with understanding the business context and requirements for the problem to be solved at hand. Definition of business requirements is crucial to convert the business problem into a data mining or analytics problem and to set expectations and success criteria for both the customer as well as the solution task force. The final deliverable from this phase would be a detailed plan with the major milestones of the project and expected timelines along with success criteria, assumptions, constraints, caveats, and challenges.

🡺Define Business Problem

🡪 The first task in this phase would be to start by understanding the business objective of the problem to be solved and build a formal definition of the problem. The following points are crucial toward clearly articulating and defining the business problem.

Get business context of the problem to be solved, assess the problem with the help of domain, and **subject matter experts (SMEs).**

* Describe main pain points or target areas for business objective to be solved.
* Understand the solutions that are currently in place, what is lacking, and what needs to be improved.
* Define the business objective along with proper deliverables and success criteria based on inputs from business, data scientists, analysts, and SMEs.

🡺Asses and Analyze Scenarios

🡪 Once the business problem is defined clearly, the main tasks involved would be to analyze and assess the current scenario with regard to the business problem definition. This includes looking at what is currently available and making a note of various items required ranging from resources, personnel, to data. Besides this, proper assessment of risks and contingency plans need to be discussed. The main steps involved in the assessment stage here are mentioned as follows.

* Assess and analyze what is currently available to solve the problem from various perspectives including data personnel, resource time, and risks.
* Build out a brief report of key resources needed (both hardware and software) and personnel involved. In case of any shortcomings, make sure to call them out as necessary.
* Discuss business objective requirements one by one and then identify and record possible assumptions and constraints for each requirement with the help of SMEs.
* Verify assumptions and constraints based on data available (a lot of this might be answered only after detailed analysis, hence it depends on the problem to be solved and the data available).
* Document and report possible risks involved in the project including timelines, resources, personnel, data , and financial based concerns. Build contingency plans for each possible scenario.
* Discuss success criteria and try to document a comparative return on investment or cost versus valuation analysis if needed. This just needs to be a rough benchmark to make sure the project aligns with the company or business vision.

🡺Define Data Mining Problem

🡪 This could be defined as the pre-analysis phase, which starts once the success criteria and the business problem is defined and all the risks, assumptions and constraints have been documented. This phase involves having detailed technical discussions with your analysts, data scientists, and developers and keeping the business stakeholders in sync. The following are the key tasks that are to be undertaken in this phase.

* Discuss and document possible Machine Learning and data mining methods suitable for the solution by assessing possible tools, algorithms, and techniques.
* Develop high-level designs for end-to-end solution architecture.
* Record notes on what the end output from the solution will be and how will it integrate with existing business components.
* Record success evaluation criteria from a Data Science standpoint. A simple example could be making sure that predictions are at least 80% accurate.

🡺Project Plan

🡪 This is the final stage under the business understanding phase. A project plan is generally created consisting of the entire major six phases in the CRISP-DM model , estimated timelines, allocated resources and personnel, and possible risks and contingency plans. Care is taken to ensure concrete high-level deliverables and success criteria are defined for each phase and iterative phases like modeling are highlighted with annotations like feedback based on SMEs might need models to be rebuilt and retuned before deployment. You should be ready for the next step once you have the following points covered.

* + - * + Definition of business objectives for the problem
        + Success criteria for business and data mining efforts
        + Budget allocation and resource planning
        + Clear, well-defined Machine Learning and data mining methodologies to be followed, including high-level workflows from exploration to deployment
        + Detailed project plan with all six phases of the CRISP-DM model defined with estimated timelines and risks

1. Data understanding

🡪 The second phase in the CRISP-DM process involves taking a deep dive into the data available and understanding it in further detail before starting the process of analysis. This involves collecting the data, describing the various attributes, performing some exploratory analysis of the data, and keeping tabs on data quality. This phase should not be neglected because bad data or insufficient knowledge about available data can have cascading adverse effects in the later stages in this process.

🡺Data Collection

🡪 This task is undertaken to extract, curate, and collect all the necessary data needed for your business objective. Usually this involves making use of the organizations historical data warehouses, data marts, data lakes and so on. An assessment is done based on the existing data available in the organization and if there is any need for additional data. This can be obtained from the web, i.e., open data sources or it can be obtained from other channels like surveys, purchases, experiments and simulations. Detailed documents should keep track of all datasets which would be used for analysis and additional data sources if any are necessary. This document can be combined with the subsequent stages of this phase.

🡺Data Description

🡪 Data description involves carrying out initial analysis on the data to understand more about the data, its source, volume, attributes, and relationships. Once these details are documented, any shortcomings if noted should be informed to relevant personnel. The following factors are crucial to building a proper data description document.

* + - * + Data sources (SQL, NoSQL, Big Data), record of origin (ROO), record of reference(ROR)
        + Data volume (size, number of records, total databases, tables)
        + Data attributes and their description (variables, data types)
        + Relationship and mapping schemes (understand attribute representations)
        + Basic descriptive statistics (mean, median, variance)
        + Focus on which attributes are important for the business

🡺Exploratory Data Analysis

🡪 Exploratory data analysis , also known as EDA, is one of the first major analysis stages in the lifecycle. Here, the main objective is to explore and understand the data in detail. You can make use of descriptive statistics, plots, charts, and visualizations to look at the various data attributes, find associations and correlations and make a note of data quality problems if any. Following are some of the major tasks in this stage.

* + - * + Explore, describe, and visualize data attributes
        + Select data and attributes subsets that seem most important for the problem
        + Extensive analysis to find correlations and associations and test hypotheses
        + Note missing data points if any

🡺Data Quality Analysis

🡪 Data quality analysis is the final stage in the data understanding phase where we analyze the quality of data in our datasets and document potential errors, shortcomings, and issues that need to be resolved before analyzing the data further or starting modeling efforts. The main focus on data quality analysis involves the following.

* Missing Values
* Inconsistent values
* Wrong information due to data errors (manual/automated)
* Wrong metadata information

1. Data Preparation

🡪 The third phase in the CRISP-DM process takes place after gaining enough knowledge on the business problem and relevant dataset. Data preparation is mainly a set of tasks that are performed to clean, wrangle, curate, and

prepare the data before running any analytical or Machine Learning methods and building models. We will briefly discuss some of the major tasks under the data preparation phase in this section. An important point to remember here is that data preparation usually is the most time consuming

phase in the data mining lifecycle and often takes 60% to 70% time in the overall project. However this phase should be taken very seriously because, like we have discussed multiple times before, bad data will lead to bad models and poor performance and results.

🡺Data Integration

🡪 The process of data integration is mainly done when we have multiple datasets that we might want to integrate or merge. This can be done in two ways. Appending several datasets by combining them, which is typically done for datasets having the same attributes. Merging several datasets together having different attributes or columns, by using common fields like keys.

🡺Data Wrangling

🡪 The process of data wrangling or data munging involves data processing, cleaning, normalization, and formatting. Data in its raw form is rarely consumable by Machine Learning methods to build models. Hence we need to process the data based on its form, clean underlying errors and inconsistencies, and format it into more consumable formats for ML algorithms. Following are the main tasks relevant to data wrangling.

Handling missing values (remove rows, impute missing values)

Handling data inconsistencies (delete rows, attributes, fix inconsistencies)

Fixing incorrect metadata and annotations

Handling ambiguous attribute values

Curating and formatting data into necessary formats (CSV, Json, relational)

🡺Attribute Generation And Selection

🡪 Data is comprised of observations or samples (rows) and attributes or features (columns). The process of attribute generation is also known as feature extraction and engineering in Machine Learning terminology. Attribute generation is basically creating new attributes or variables from existing attributes based on some rules, logic, or hypothesis. A simple example would be creating a new numeric variable called age based on two date-time fields—current\_date and birth\_date—for a dataset of employees in an organization. There are several techniques with regard to attribute generation that we discuss in future chapters.

🡪Attribute selection is basically selecting a subset of features or attributes from the dataset based on parameters like attribute importance, quality, relevancy, assumptions, and constraints. Sometimes even Machine Learning methods are used to select relevant attributes based on the data. This is popularly known as feature selection in Machine Learning terminology.

1. Modelling

🡪The fourth phase in the CRISP-DM process is the core phase in the process where most of the analysis takes place with regard to using clean, formatted data and its attributes to build models to solve business problems. This is an

iterative process, along with model evaluation and all the preceding steps leading up to modelling. The basic idea is to build multiple models iteratively trying to get to the best model that satisfies our success criteria, data mining objectives, and business objectives. We briefly talk about some of the major stages relevant to modelling in this section.

🡺Selecting Modelling Techniques

🡪In this stage, we pick up a list of relevant Machine Learning and data mining tools, frameworks, techniques, and algorithms listed in the “Business Understanding phase. Techniques that are proven to be robust and useful in solving the problem are usually selected based on inputs and insights from data analysts and data scientists. These are mainly decided by the current data available, business goals, data mining goals, algorithm requirements, and constraints.

🡺Model Building

🡪**The process of model building is also known as training the model using data and features from our dataset**. **A combination of data (features) and Machine Learning algorithms together give us a model that tries to generalize on the training data and give necessary results in the form of insights and/or predictions**. Generally various algorithms are used to try out multiple modeling approaches on the same data to solve the same problem to get the best model that performs and gives outputs that are the closest to the business success criteria. Key things to keep track here are the models created, model parameters being used, and their results.

🡺Model Evaluation and Tuning

🡪In this stage, we **evaluate each model based on several metrics like model accuracy, precision, recall, F1 score, and so on**. We also tune the model parameters based on techniques like grid search and cross validation to get to the model that gives us the best results. Tuned models are also matched with the data mining goals to see if we are able to get the desired results as well as performance. Model tuning is also termed as hyperparameter optimization in the Machine Learning world.

🡺Model Assessment

🡪Once we have models that are providing desirable and relevant results, a detailed assessment of the model is performed based on the following parameters.

* Model Performance is in line with defined success criteria
* Reproducible and consistent results from models
* Scalability, robustness, and ease of deployment
* Future extensibility of the model
* Model evaluation gives satisfactory results

1. Evaluation

🡪The fifth phase in the CRISP-DM process takes place once we have the final models from the modeling phase that satisfy necessary success criteria with respect to our data mining goals and have the desired performance and

results with regard to model evaluation metrics like accuracy. The evaluation phase involves carrying out a detailed assessment and review of the final models and the results which are obtained from them. Some of the main points that are evaluated in this section are as follows.

* Ranking final models based on the quality of results and their relevancy based on alignment with business objectives
* Any assumptions or constraints that were invalidated by the models
* Cost of deployment of the entire Machine Learning pipeline from data extraction and processing to modelling and predictions
* Any pain points in the whole process? What should be recommended? What should be avoided?
* Data sufficiency report based on results
* Final suggestions, feedback, and recommendations from solutions team and SMEs

🡪 Based on the report formed from these points, after a discussion, the team can decide whether they want to proceed to the next phase of model deployment or a full reiteration is needed, starting from business and data

understanding to modelling.

1. Deployment

🡪 The final phase in the CRISP-DM process is all about deploying your selected models to production and making sure the transition from development to production is seamless. Usually most organizations follow a standard path-to-production methodology. A proper plan for deployment is built based on resources required, servers, hardware, software, and so on. Models are validated, saved, and deployed on necessary systems and servers. A plan is also put in place for regular monitoring an maintenance of models to continuously evaluate their performance, check for results and their validity, and retire, replace, and update models as and when needed.

1. Building Machine Intelligence

🡪The objective of Machine Learning, data mining, or artificial intelligence is to make our lives easier, automate tasks, and take better decisions. Building machine intelligence involves everything we have learned until now starting from Machine Learning concepts to actually implementing and building models and using them in the real world. Machine intelligence can be built

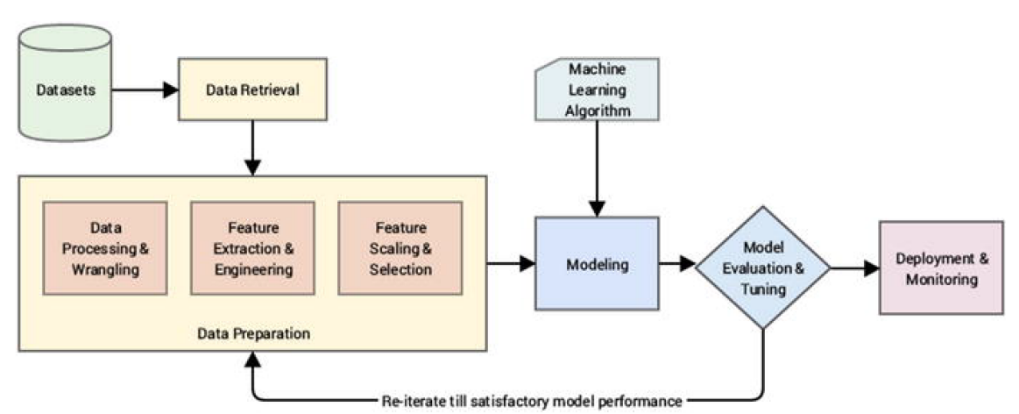
using non-traditional computing approaches like Machine Learning. In this section, we establish full-fledged end-to-end Machine Learning pipelines based on the CRISP-DM model, which will help us solve real-world problems by building machine intelligence using a structured process.

1. Machine Learning Pipelines

🡪The best way to solve a real-world Machine Learning or analytics problem is to use a Machine Learning pipeline starting from getting your data to transforming it into information and insights using Machine Learning algorithms and techniques. This is more of a technical or solution based pipeline and it assumes that several aspects of the CRISP-DM model are already covered, including the following points.

* Business and data understanding
* ML/DM technique selection
* Risk, assumptions, and constraints assessment

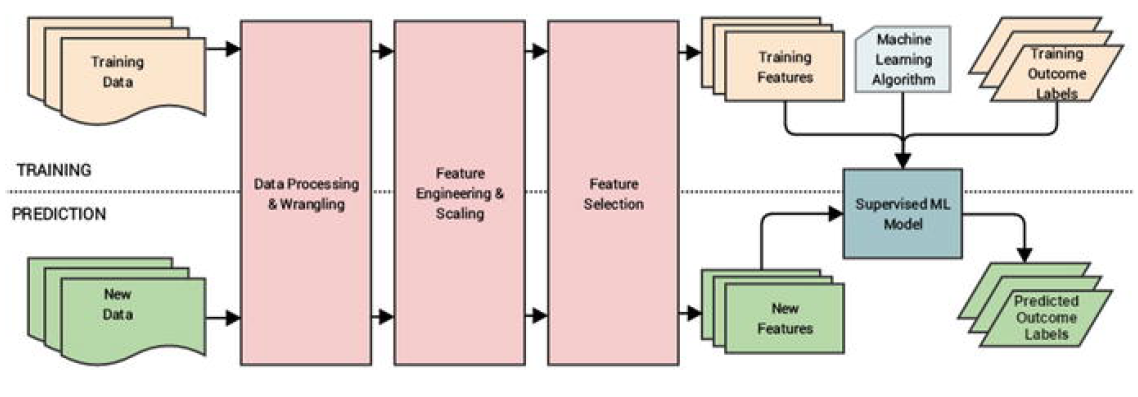
🡪 A Machine Learning pipeline will mainly consist of elements related to data retrieval and extraction, preparation, modelling, evaluation, and deployment. Figure below shows a high-level overview of a standard Machine Learning pipeline with the major phases highlighted in their blocks.



🡪 From Figure above, it is evident that there are several major phases in the Machine Learning pipeline and they are quite similar to the CRISP-DM process model, which is why we talked about it in detail earlier. The major steps in the pipeline are briefly mentioned here.

* **Data retrieval**: This is mainly data collection, extraction, and acquisition from various data sources and data stores.
* **Data preparation**: In this step, we pre-process the data, clean it, wrangle it, and manipulate it as needed. Initial exploratory data analysis is also carried out. Next steps involved extracting, engineering, and selecting features/attributes from the data.
  + **Data processing and wrangling**: Mainly concerned with data processing, cleaning, munging , wrangling and performing initial descriptive and exploratory data analysis.
  + **Feature extraction and engineering**: Here, we extract important features or attributes from the raw data and even create or engineer new features from existing features.
  + **Feature scaling and selection**: Data features often need to be normalized and scaled to prevent Machine Learning algorithms from getting biased. Besides this, often we need to select a subset of all available features based on feature importance and quality. This process is known as feature selection.
* **Modelling**: In the process of modelling, we usually feed the data features to a Machine Learning method or algorithm and train the model, typically to optimize a specific cost function in most cases with the objective of reducing errors and generalizing the representations learned from the data.
* **Model evaluation and tuning**: Built models are evaluated and tested on validation datasets and, based on metrics like accuracy, F1 score, and others, the model performance is evaluated. Models have various parameters that are tuned in a process called hyperparameter optimization to get models with the best and optimal results.
* **Deployment and monitoring**: Selected models are deployed in production and are constantly monitored based on their predictions and results.
  1. Supervised Machine Learning Pipeline

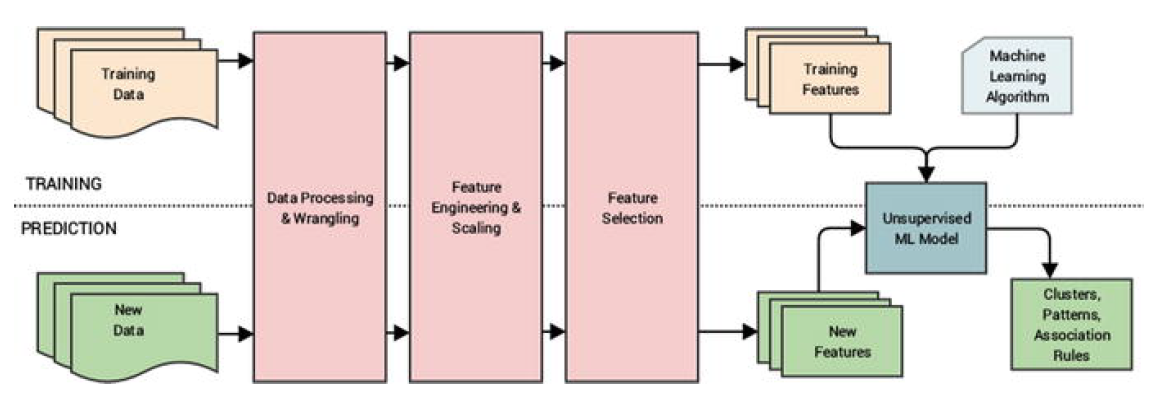
🡪 By now we know that supervised Machine Learning methods are all about working with supervised labeled data to train models and then predict outcomes for new data samples. Some processes like feature engineering, scaling, and selection should always remain constant so that the same features are used for training the model and the same features are extracted from new data samples to feed the model in the prediction phase. Based on our earlier generic Machine Learning pipeline, Figure below shows a standard supervised Machine Learning pipeline.



🡪 You can clearly see the two phases of model training and prediction highlighted in Figure above. Also, based on what we had mentioned earlier, the same sequence of data processing, wrangling, feature engineering, scaling, and selection is used for both data used in training the model and future data samples for which the model predicts outcomes. This is a very important point that you must remember whenever you are building any supervised model. Besides this, as depicted, the model is a combination of a Machine Learning (supervised) algorithm and training data features and corresponding labels. This model will take features from new data samples and output predicted labels in the prediction phase.

* 1. Un-supervised Machine Learning Pipeline

🡪 Unsupervised Machine Learning is all about extracting patterns, relationships, associations, and clusters from data. The processes related to feature engineering, scaling and selection are similar to supervised learning. However there is no concept of pre-labelled data here. Hence the unsupervised Machine Learning pipeline would be slightly different in contrast to the supervised pipeline. Figure below depicts a standard unsupervised Machine Learning pipeline.



🡪 Figure above clearly depicts that no supervised labelled data is used for training the model. With the absence of labels, we just have training data that goes through the same data preparation phase as in the supervised learning pipeline and we build our unsupervised model with an unsupervised Machine Learning algorithm and training features. In the prediction phase, we extract features from new data samples and pass them through the model which gives relevant results according to the type of Machine Learning task we are trying to perform, which can be clustering, pattern detection, association rules, or dimensionality reduction.

1. Real-World Case Study: Predicting Student Grant Recommendations

🡪 Let’s take a step back from what we have learned so far! The main objective here was to gain a solid grasp over the entire Machine Learning landscape, understand crucial concepts, build on the basic foundations, and understand how to execute Machine Learning projects with the help of Machine Learning pipelines with the CRISP-DM process model being the source of all inspiration. Let’s put all this together to take a very basic real world case study by building a supervised Machine Learning pipeline on a toy dataset. Our major objective is as follows. Given that you have several students with multiple attributes like grades, performance, and scores, can you build a model based on past historical data to predict the chance of the student getting a recommendation grant for a research project?

🡪 This will be a quick walkthrough with the main intent of depicting how to build and deploy a real-world Machine Learning pipeline and perform predictions. This will also give you a good hands-on experience to get started with Machine Learning. Do not worry too much if you don’t understand the details of each and every line of code; the subsequent chapters cover all the tools, techniques, and frameworks used here in detail. We will be using Python 3.5 in this book. You can follow along with the code snippets in this section or open the Predicting Student Recommendation Machine Learning Pipeline.ipynb jupyter notebook by running jupyter notebook in the command line/terminal in the same directory as this notebook. You can then run the relevant code snippets in the notebook from you browser.

🡺Objective

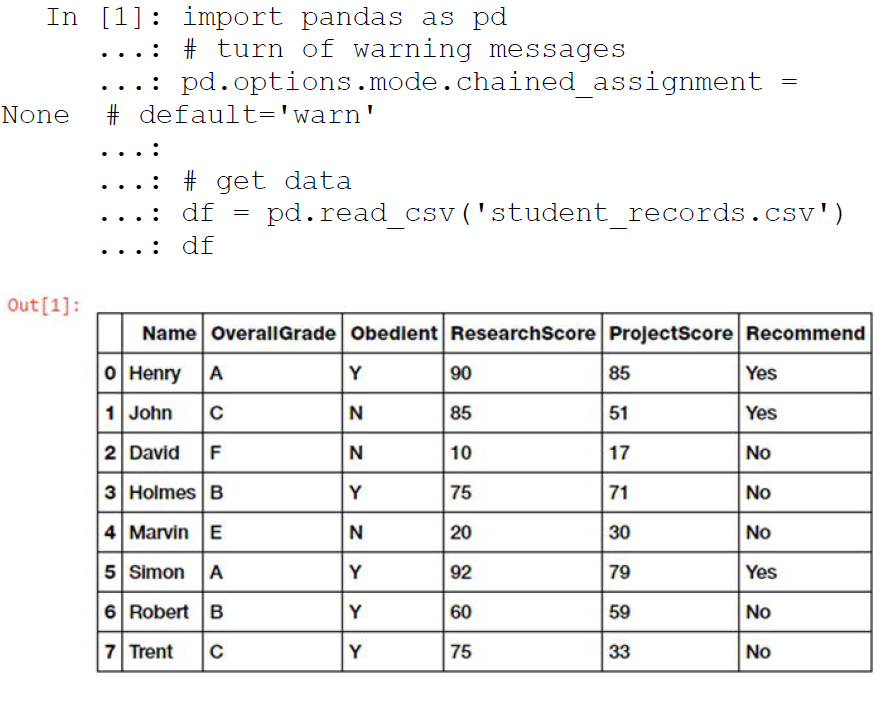
🡪 You have historical student performance data and their grant recommendation outcomes in the form of a comma separated value file named student\_records.csv. Each data sample consists of the following attributes.

* Name (the student name)
* OverallGrade (overall grade obtained)
* Obedient (whether they were diligent during their course of stay)
* ResearchScore (marks obtained in their research work)
* ProjectScore (marks obtained in the project)
* Recommend (whether they got the grant recommendation)

🡪 You main objective is to build a predictive model based on this data such that you can predict for any future student whether they will be recommended for the grant based on their performance attributes.

🡺Data Retrieval

🡪 Here, we will leverage the pandas framework to retrieve the data from the CSV file. The following snippet shows us how to retrieve the data and view it.



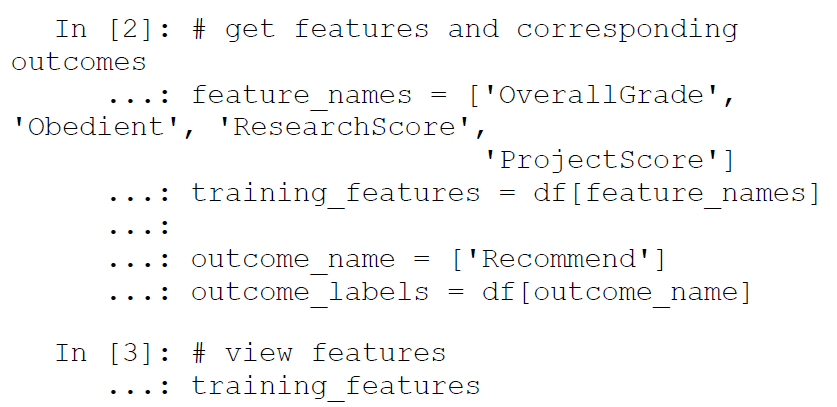
🡪 Now that we can see data samples showing records for each student and their corresponding recommendation outcomes in Figure above, we will perform necessary tasks relevant to data preparation .

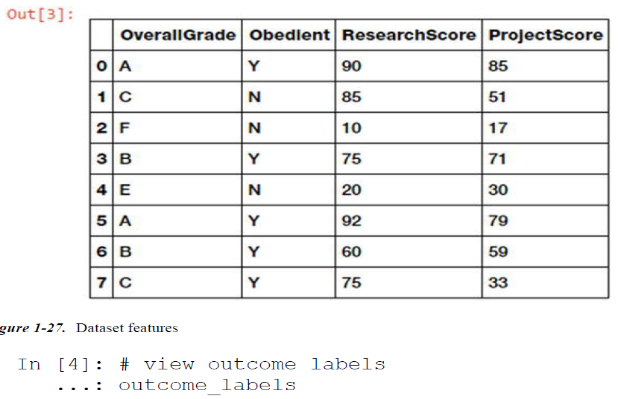
🡺Data Preparation

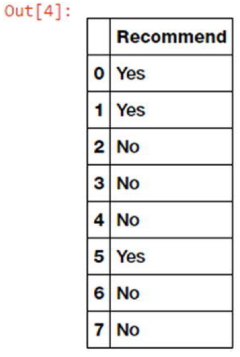
🡪 Based on the dataset we saw earlier, we do not have any data errors or missing values, hence we will mainly focus on feature engineering and scaling in this section.

🡺Feature Extraction And Engineering

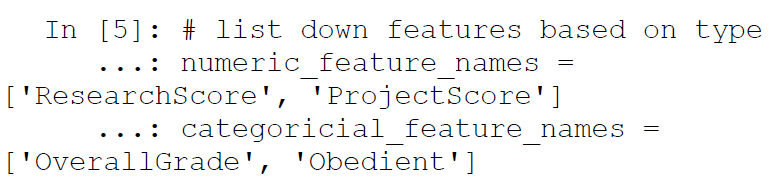
🡪 Let’s start by extracting the existing features from the dataset and the outcomes in separate variables. The following snippet shows this process. See Figures figures shown below :



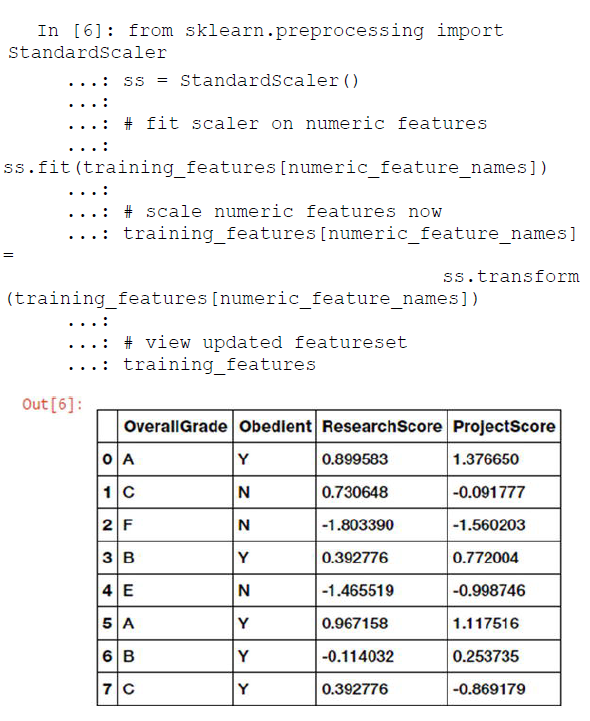




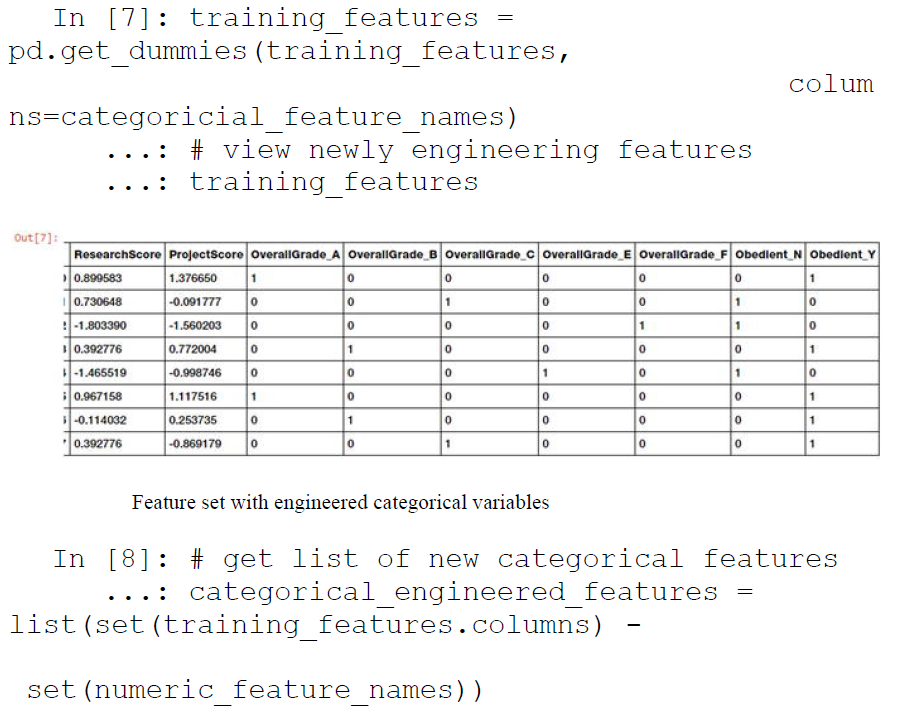
🡪 Now that we have extracted our initial available features from the data and their corresponding outcome labels, let’s separate out our available features based on their type (numerical and categorical).



🡪 We will now use a standard scalar from scikit-learn to scale or normalize our two numeric score-based attributes using the following code.



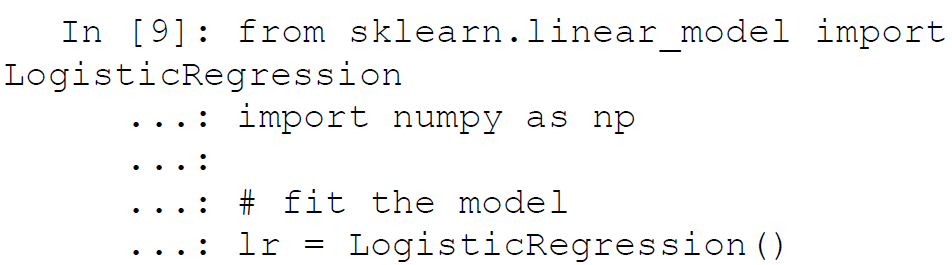
🡪 Now that we have successfully scaled our numeric features, let’s handle our categorical features and carry out the necessary feature engineering needed based on the following code.

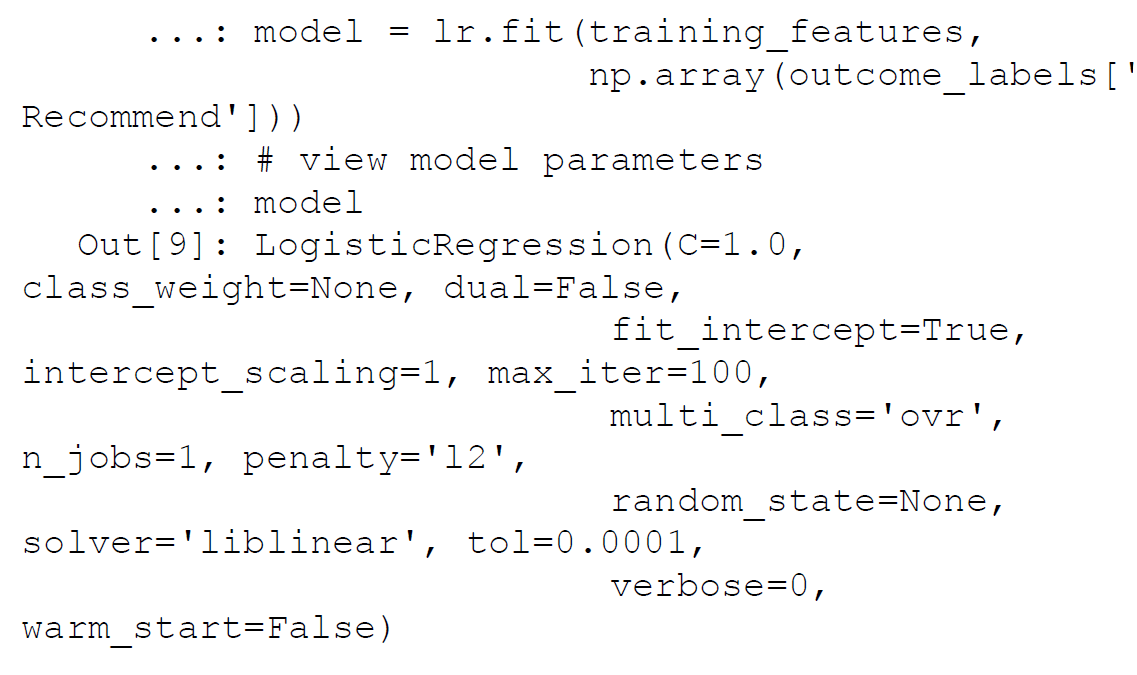


🡪 Figure above shows us the updated feature set with the newly engineered categorical variables. This process is also known as one hot encoding.

🡺Modelling

🡪 We will now build a simple classification (supervised) model based on our feature set by using the logistic regression algorithm . The following code depicts how to build the supervised model.

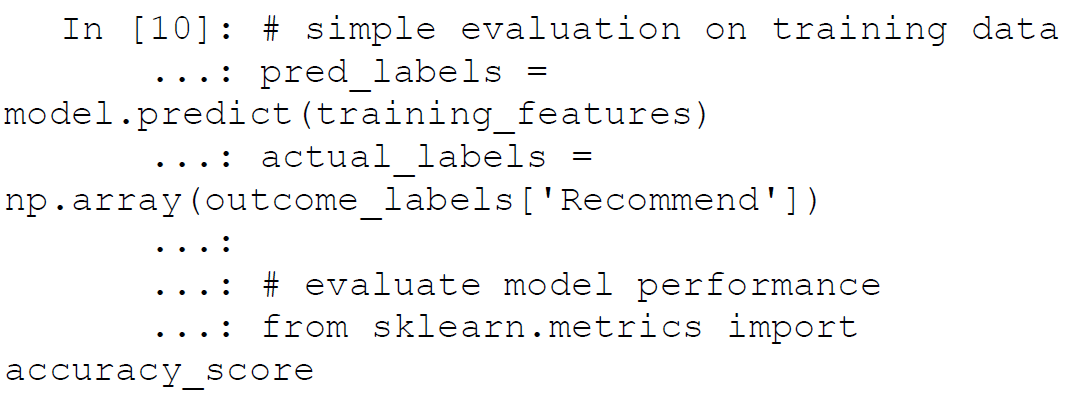


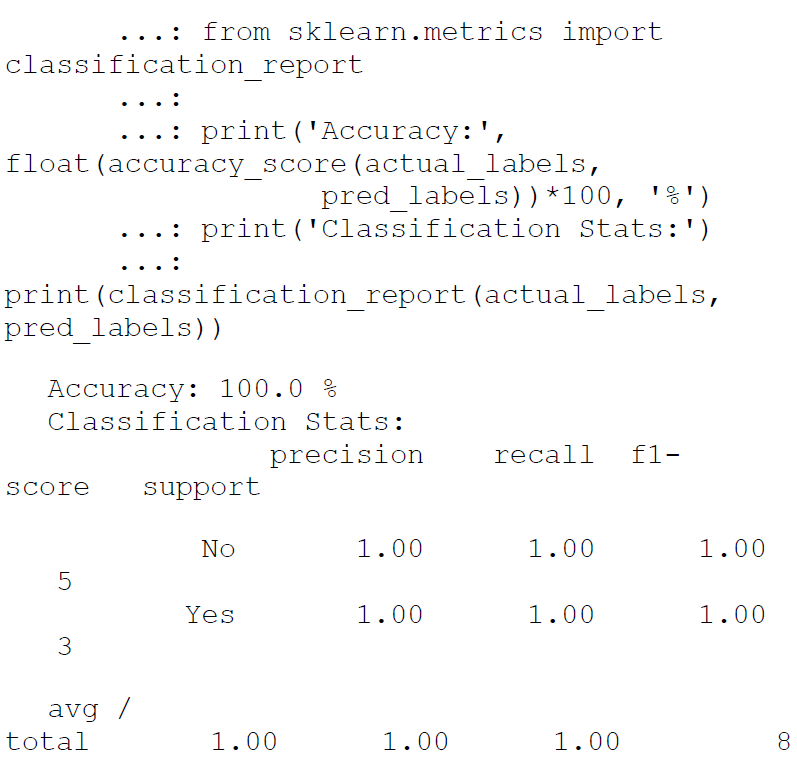


🡪 Thus, we now have our supervised learning model based on the logistic regression model with L2 regularization, as you can see from the parameters in the previous output.

🡺Model Evaluation

🡪 Typically model evaluation is done based on some holdout or validation dataset that is different from the training dataset to prevent overfitting or biasing the model. Since this is an example on a toy dataset, let’s evaluate the performance of our model on the training data using the following snippet.

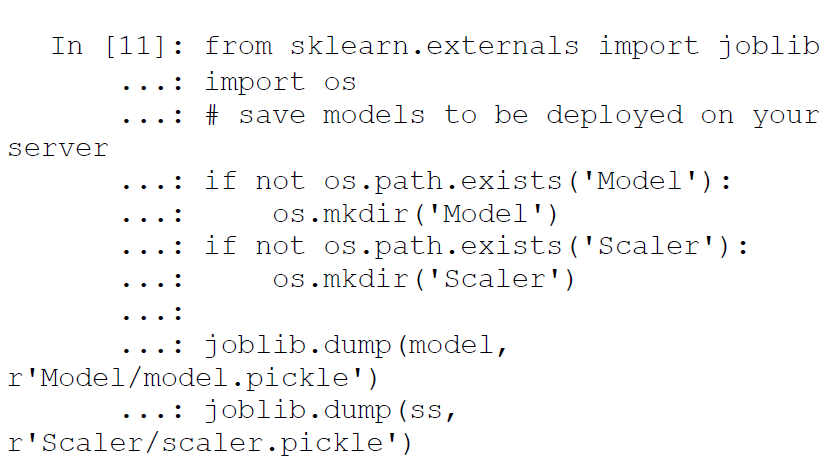




🡪 Thus you can see the various metrics that we had mentioned earlier, like accuracy, precision, recall, and F1 score depicting the model performance.

🡺Model Development

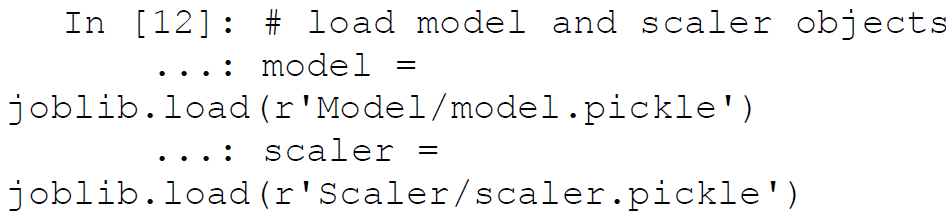
🡪 We built our first supervised learning model, and to deploy this model typically in a system or server, we need to persist the model. We also need to save the scalar object we used to scale the numerical features since we use it to transform the numeric features of new data samples. The following snippet depicts a way to store the model and scalar objects.



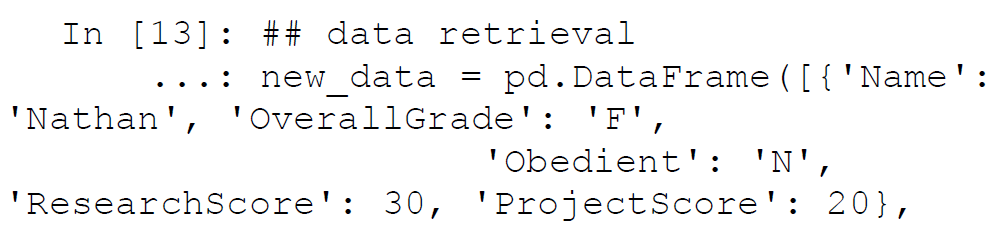
🡪 These files can be easily deployed on a server with necessary code to reload the model and predict new data samples, which we will see in the upcoming sections.

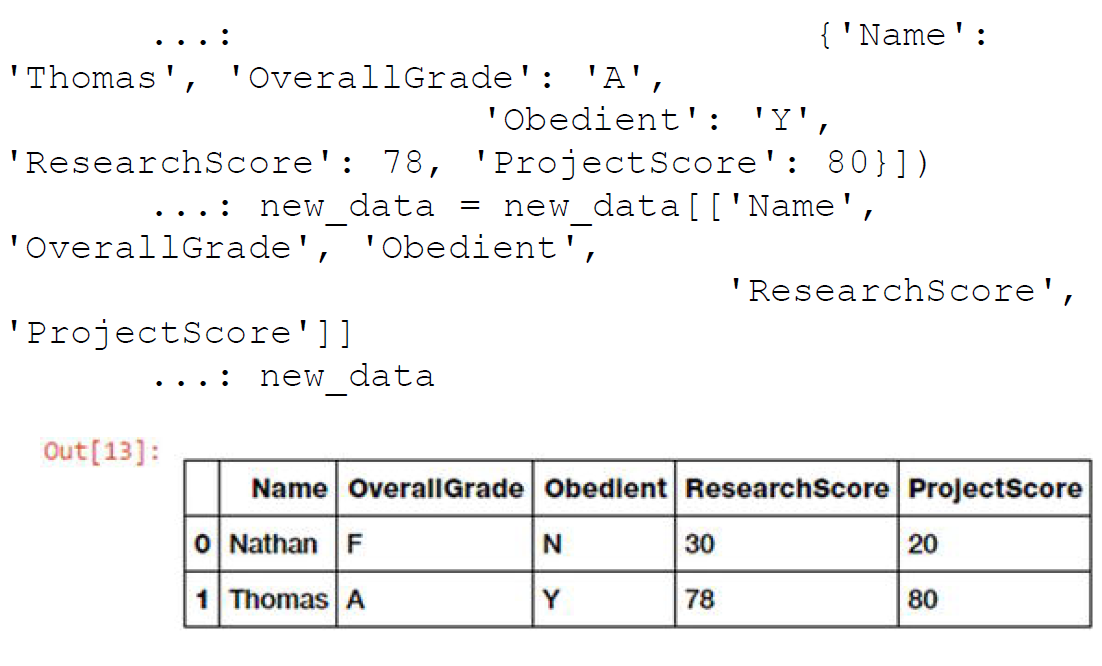
🡺Prediction in Action

🡪 We are now ready to start predicting with our newly built and deployed model! To start predictions, we need to load our model and scalar objects into memory. The following code helps us do this.

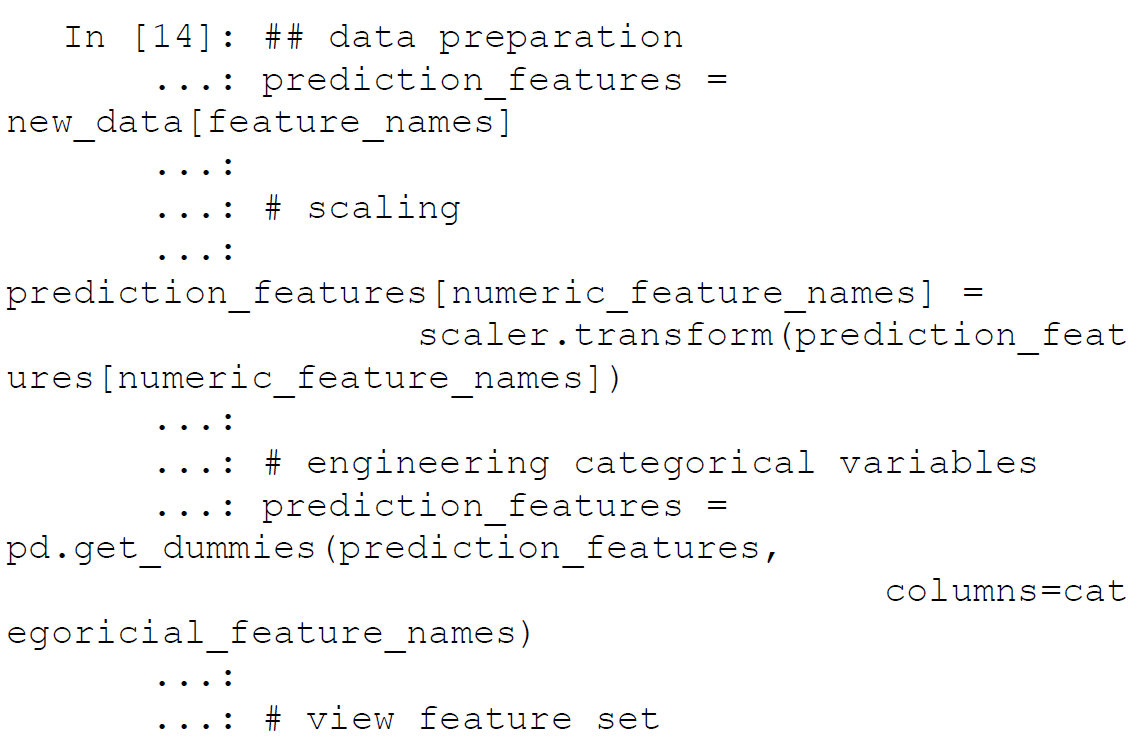


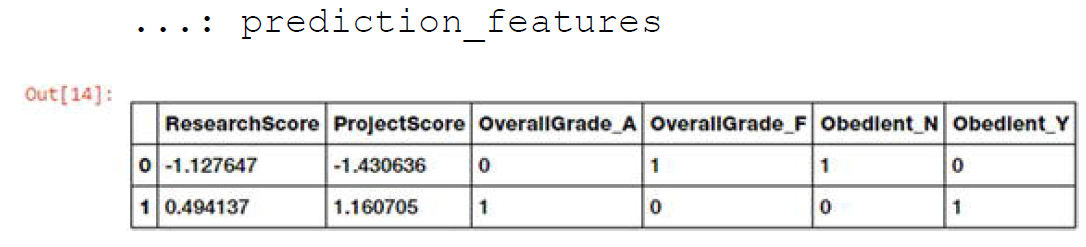
🡪 We have some sample new student records (for two students) for which we want our model to predict if they will get the grant recommendation. Let’s retrieve and view this data using the following code.



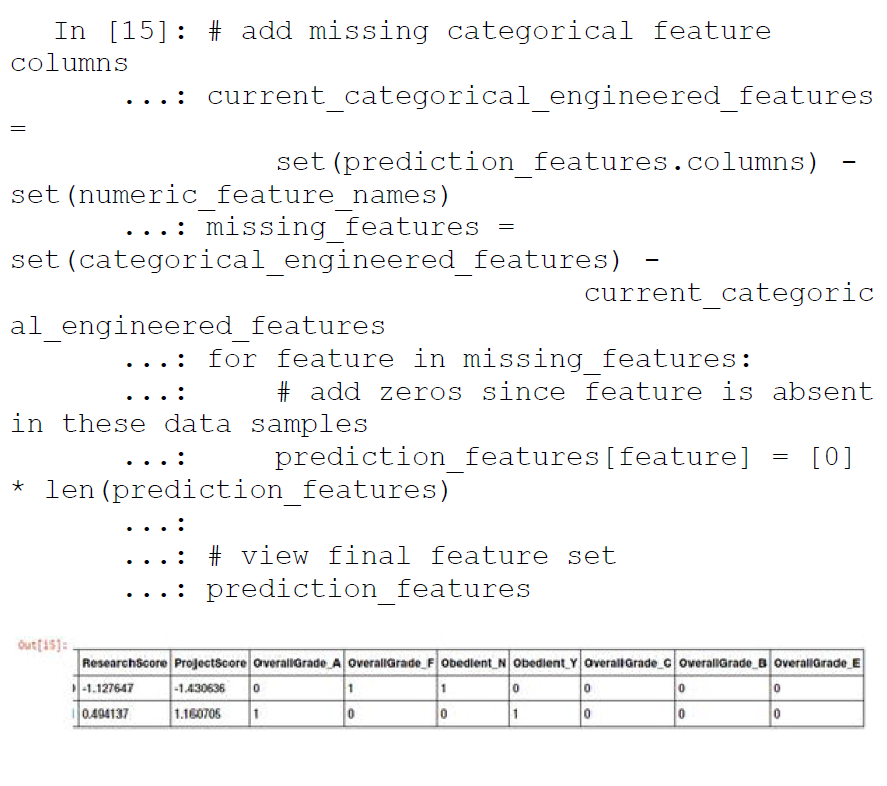


🡪 We will now carry out the tasks relevant to data preparation—feature extraction, engineering, and scaling—in the following code snippet.

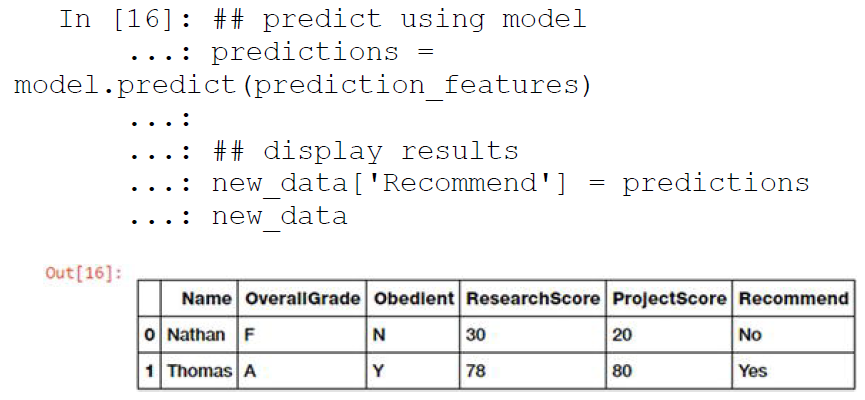




🡪 We now have the relevant features for the new students! However you can see that some of the categorical features are missing based on some grades like B, C, and E. This is because none of these students obtained those grades but we still need those attributes because the model was trained on all attributes including these. The following snippet helps us identify and add the missing categorical features. We add the value for each of those features as 0 for each student since they did not obtain those grades.



🡪 We have our complete feature set ready for both the new students. Let’s put our model to the test and get the predictions with regard to grant recommendations!



🡪 We can clearly see from Figure above that our model has predicted grant recommendation labels for both the new students. Thomas clearly being diligent, having a straight A average and decent scores, is most likely to get the grant recommendation as compared to Nathan. Thus you can see that our model has learned how to predict grant recommendation outcomes based on past historical student data. This should whet your appetite on getting started with Machine Learning. We are about to deep dive into more complex real-world problems in the upcoming chapters!

1. Challenges In Machine Learning

🡪 Machine Learning is a rapidly evolving, fast-paced, and exciting field with a lot of prospect, opportunity, and scope. However it comes with its own set of challenges, due to the complex nature of Machine Learning methods, its dependency on data, and not being one of the more traditional computing paradigms. The following points cover some of the main challenges in Machine Learning.

* Data quality issues lead to problems, especially with regard to data processing and feature extraction.
* Data acquisition, extraction, and retrieval is an extremely tedious and time consuming process.
* Lack of good quality and sufficient training data in many scenarios.
* Formulating business problems clearly with well-defined goals and objectives.
* Feature extraction and engineering, especially hand-crafting features, is one of the most difficult yet important tasks in Machine Learning. Deep Learning seems to have gained some advantage in this area recently.
* Overfitting or underfitting models can lead to the model learning poor representations and relationships from the training data leading to detrimental performance.
* The curse of dimensionality: too many features can be a real hindrance.
* Complex models can be difficult to deploy in the real world.

🡪 This is not an exhaustive list of challenges faced in Machine Learning today, but it is definitely a list of the top problems data scientists or analysts usually face in Machine Learning projects and tasks. We will cover dealing with these issues in detail when we discuss more about the various stages in the Machine Learning pipeline as well as solve real-world problems in subsequent sections.

1. Real-World Applications Of Machine Learning

🡪 Machine Learning is widely being applied and used in the real world today to solve complex problems that would otherwise have been impossible to solve based on traditional approaches and rule-based systems. The following list depicts some of the real-world applications of Machine Learning.

* Product recommendations in online shopping platforms
* Sentiment and emotion analysis
* Anomaly detection
* Fraud detection and prevention
* Content recommendation (news, music, movies, and so on)
* Weather forecasting
* Stock market forecasting
* Market basket analysis
* Customer segmentation
* Object and scene recognition in images and video
* Speech recognition
* Churn analytics
* Click through predictions
* Failure/defect detection and prevention
* E-mail spam filtering