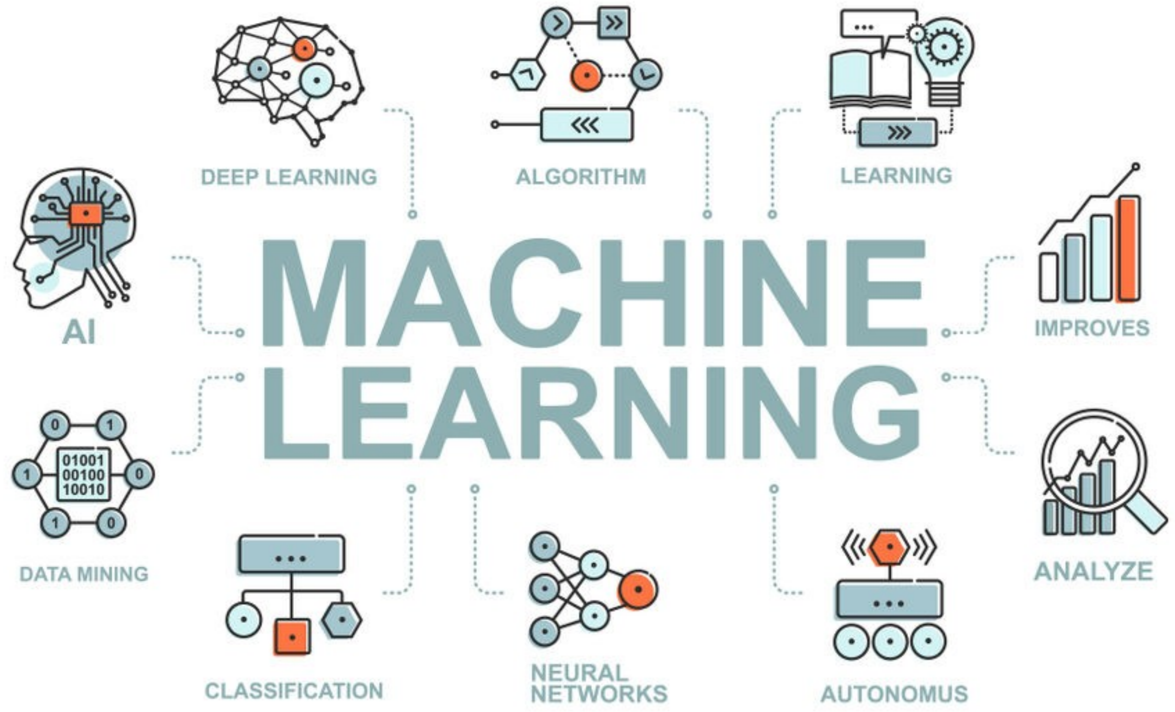
MACHINE LEARNING

Surabhi Rastogi

Machine Learning

- Machine Learning (ML) is a subfield of Artificial Intelligence (AI). It deals with training learning algorithms to deal with all sorts of predictive problems and tasks.
- Machine learning is a growing technology which enables computers to learn automatically from past data.
- Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information.
- Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.

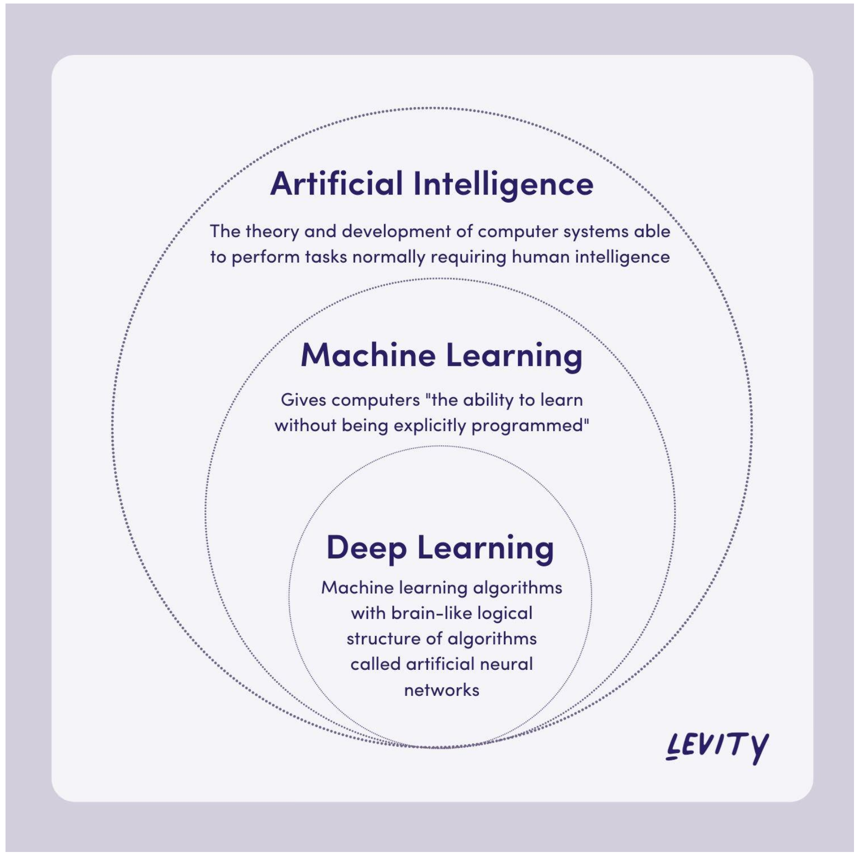
Machine Learning



Machine Learning(Contd.)

- A Machine Learning system **learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it.**
- The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.
- Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output.

AI v/s ML v/s DL


Artificial Intelligence
The theory and development of computer systems able to perform tasks normally requiring human intelligence
Machine Learning
Gives computers "the ability to learn without being explicitly programmed"
Deep Learning
Machine learning algorithms with brain-like logical structure of algorithms called artificial neural networks
LEVITY

Features of Machine Learning

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

Need/Importance of Machine Learning

- Rapid increment in the production of data
- Solving complex problems, which are difficult for a human
- Decision making in various sector including finance
- Finding hidden patterns and extracting useful information from data.

Develop a learning system

Developing a learning system involves several stages, from planning and design to implementation and evaluation. Whether you’re creating an educational platform, a corporate training module, or an adaptive learning system, a systematic approach will ensure effectiveness and scalability. Here’s a comprehensive guide to developing a learning system:

1. Define Objectives and Scope

- **Identify Goals**: Determine what the learning system aims to achieve. Goals could include improving skills, increasing knowledge, or achieving compliance.
- **Target Audience**: Define who will use the system (e.g., students, employees, or healthcare professionals). Understand their needs, preferences, and existing knowledge.
- **Scope and Requirements**: Outline the content, features, and functionalities required. Decide on the breadth and depth of the learning materials.

Develop a learning system(Contd.)

2. Conduct Needs Analysis

- **Assess Learning Needs**: Conduct surveys, interviews, or focus groups to identify the learning needs and preferences of your target audience.
- **Gap Analysis**: Evaluate existing resources and identify gaps that the new learning system should address.

3. Design the Learning System

- **Learning Objectives**: Develop clear, measurable learning objectives that align with the overall goals.
- **Content Structure**: Organize content into modules or units. Define the sequence and how each piece will build on previous knowledge.
- **Instructional Strategies**: Choose instructional methods (e.g., lectures, discussions, hands-on activities) and learning theories (e.g., constructivism, behaviorism) that best suit your objectives.

Develop a learning system(Contd.)

4. Select Technology and Tools

- **Learning Management System (LMS)**: Choose an LMS that fits your needs. Popular options include Moodle, Blackboard, Canvas, and proprietary solutions.
- **Authoring Tools**: Use tools to create content, such as Articulate Storyline, Adobe Captivate, or Lectora.
- **Multimedia and Interactive Elements**: Incorporate videos, quizzes, simulations, and interactive elements to enhance engagement.
- **Integration**: Ensure compatibility with existing systems (e.g., HR systems for corporate training).

5. Develop Content

- **Create Materials**: Develop or source content such as text, graphics, videos, and audio. Ensure materials are accurate, relevant, and engaging.
- **Interactive Elements**: Design quizzes, simulations, and other interactive components to reinforce learning and assess comprehension.
- **Accessibility**: Ensure content is accessible to all users, including those with disabilities (e.g., through captions, screen readers, and alternative text).

Develop a learning system(Contd.)

6. Build and Test the System

- **System Development**: Develop the system based on the design and technology chosen. This includes setting up the LMS, uploading content, and configuring features.
- **Testing**: Conduct thorough testing to identify and fix bugs or issues. Test with a small group of users to gather feedback and make improvements.

7. Implement the System

- **Launch**: Roll out the system to the broader audience. This might involve a phased approach or a full launch.
- **Training and Support**: Provide training for users on how to use the system effectively. Offer ongoing support to address any issues that arise.

Develop a learning system(Contd.)

8. Evaluate and Improve

- **Gather Feedback**: Collect feedback from users through surveys, focus groups, or usage data. Assess how well the system meets learning objectives and user needs.
- **Analyze Data**: Review data on user engagement, performance, and system functionality. Identify areas for improvement.
- **Continuous Improvement**: Make iterative improvements based on feedback and data. Update content and features as needed.

9. Ensure Compliance and Security

- **Data Privacy**: Implement measures to protect user data and ensure compliance with relevant regulations (e.g., GDPR, FERPA).
- **Security Measures**: Ensure the system is secure from unauthorized access and cyber threats.

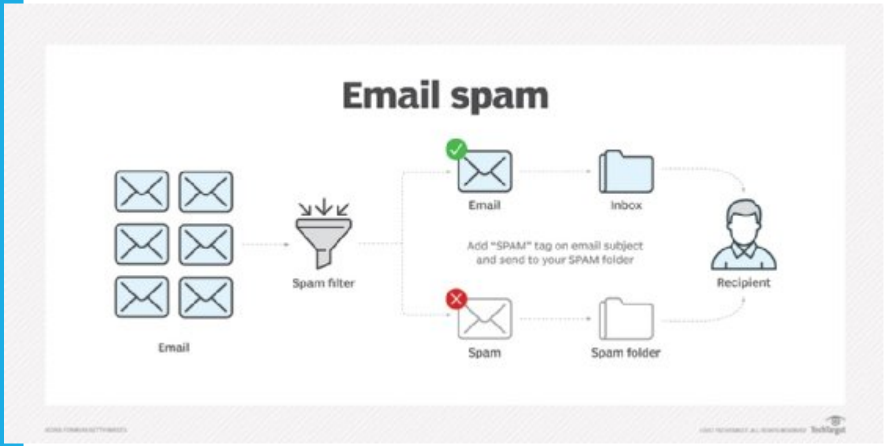
Develop a learning system(Contd.)

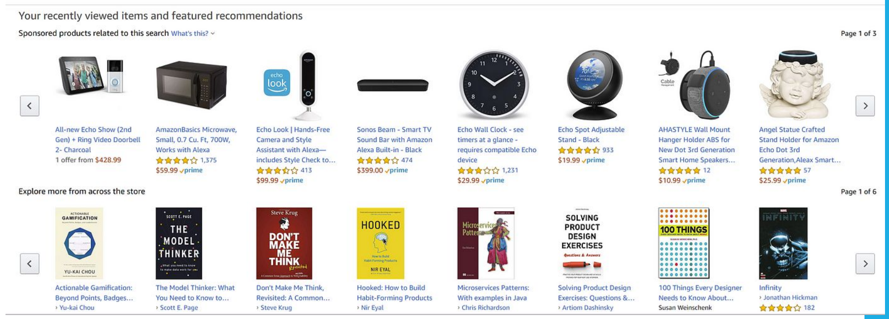
10. Scale and Adapt

- **Scaling**: Plan for scalability if the system needs to support a larger audience or additional content in the future.
- **Adaptation**: Stay adaptable to changing needs and technological advancements. Continuously evolve the system to keep it relevant and effective.

Machine Learning Applications

- Machine Learning is used in various applications such as ***email filtering, speech recognition, computer vision, self-driven cars, Amazon product recommendation*****, etc.**


Email spam
Email
Spam filter
Email
Inbox
Add "SPAM" tag on email subject and send to your SPAM folder
Recipient
Spam
Spam folder



Your recently viewed items and featured recommendations

Sponsored products related to this search What's this?

Page 1 of 3

All-new Echo Show (2nd Gen) + Ring Video Doorbell 2- Charcoal
1 offer from $428.99

AmazonBasics Microwave, Small, 0.7 Cu. Ft, 700W, Works with Alexa
1,375
$59.99 prime

Echo Look | Hands-Free Camera and Style Assistant with Alexa—includes Style Check to...
413
$99.99 prime

Sonos Beam - Smart TV Sound Bar with Amazon Alexa Built-in - Black
474
$399.00 prime

Echo Wall Clock - see timers at a glance - requires compatible Echo device
1,231
$29.99 prime

Echo Spot Adjustable Stand - Black
933
$19.99 prime

AHASTYLE Wall Mount Hanger Holder ABS for New Dot 3rd Generation Smart Home Speakers...
12
$10.99 prime

Angel Statue Crafted Stand Holder for Amazon Echo Dot 3rd Generation,Aleax Smart...
57
$25.99 prime

Explore more from across the store

Page 1 of 6

Actionable Gamification: Beyond Points, Badges...
› Yu-kai Chou

The Model Thinker: What You Need to Know to...
› Scott E. Page

Don't Make Me Think, Revisited: A Common...
› Steve Krug

Hooked: How to Build Habit-Forming Products
› Nir Eyal

Microservices Patterns: With examples in Java
› Chris Richardson

Solving Product Design Exercises: Questions &...
› Artiom Dashinsky

100 Things Every Designer Needs to Know About...
Susan Weinschenk

Infinity
› Jonathan Hickman
182

Issues in Machine Learning

1. Inadequate Training Data

- The major issue that comes while using machine learning algorithms is the lack of quality as well as quantity of data. Although data plays a vital role in the processing of machine learning algorithms, many data scientists claim that inadequate data, noisy data, and unclean data are extremely exhausting the machine learning algorithms. For example, a simple task requires thousands of sample data, and an advanced task such as speech or image recognition needs millions of sample data examples. Further, data quality is also important for the algorithms to work ideally, but the absence of data quality is also found in Machine Learning applications. Data quality can be affected by some factors as follows:
- **Noisy Data-** It is responsible for an inaccurate prediction that affects the decision as well as accuracy in classification tasks.
- **Incorrect data-** It is also responsible for faulty programming and results obtained in machine learning models. Hence, incorrect data may affect the accuracy of the results also.
- **Generalizing of output data-** Sometimes, it is also found that generalizing output data becomes complex, which results in comparatively poor future actions.

Issues in Machine Learning(Contd.)

2. Poor quality of data

- As we have discussed above, data plays a significant role in machine learning, and it must be of good quality as well. Noisy data, incomplete data, inaccurate data, and unclean data lead to less accuracy in classification and low-quality results. Hence, data quality can also be considered as a major common problem while processing machine learning algorithms.

Issues in Machine Learning(Contd.)

3. Non-representative training data

- To make sure our training model is generalized well or not, we have to ensure that sample training data must be representative of new cases that we need to generalize. The training data must cover all cases that are already occurred as well as occurring.
- Further, if we are using non-representative training data in the model, it results in less accurate predictions. A machine learning model is said to be ideal if it predicts well for generalized cases and provides accurate decisions. If there is less training data, then there will be a sampling noise in the model, called the non-representative training set. It won't be accurate in predictions. To overcome this, it will be biased against one class or a group.

Issues in Machine Learning(Contd.)

4. Overfitting and Underfitting

Overfitting:

Overfitting is one of the most common issues faced by Machine Learning engineers and data scientists. Whenever a machine learning model is trained with a huge amount of data, it starts capturing noise and inaccurate data into the training data set. It negatively affects the performance of the model. Let's understand with a simple example where we have a few training data sets such as 1000 mangoes, 1000 apples, 1000 bananas, and 5000 papayas. Then there is a considerable probability of identification of an apple as papaya because we have a massive amount of biased data in the training data set; hence prediction got negatively affected. The main reason behind overfitting is using non-linear methods used in machine learning algorithms as they build non-realistic data models. We can overcome overfitting by using linear and parametric algorithms in the machine learning models.

Methods to reduce overfitting:

- Increase training data in a dataset.
- Reduce model complexity by simplifying the model by selecting one with fewer parameters
- Ridge Regularization and Lasso Regularization
- Early stopping during the training phase
- Reduce the noise
- Reduce the number of attributes in training data.
- Constraining the model.

Issues in Machine Learning(Contd.)

Underfitting:

- Underfitting is just the opposite of overfitting. Whenever a machine learning model is trained with fewer amounts of data, and as a result, it provides incomplete and inaccurate data and destroys the accuracy of the machine learning model.
- Underfitting occurs when our model is too simple to understand the base structure of the data, just like an undersized pant. This generally happens when we have limited data into the data set, and we try to build a linear model with non-linear data. In such scenarios, the complexity of the model destroys, and rules of the machine learning model become too easy to be applied on this data set, and the model starts doing wrong predictions as well.

Issues in Machine Learning(Contd.)

Methods to reduce Underfitting:

- Increase model complexity
- Remove noise from the data
- Trained on increased and better features
- Reduce the constraints
- Increase the number of epochs to get better results.

Issues in Machine Learning(Contd.)

5. Monitoring and maintenance

- As we know that generalized output data is mandatory for any machine learning model; hence, regular monitoring and maintenance become compulsory for the same. Different results for different actions require data change; hence editing of codes as well as resources for monitoring them also become necessary.

6. Getting bad recommendations

- A machine learning model operates under a specific context which results in bad recommendations and concept drift in the model. Let's understand with an example where at a specific time customer is looking for some gadgets, but now customer requirement changed over time but still machine learning model showing same recommendations to the customer while customer expectation has been changed. This incident is called a Data Drift. It generally occurs when new data is introduced or interpretation of data changes. However, we can overcome this by regularly updating and monitoring data according to the expectations.

Issues in Machine Learning(Contd.)

7. Lack of skilled resources

- Although Machine Learning and Artificial Intelligence are continuously growing in the market, still these industries are fresher in comparison to others. The absence of skilled resources in the form of manpower is also an issue. Hence, we need manpower having in-depth knowledge of mathematics, science, and technologies for developing and managing scientific substances for machine learning.

8. Customer Segmentation

- Customer segmentation is also an important issue while developing a machine learning algorithm. To identify the customers who paid for the recommendations shown by the model and who don't even check them. Hence, an algorithm is necessary to recognize the customer behavior and trigger a relevant recommendation for the user based on past experience.

Issues in Machine Learning(Contd.)

9. Process Complexity of Machine Learning

- The machine learning process is very complex, which is also another major issue faced by machine learning engineers and data scientists. However, Machine Learning and Artificial Intelligence are very new technologies but are still in an experimental phase and continuously being changing over time.
- There is the majority of hits and trial experiments; hence the probability of error is higher than expected. Further, it also includes analyzing the data, removing data bias, training data, applying complex mathematical calculations, etc., making the procedure more complicated and quite tedious.

Issues in Machine Learning(Contd.)

10. Data Bias

Data Biasing is also found a big challenge in Machine Learning. These errors exist when certain elements of the dataset are heavily weighted or need more importance than others. Biased data leads to inaccurate results, skewed outcomes, and other analytical errors. However, we can resolve this error by determining where data is actually biased in the dataset. Further, take necessary steps to reduce it.

- **Methods to remove Data Bias:**
- Research more for customer segmentation.
- Be aware of your general use cases and potential outliers.
- Combine inputs from multiple sources to ensure data diversity.
- Include bias testing in the development process.
- Analyze data regularly and keep tracking errors to resolve them easily.
- Review the collected and annotated data.
- Use multi-pass annotation such as sentiment analysis, content moderation, and intent recognition.
-

Issues in Machine Learning(Contd.)

11. Lack of Explainability

- This basically means the outputs cannot be easily comprehended as it is programmed in specific ways to deliver for certain conditions. Hence, a lack of explainability is also found in machine learning algorithms which reduce the credibility of the algorithms.

12. Slow implementations and results

- This issue is also very commonly seen in machine learning models. However, machine learning models are highly efficient in producing accurate results but are time-consuming. Slow programming, excessive requirements' and overloaded data take more time to provide accurate results than expected. This needs continuous maintenance and monitoring of the model for delivering accurate results.

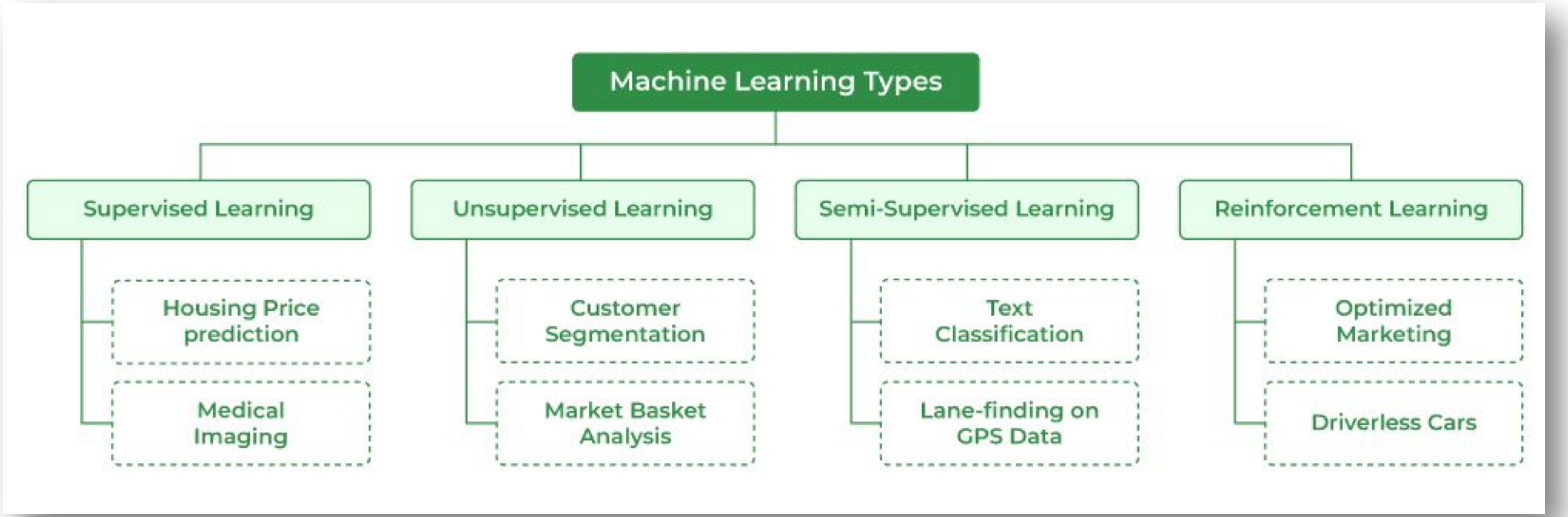
13. Irrelevant features

- Although machine learning models are intended to give the best possible outcome, if we feed garbage data as input, then the result will also be garbage. Hence, we should use relevant features in our training sample. A machine learning model is said to be good if training data has a good set of features or less to no irrelevant features.

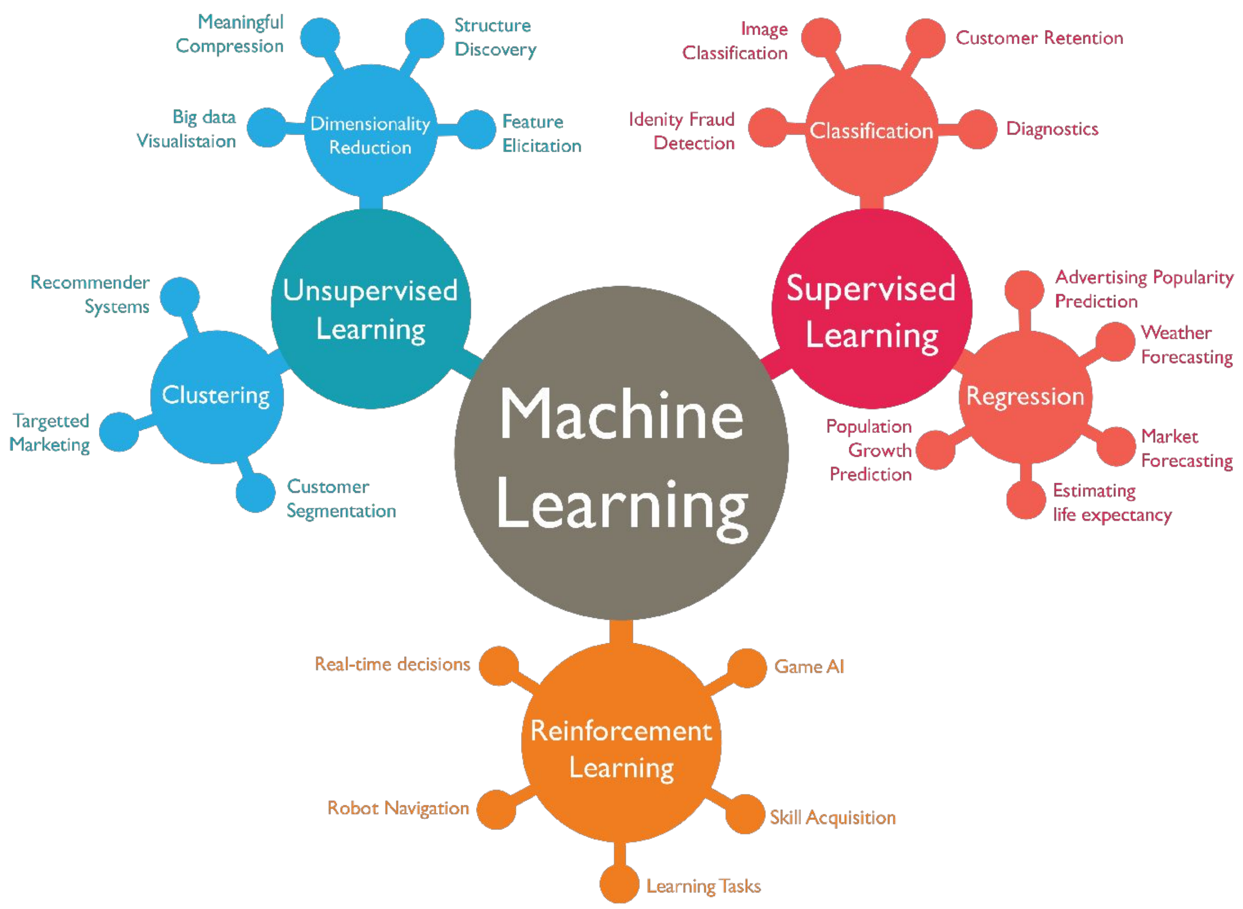
Issues in Machine Learning(Contd.)

- Machine learning is all set to bring a big bang transformation in technology.
- It is one of the most rapidly growing technologies used in medical diagnosis, speech recognition, robotic training, product recommendations, video surveillance, and this list goes on.
- This continuously evolving domain offers immense job satisfaction, excellent opportunities, global exposure, and exorbitant salary. It is a high risk and a high return technology.

Types of Machine Learning



Types of Machine Learning

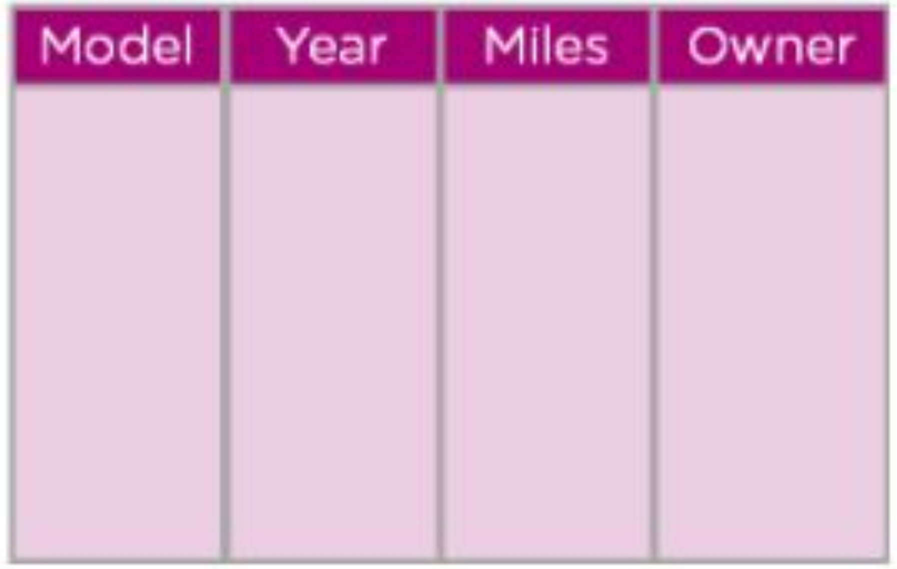


Why do we need Feature Selection?

- Machine learning models follow a simple rule: **whatever goes in, comes out**. If we put garbage into our model, we can expect the output to be garbage too. In this case, garbage refers to noise in our data.
- To train a model, we collect enormous quantities of data to help the machine learn better. Usually, a good portion of the data collected is noise, while some of the columns of our dataset might not contribute significantly to the performance of our model. Further, having a lot of data can slow down the training process and cause the model to be slower. The model may also learn from this irrelevant data and be inaccurate.
- Feature selection is what separates good data scientists from the rest. Given the same model and computational facilities, why do some people win in competitions with faster and more accurate models? The answer is **Feature Selection**. Apart from choosing the right model for our data, we need to choose the right data to put in our model.

Feature Selection Importance with Example

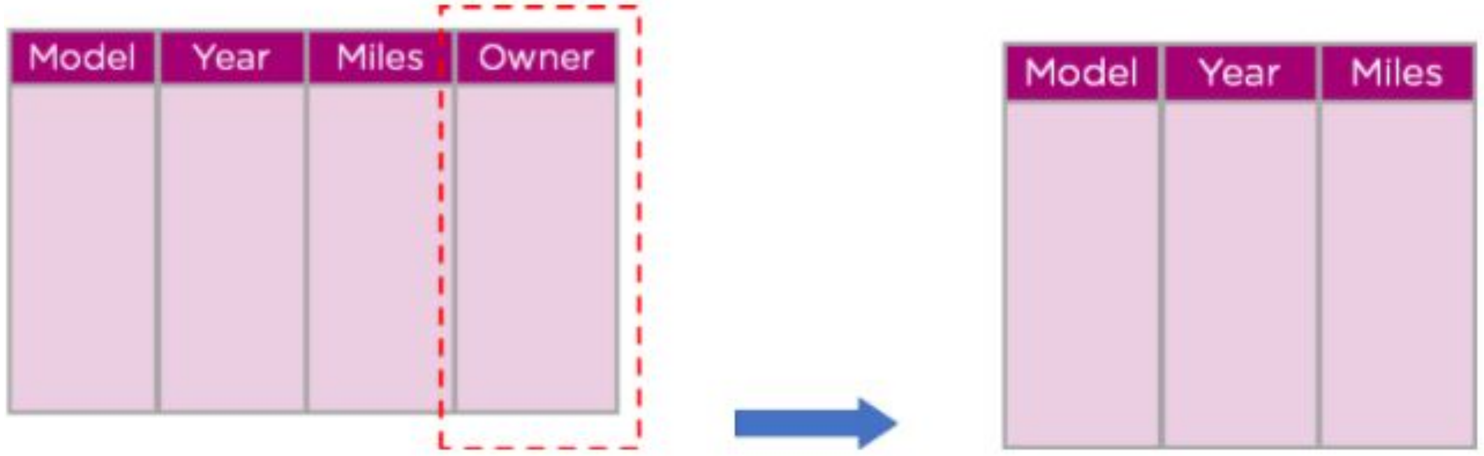
- Consider a table which contains information on old cars. The model decides which cars must be crushed for spare parts.



Model	Year	Miles	Owner

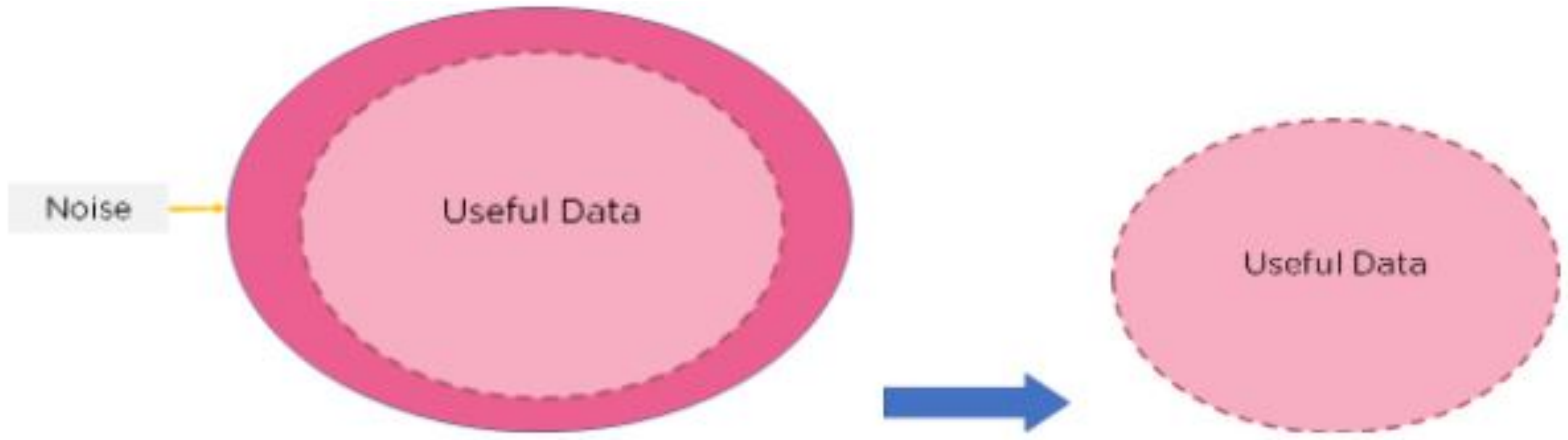
Feature Selection Importance with Example

- In the previous table, we can see that the model of the car, the year of manufacture, and the miles it has travelled are very important to find out if the car is old enough to be crushed or not. However, the name of the previous owner of the car does not decide if the car should be crushed or not. Further, it can confuse the algorithm into finding patterns between names and the other features. Hence we can drop the column.



What is feature Selection?

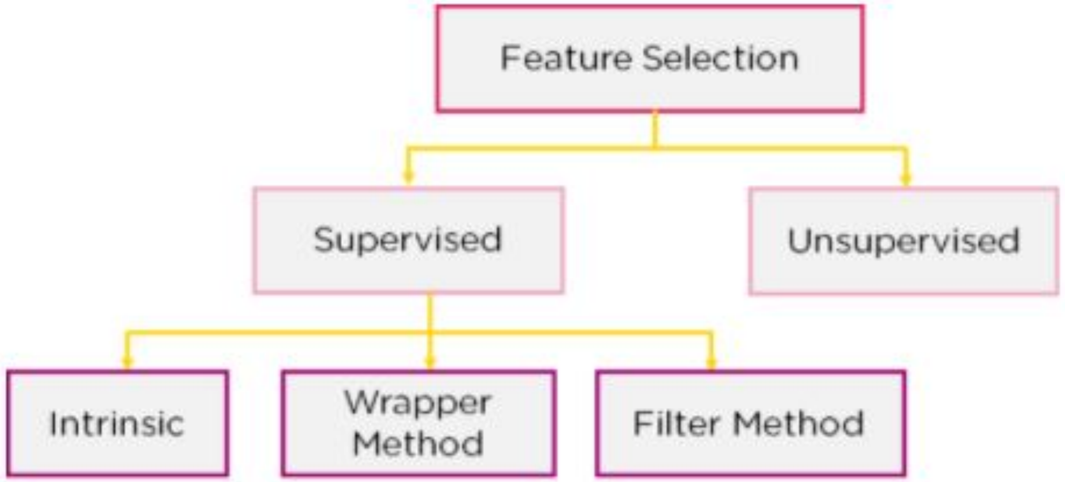
- Feature Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data.
- It is the process of automatically choosing relevant features for your machine learning model based on the type of problem you are trying to solve. We do this by including or excluding important features without changing them. It helps in cutting down the noise in our data and reducing the size of our input data.



Feature Selection Models

Feature selection models are of two types:

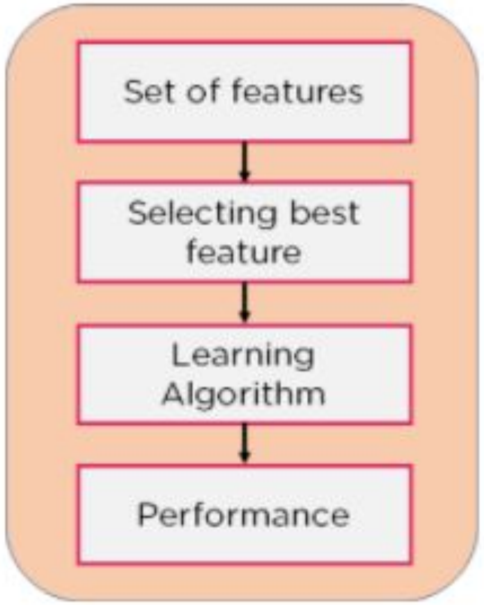
1. **Supervised Models:** Supervised feature selection refers to the method which uses the output label class for feature selection. They use the target variables to identify the variables which can increase the efficiency of the model.
2. **Unsupervised Models:** Unsupervised feature selection refers to the method which does not need the output label class for feature selection. We use them for unlabeled data.



Feature Selection Models(Contd.)

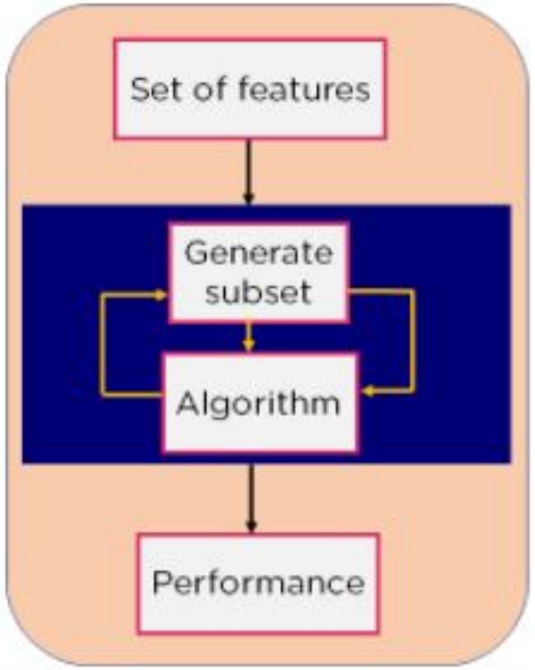
We can further divide the supervised models into three :

1. Filter Method: In this method, features are dropped based on their relation to the output, or how they are correlating to the output. We use correlation to check if the features are positively or negatively correlated to the output labels and drop features accordingly. E.g. Information Gain, Chi-Square Test, Fisher's Score, etc.



Feature Selection Models(Contd.)

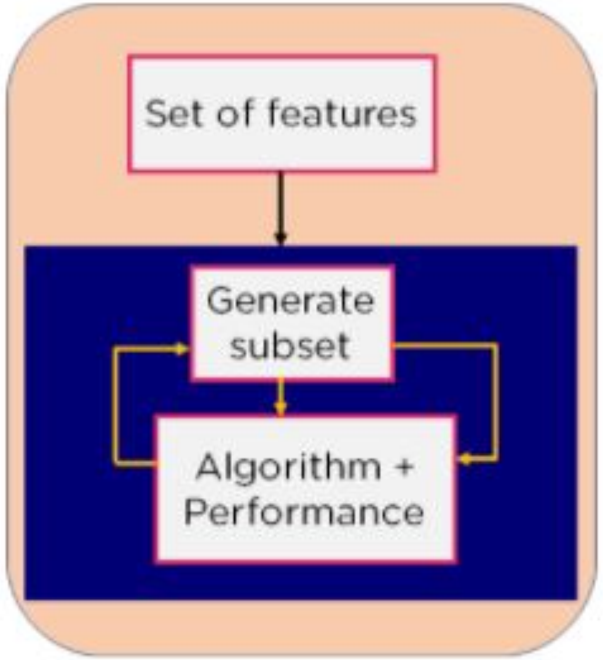
2. Wrapper Method: We split our data into subsets and train a model using this. Based on the output of the model, we add and subtract features and train the model again. It forms the subsets using a greedy approach and evaluates the accuracy of all the possible combinations of features. E.g. Forward Selection, Backwards Elimination, etc.



Feature Selection Models(Contd.)

3. Intrinsic Method: This method combines the qualities of both the Filter and Wrapper method to create the best subset.

This method takes care of the machine training iterative process while maintaining the computation cost to be minimum. E.g. Lasso and Ridge Regression.



How to Choose a Feature Selection Model?

The process is relatively simple, with the model depending on the types of input and output variables.

Variables are of two main types:

- **Numerical Variables:** Which include integers, float, and numbers.
- **Categorical Variables:** Which include labels, strings, Boolean variables, etc.

How to Choose a Feature Selection Model?

Based on whether we have numerical or categorical variables as inputs and outputs, we can choose our feature selection model as follows:

Input Variable	Output Variable	Feature Selection Model
Numerical	Numerical	• Pearson’s correlation coefficient • Spearman’s rank coefficient
Numerical	Categorical	• ANOVA correlation coefficient (linear) • Kendall’s rank coefficient (nonlinear)
Categorical	Numerical	• Kendall’s rank coefficient (linear) • ANOVA correlation coefficient (nonlinear)
Categorical	Categorical	• Chi-Squared test (contingency tables) • Mutual Information

Imbalanced Data

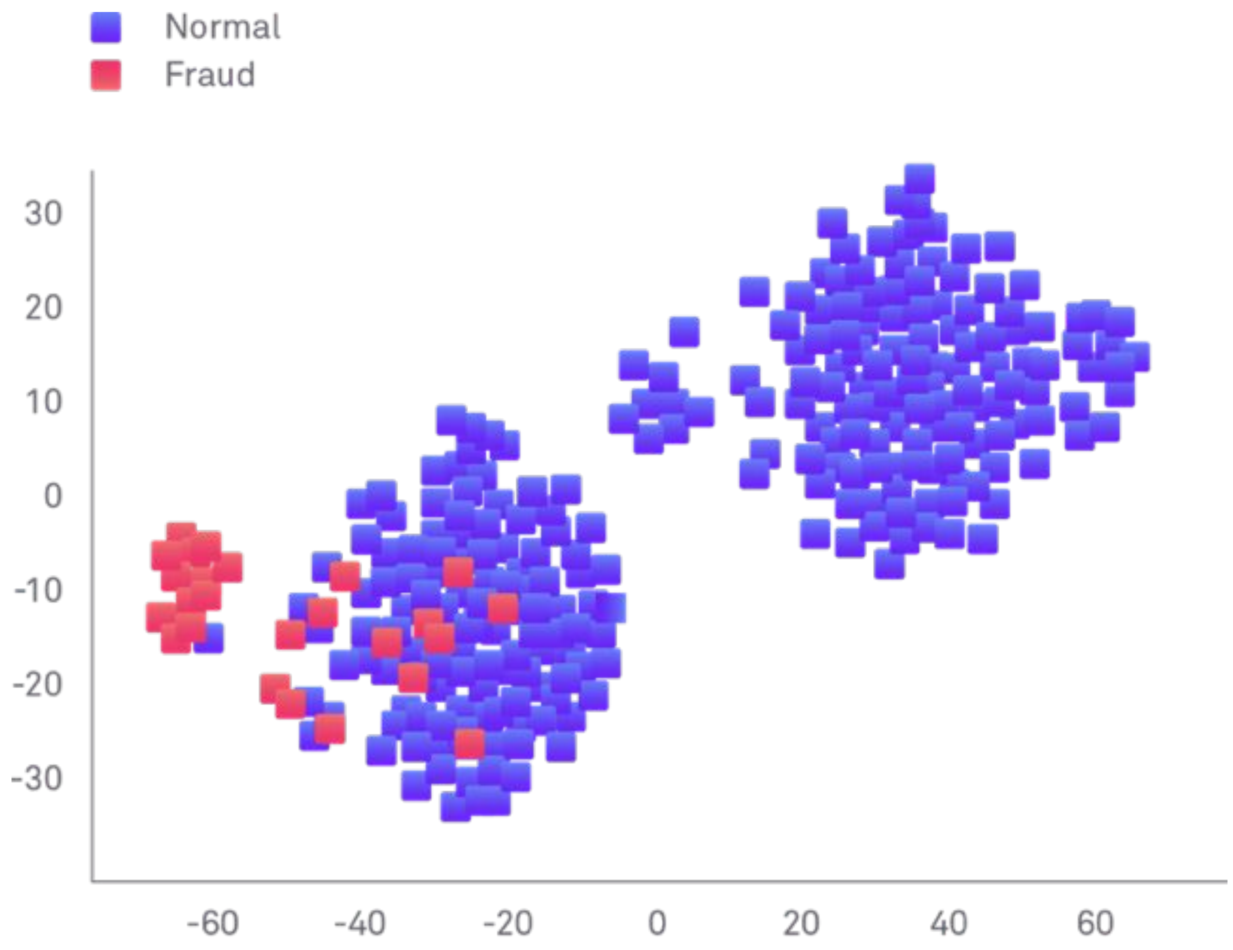
Imbalanced data is a term used to characterize certain types of datasets and represents a critical challenge associated with classification problems. It can be found in a myriad of applications including finance, healthcare, and public sectors. While no rigorous definition exists, it refers to a scenario where the number of samples associated with each class is highly variable. Let's consider the following example:

Example

You are a bank employee responsible for detecting the **validity of credit card transactions.** To do so, you have a training set of previously observed transactions, each of which was either:

- a) Normal
- b) Fraudulent
- Most transactions are normal and it is not unlikely that fraudulent account for **less than 0.1%** of the total transactions! Creating a model for this task can be tricky – considering only performance as given by an accuracy score, then a model always predicting "regular" will have a really high performance value!

Imbalanced Data



Bias in Data

- Bias in data is an error that occurs when certain elements of a dataset are overweighted or overrepresented. Biased datasets don't accurately represent ML model's use case, which leads to skewed outcomes, systematic prejudice, and low accuracy.
- Often, the erroneous result discriminates against a specific group or groups of people. For example, data bias reflects prejudice against age, race, culture, or sexual orientation. In a world where AI systems are increasingly used everywhere, the danger of bias lies in amplifying discrimination.
- It takes a lot of training data for machine learning models to produce viable results. If you want to perform advanced operations (such as text, image, or video recognition), you need millions of data points. Poor or incomplete data as well as biased data collection & analysis methods will result in inaccurate predictions because the quality of the outputs is determined by the quality of the inputs.

Types of Bias in Data

- **Reporting bias :** It occurs when the frequency of events, properties, and/or outcomes captured in a dataset does not accurately reflect their real-world frequency. This bias can arise because people tend to focus on documenting circumstances that are unusual or especially memorable, assuming that the ordinary does not need to be recorded.
- For example : A sentiment-analysis model is trained to predict whether book reviews are positive or negative based on a corpus of user submissions to a popular website. The majority of reviews in the training dataset reflect extreme opinions (reviewers who either loved or hated a book), because people were less likely to submit a review of a book if they did not respond to it strongly. As a result, the model is less able to correctly predict sentiment of reviews that use more subtle language to describe a book.

Types of Bias in Data

- **Historical bias :** It occurs when historical data reflects inequities that existed in the world at that time.
- **Example :** A city housing dataset from the 1960s contains home-price data that reflects discriminatory lending practices in effect during that decade.

Types of Bias in Data

- **Automation bias :** It is a tendency to favor results generated by automated systems over those generated by non-automated systems, irrespective of the error rates of each.
- **Example :** ML practitioners working for a sprocket manufacturer were eager to deploy the new "groundbreaking" model they trained to identify tooth defects, until the factory supervisor pointed out that the model's precision and recall rates were both 15% lower than those of human inspectors.

Types of Bias in Data

- **Selection bias :** It occurs if a dataset's examples are chosen in a way that is not reflective of their real-world distribution. Selection bias can take many different forms, including coverage bias, non-response bias, and sampling bias.
  - **Coverage bias :** It occurs if data is not selected in a representative fashion.
  - **Example :** A model is trained to predict future sales of a new product based on phone surveys conducted with a sample of consumers who bought the product. Consumers who instead opted to buy a competing product were not surveyed, and as a result, this group of people was not represented in the training data.

Types of Bias in Data

- **Non-response bias (also knows as participation bias) :** It occurs if data ends up being unrepresentative due to participation gaps in the data-collection process.
- **Example :** A model is trained to predict future sales of a new product based on phone surveys conducted with a sample of consumers who bought the product and with a sample of consumers who bought a competing product. Consumers who bought the competing product were 80% more likely to refuse to complete the survey, and their data was underrepresented in the sample.

Types of Bias in Data

- **Sampling bias :** It occurs if proper randomization is not used during data collection.
- **Example :** A model is trained to predict future sales of a new product based on phone surveys conducted with a sample of consumers who bought the product and with a sample of consumers who bought a competing product. Instead of randomly targeting consumers, the surveyor chose the first 200 consumers that responded to an email, who might have been more enthusiastic about the product than average purchasers.

Types of Bias in Data

- **In-group bias :** It is a preference for members of your own group you also belong, or for characteristics that you also share.
- **Example :** Two ML practitioners training a résumé-screening model for software developers are predisposed to believe that applicants who attended the same computer-science academy as they both did are more qualified for the role.

Types of Bias in Data

- **Out-group homogeneity bias :** It is a tendency to stereotype individual members of a group to which you do not belong, or to see their characteristics as more uniform.
- **Example :** Two ML practitioners training a résumé-screening model for software developers are predisposed to believe that all applicants who did not attend a computer-science academy don't have sufficient expertise for the role.

Types of Bias in Data

- **Implicit bias :** It occurs when assumptions are made based on one's own model of thinking and personal experiences that don't necessarily apply more generally.
- **Example :** An ML practitioner training a gesture-recognition model uses a head shake as a feature to indicate a person is communicating the word "no." However, in some regions of the world, a head shake actually signifies "yes."

Types of Bias in Data

- **Confirmation bias :** It occurs when model builders unconsciously process data in ways that affirm pre-existing beliefs and hypotheses.
- **Example :** An ML practitioner is building a model that predicts aggressiveness in dogs based on a variety of features (height, weight, breed, environment). The practitioner had an unpleasant encounter with a hyperactive toy poodle as a child, and ever since has associated the breed with aggression. When curating the model's training data, the practitioner unconsciously discarded features that provided evidence of docility in smaller dogs.

Types of Bias in Data

- **Experimenter's bias :** It occurs when a model builder keeps training a model until it produces a result that aligns with their original hypothesis.
- **Example :** An ML practitioner is building a model that predicts aggressiveness in dogs based on a variety of features (height, weight, breed, environment). The practitioner had an unpleasant encounter with a hyperactive toy poodle as a child, and ever since has associated the breed with aggression. When the trained model predicted most toy poodles to be relatively docile, the practitioner retrained the model several more times until it produced a result showing smaller poodles to be more violent.

Outliers

- An outlier is a data point that significantly deviates from the rest of the data. It can be either much higher or much lower than the other data points, and its presence can have a significant impact on the results of machine learning algorithms. They can be caused by measurement or execution errors. The analysis of outlier data is referred to as outlier analysis or outlier mining.
- Types of Outliers

There are two main types of outliers:

- **Global outliers:** Global outliers are isolated data points that are far away from the main body of the data. They are often easy to identify and remove.
- **Contextual outliers:** Contextual outliers are data points that are unusual in a specific context but may not be outliers in a different context. They are often more difficult to identify and may require additional information or domain knowledge to determine their significance.

Outlier Detection

- 1. **Removal:**

This involves identifying and removing outliers from the dataset before training the model. Common methods include:

- Thresholding: Outliers are identified as data points exceeding a certain threshold (e.g., Z-score > 3).
- Distance-based methods: Outliers are identified based on their distance from their nearest neighbors.
- Clustering: Outliers are identified as points not belonging to any cluster or belonging to very small clusters.

2. **Transformation:**

This involves transforming the data to reduce the influence of outliers. Common methods include:

- Scaling: Standardizing or normalizing the data to have a mean of zero and a standard deviation of one.
- Winsorization: Replacing outlier values with the nearest non-outlier value.
- Log transformation: Applying a logarithmic transformation to compress the data and reduce the impact of extreme values.

Outlier Detection

3. **Robust Estimation:**

This involves using algorithms that are less sensitive to outliers. Some examples include:

- Robust regression: Algorithms like L1-regularized regression or Huber regression are less influenced by outliers than least squares regression.
- M-estimators: These algorithms estimate the model parameters based on a robust objective function that down weights the influence of outliers.
- Outlier-insensitive clustering algorithms: Algorithms like DBSCAN are less susceptible to the presence of outliers than K-means clustering.

4. **Modeling Outliers:**

This involves explicitly modeling the outliers as a separate group. This can be done by:

- Adding a separate feature: Create a new feature indicating whether a data point is an outlier or not.
- Using a mixture model: Train a model that assumes the data comes from a mixture of multiple distributions, where one distribution represents the outliers.

MACHINE LEARNING

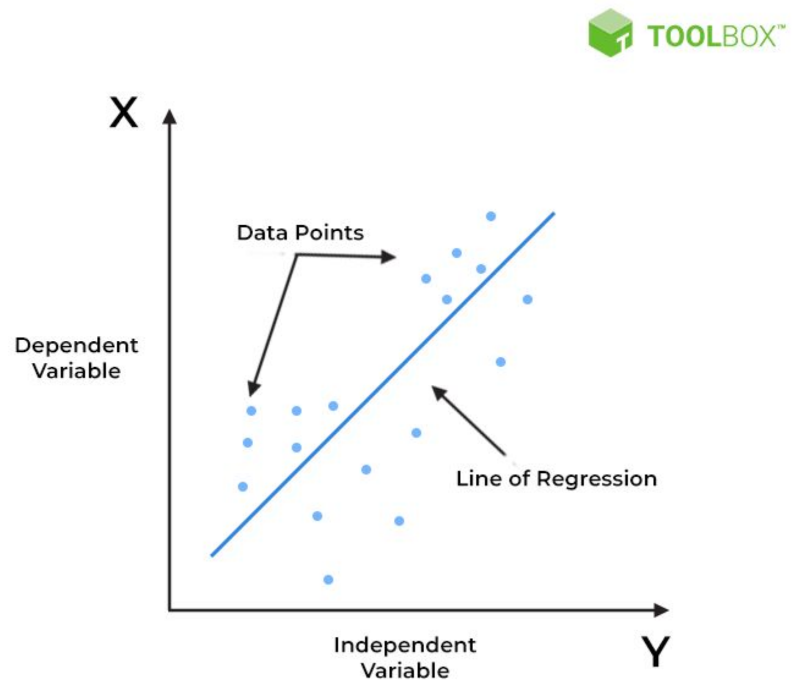
Surabhi Rastogi

Linear Regression

- Linear regression is defined as an algorithm that provides a linear relationship between an independent variable and a dependent variable to predict the outcome of future events.
- It is a statistical method used in data science and machine learning for predictive analysis.
- The independent variable is also the predictor or explanatory variable that remains unchanged due to the change in other variables. However, the dependent variable changes with fluctuations in the independent variable. The regression model predicts the value of the dependent variable, which is the response or outcome variable being analyzed or studied.
- Thus, linear regression is a supervised learning algorithm that simulates a mathematical relationship between variables and makes predictions for continuous or numeric variables such as sales, salary, age, product price, etc.

Linear Regression

- This analysis method is advantageous when at least two variables are available in the data, as observed in stock market forecasting, portfolio management, scientific analysis, etc.



Linear Regression

- In the previous figure,
- X-axis = Independent variable
- Y-axis = Output / dependent variable
- Line of regression = Best fit line for a model
- Here, a line is plotted for the given data points that suitably fit all the issues. Hence, it is called the ‘best fit line.’ The goal of the linear regression algorithm is to find this best fit line seen in the above figure.

Linear Regression

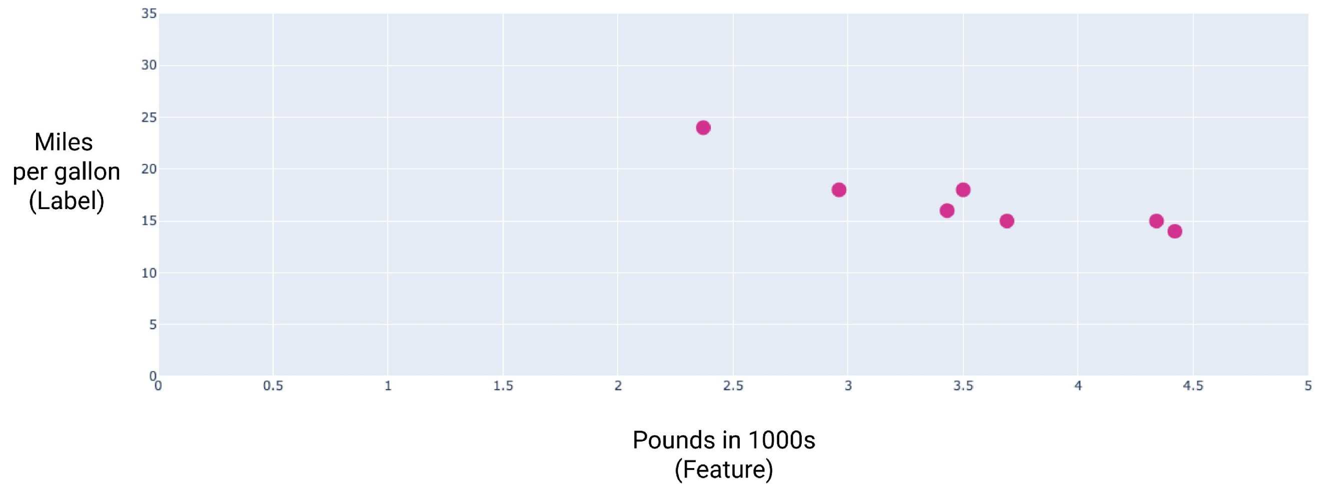
- For example, suppose we want to predict a car's fuel efficiency in miles per gallon based on how heavy the car is, and we have the following dataset:

Pounds in 1000s (feature)	Miles per gallon (label)
3.5	18
3.69	15
3.44	18
3.43	16
4.34	15
4.42	14
2.37	24

Linear Regression

- If we plotted these points, we'd get the following graph:

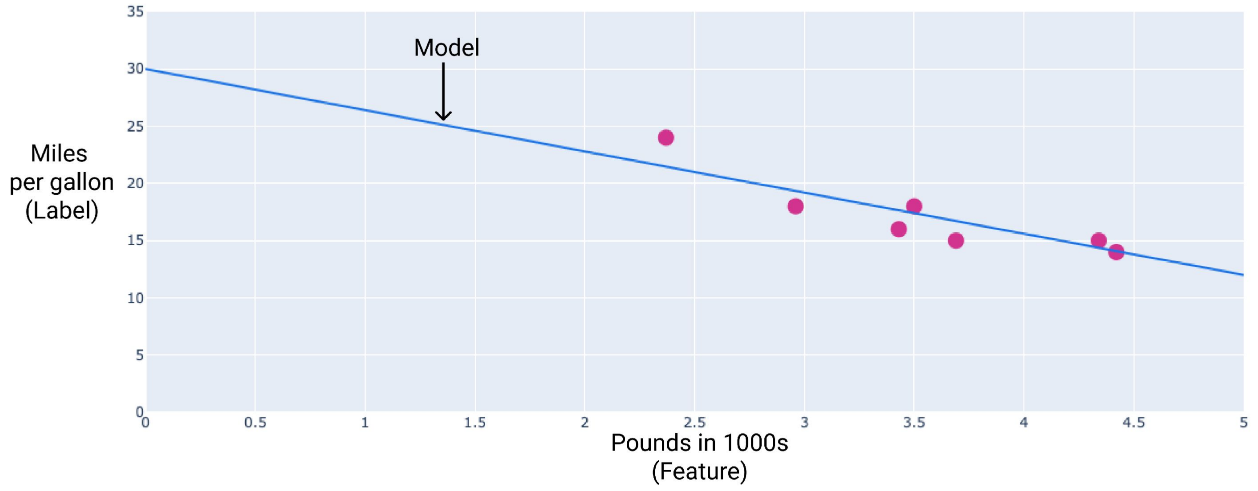
Figure 1 : Car heaviness (in pounds) versus miles per gallon rating. As a car gets heavier, its miles per gallon rating generally decreases.



Linear Regression

- We could create our own model by drawing a best fit line through the points:

Figure 2 : A best fit line drawn through the data from the previous figure.



Linear Regression Equation

- In algebraic terms, the model would be defined as : $y = mx + b$

where

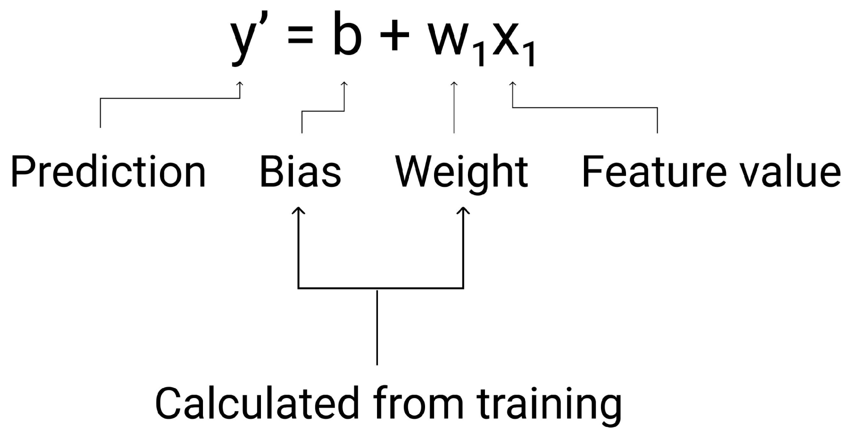
y is miles per gallon—the value we want to predict.

m is the slope of the line.

x is pounds—our input value.

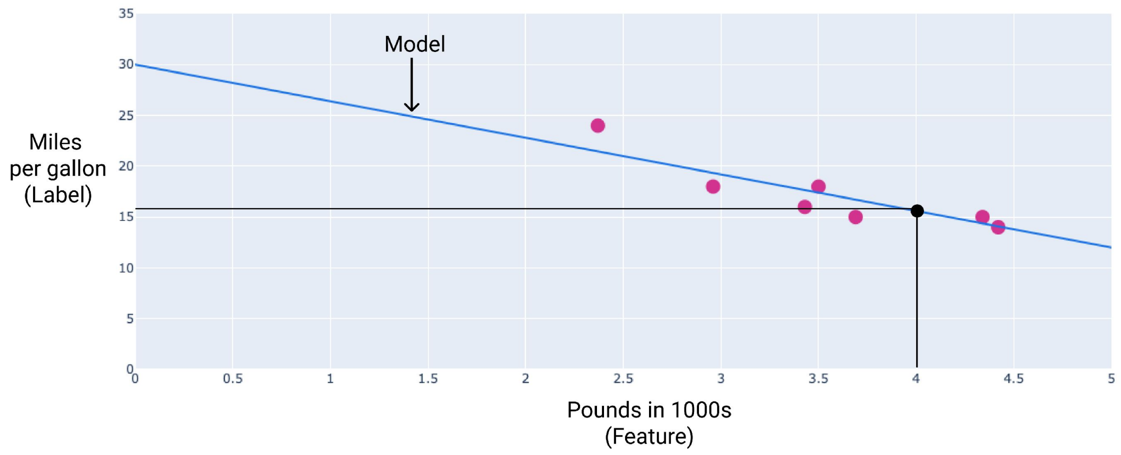
b is the y-intercept.

- In ML, we write the equation for a linear regression model as follows:



Linear regression equation

- In our example, we'd calculate the weight and bias from the line we drew. The bias is 30 (where the line intersects the y-axis), and the weight is -3.6 (the slope of the line). The model would be defined as y = 30 + (-3.6) (x1) and we could use it to make predictions. For instance, using this model, a 4,000-pound car would have a predicted fuel efficiency of 15.6 miles per gallon.



Multiple Regression

- Multiple linear regression is used to estimate the relationship between two or more independent variables and one dependent variable. You can use multiple linear regression when you want to know:
- How strong the relationship is between two or more independent variables and one dependent variable (e.g. how rainfall, temperature, and amount of fertilizer added affect crop growth).
- The value of the dependent variable at a certain value of the independent variables (e.g. the expected yield of a crop at certain levels of rainfall, temperature, and fertilizer addition).
- Example: Consider the task of calculating blood pressure. In this case, height, weight, and amount of exercise can be considered independent variables. Here, we can use multiple linear regression to analyze the relationship between the three independent variables and one dependent variable, as all the variables considered are quantitative.

Multiple Linear Regression Example

- You are a public health researcher interested in social factors that influence heart disease. You survey 500 towns and gather data on the percentage of people in each town who smoke, the percentage of people in each town who bike to work, and the percentage of people in each town who have heart disease.
- Because you have two independent variables and one dependent variable, and all your variables are quantitative, you can use multiple linear regression to analyze the relationship between them.

Assumptions of Linear Regression

- Multiple linear regression makes all of the same assumptions as simple linear regression:
- Homogeneity of variance (homoscedasticity): the size of the error in our prediction doesn’t change significantly across the values of the independent variable.
- Independence of observations: the observations in the dataset were collected using statistically valid sampling methods, and there are no hidden relationships among variables.
- In multiple linear regression, it is possible that some of the independent variables are actually correlated with one another, so it is important to check these before developing the regression model. If two independent variables are too highly correlated (r2 > ~0.6), then only one of them should be used in the regression model.
- Normality: The data follows a normal distribution.
- Linearity: the line of best fit through the data points is a straight line, rather than a curve or some sort of grouping factor.
- Receive feedback

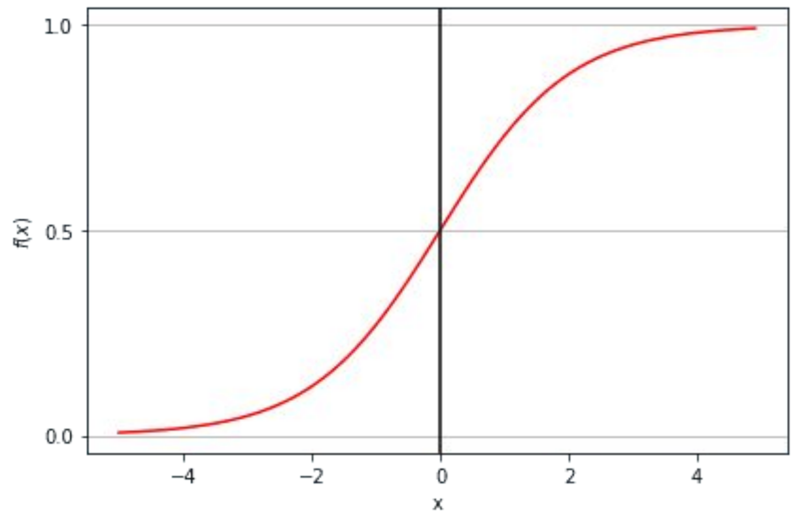
Logistic Regression

- Logistic regression is used for binary classification where we use sigmoid function, that takes input as independent variables and produces a probability value between 0 and 1.
- For example, we have two classes Class 0 and Class 1 if the value of the logistic function for an input is greater than 0.5 (threshold value) then it belongs to Class 1 otherwise it belongs to Class 0. It’s referred to as regression because it is the extension of linear regression but is mainly used for classification problems

Logistic Function – Sigmoid Function

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1. The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form.
- The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

$$f(x) = \frac{1}{1 + e^{-x}}$$



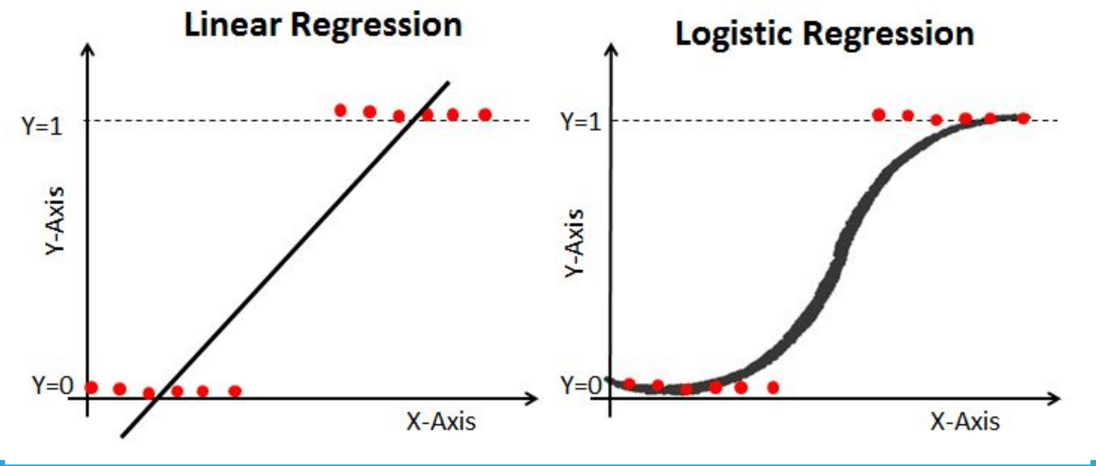
Types of Logistic Regression

On the basis of the categories, Logistic Regression can be classified into three types:

- Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as “cat”, “dogs”, or “sheep”
- Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as “low”, “Medium”, or “High”.

Linear Regression Vs. Logistic Regression

- Linear regression gives you a continuous output, but logistic regression provides a constant output.
- An example of the continuous output is house price and stock price.
- Examples of the discrete output are predicting whether a patient has cancer or not and predicting whether a customer will churn.
- Logistic regression is estimated using the maximum likelihood estimation (MLE) approach, while linear regression is typically estimated using ordinary least squares (OLS), which can also be considered a special case of MLE when the errors in the model are normally distributed.



Classification

- Classification is a supervised machine learning process of categorizing a given set of input data into classes based on one or more variables. Additionally, a classification problem can be performed on structured and unstructured data to accurately predict whether or not the data will fall into predetermined categories.
- Classification in machine learning can require two or more categories of a given data set. Therefore, it generates a probability score to assign the data into a specific category, such as spam or not spam, yes or no, disease or no disease, red or green, male or female, etc.

Applications of Machine Learning Classification Problems

- Image classification
- Fraud detection
- Document classification
- Spam filtering
- Facial recognition
- Voice recognition
- Medical diagnostic test
- Customer behaviour prediction
- Product categorization
- Malware classification

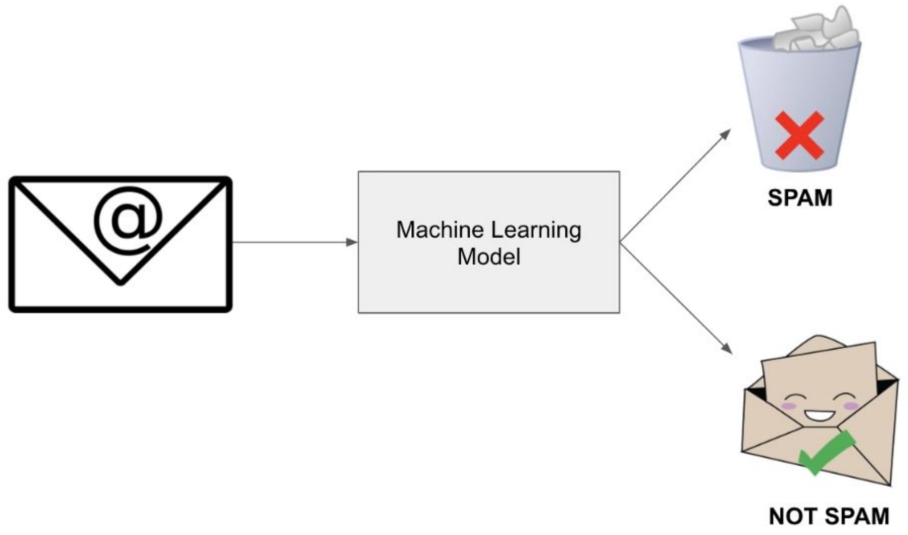
Types of Classification Tasks in ML

Classification Predictive Modeling

- In machine learning, classification is a predictive modeling problem where the class label is anticipated for a specific example of input data. For example, in determining handwriting characters, identifying spam, and so on, the classification requires training data with a large number of datasets of input and output. The most common classification algorithms are binary classification, multi-class classification, multi-label classification, and imbalanced classification, which are described further.

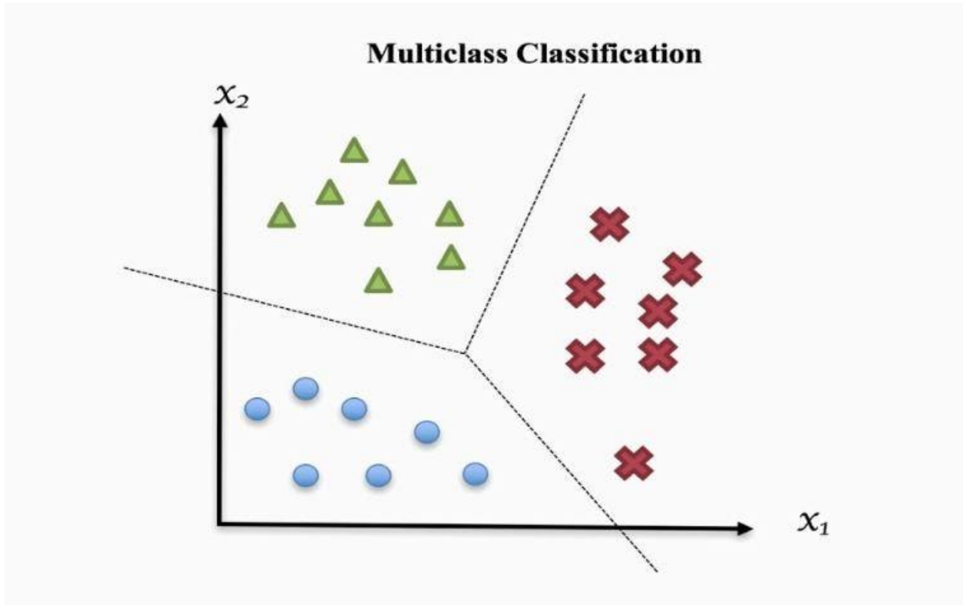
Binary Classification

- Binary is a type of problem in classification in machine learning that has only two possible outcomes. For example, yes or no, true or false, spam or not spam, etc. Some common binary classification algorithms are logistic regression, decision trees, simple bayes, and support vector machines.

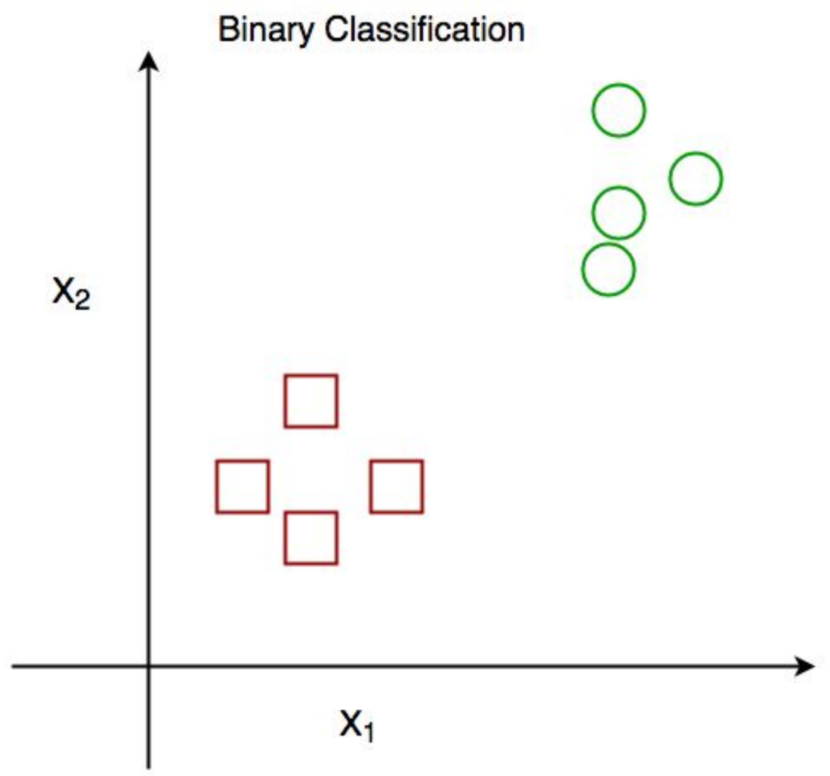


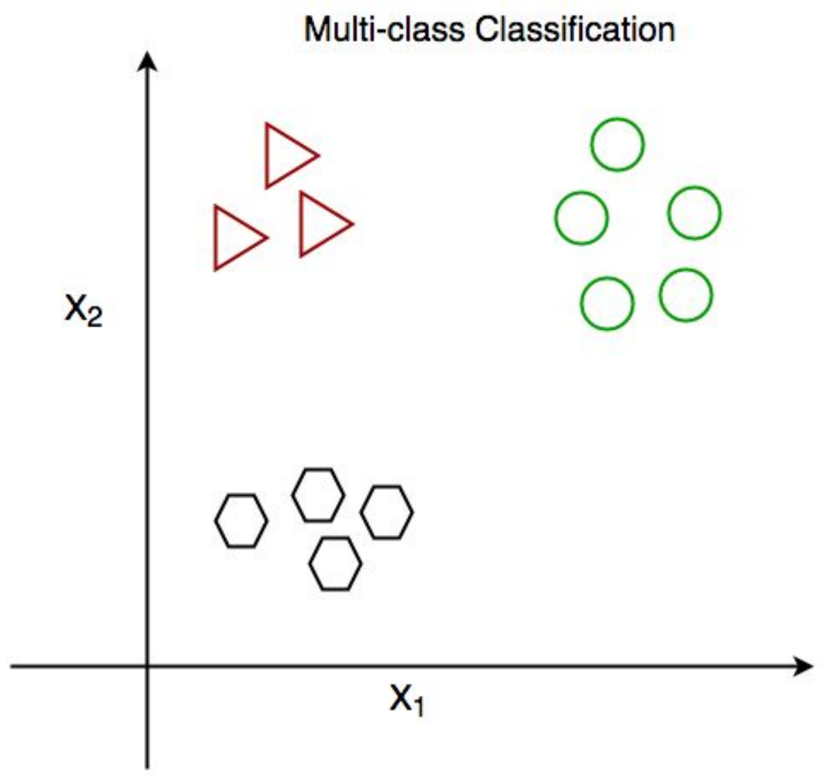
Multi-Class Classification

- Multi-class is a type of classification problem with more than two outcomes and does not have the concept of normal and abnormal outcomes. Here each outcome is assigned to only one label. For example, classifying images, classifying species, and categorizing faces, among others. Some common multi-class algorithms are choice trees, progressive boosting, nearest k neighbors, and rough forest.



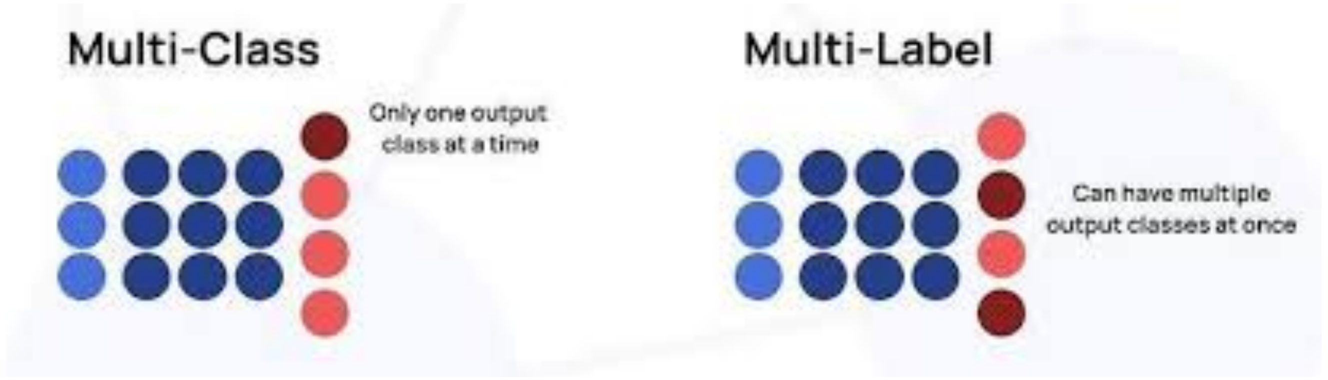
Binary Classification v/s Multi-Class Classification



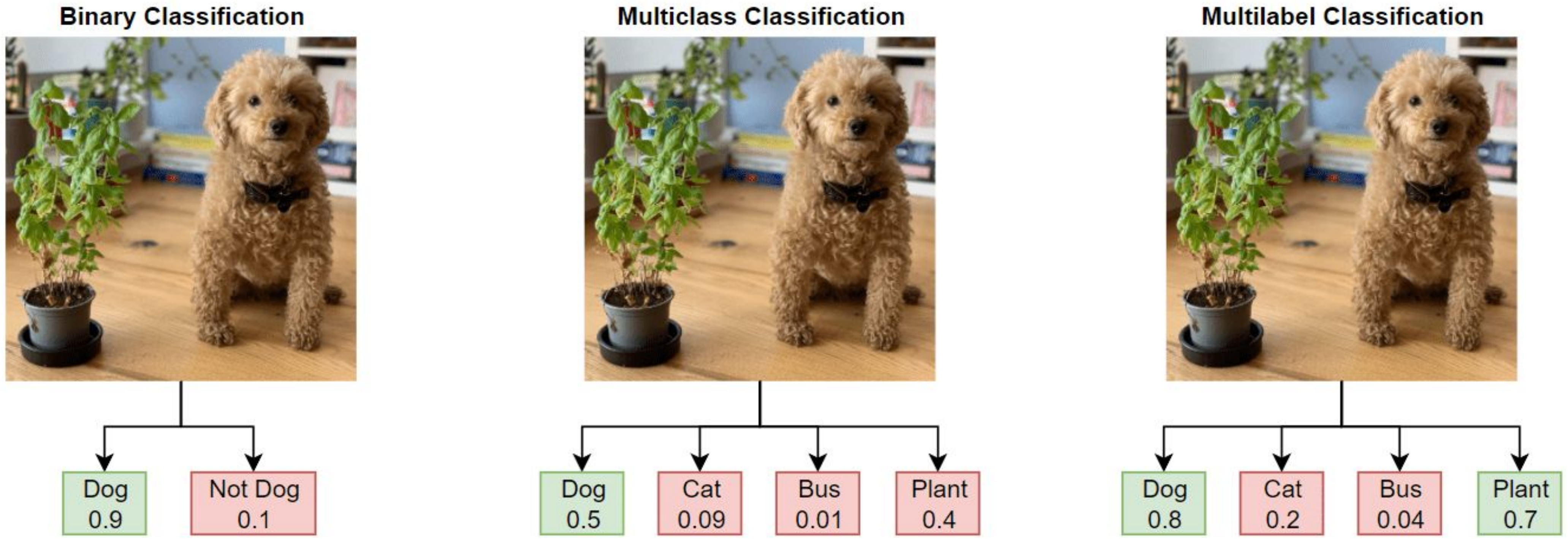


Multi-Label Classification

- Multi-label is a type of classification problem that may have more than one class label assigned to the data. Here the model will have multiple outcomes. For example, a book or a movie can be categorized into multiple genres, or an image can have multiple objects. Some common multi-label algorithms are multi-label decision trees, multi-label gradient boosting, and multi-label random forests.

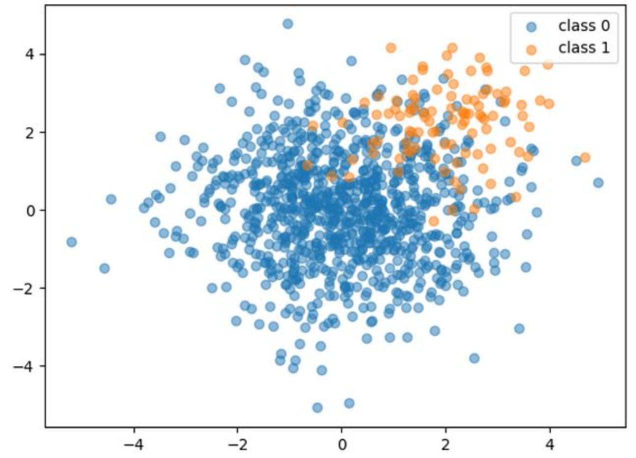

Multi-Class
Only one output class at a time
Multi-Label
Can have multiple output classes at once

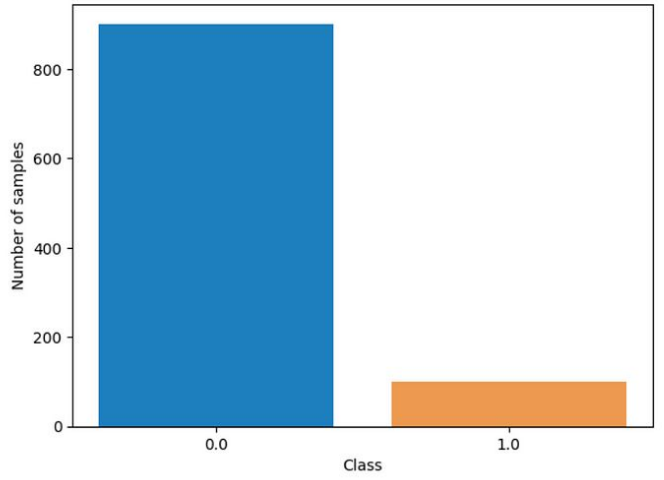
Binary v/s Multiclass v/s Multilabel Classification



Imbalanced Classification

- Most machine learning algorithms assume equal data distribution. When the data distribution is not equal, it leads to imbalance. An imbalanced classification problem is a classification problem where the distribution of the dataset is skewed or biased. This method employs specialized techniques to change the composition of data samples. Some examples of imbalanced classification are spam filtering, disease screening, and fraud detection.





Classification Problem

- A classification algorithm is a supervised learning technique that uses data training to determine data into different classes. Classification predictive modeling is trained using data or observations, and new observations are categorized into classes or groups. Classification predictive modeling is the task of a mapping function (f) from input variables (x) to discrete output variables (y). In this approach, the algorithm generates a probability score and assigns this score to the input. For example, email service providers use classification to generate probability scores for email identification to determine if the email is in the spam class or not.

Learners in Classification Problem

Lazy Learners

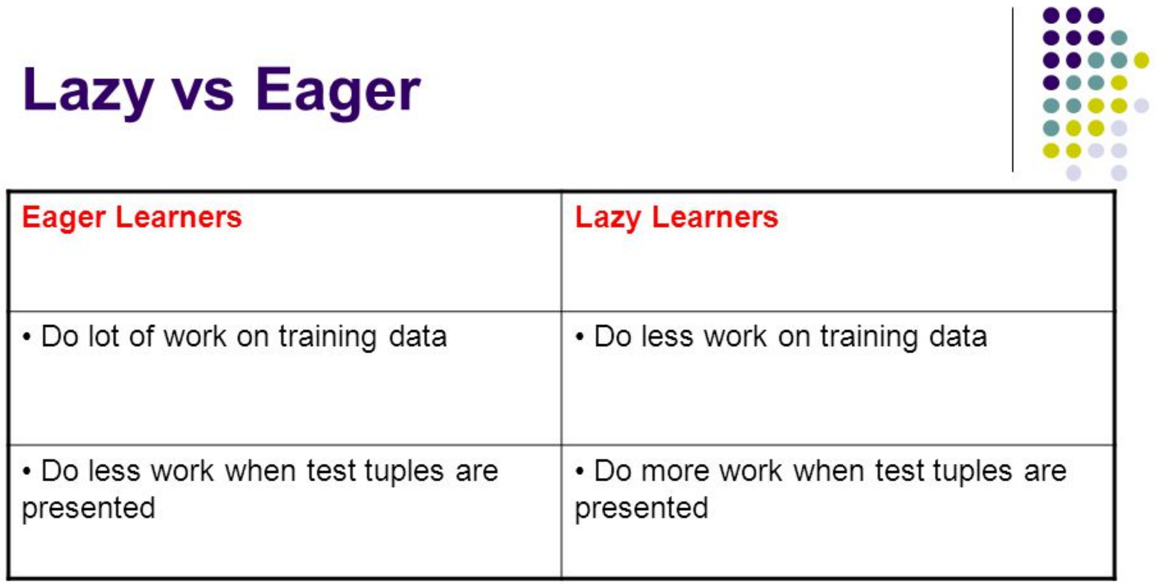
- Lazy learners store the training data and wait until a testing dataset appears. The primary aim of lazy learning is to continuously update the dataset with new entries. However, as the data is continually updated, it becomes outdated frequently. Thus, these algorithms take comparatively less time to train and more time to predict. Lazy learning algorithms are beneficial when working with large, changing datasets with a smaller set of queried attributes. Lazy learning is easy to maintain and can be applied to multiple problems. Some examples of lazy learners include local regression, lazy Bayesian rules and k-nearest neighbor (KNN) algorithm, instance-based learning, and case-based reasoning.

Learners in Classification Problem

Eager Learners

Eager learners construct a classification layer before receiving the training and testing the dataset. Before it observes the input queries, eager learning builds an explicit description of the training function based on the training data. Because it is building a classification model, eager learning takes more time to train the dataset and less time to predict as compared to the lazy learning system. Eager learning is required to commit to a single hypothesis that covers the entire instance space. Some examples of eager learners include decision trees, naive Bayes, and artificial neural networks (ANN).

Lazy Learners v/s Eager Learners



Lazy vs Eager

Eager Learners	Lazy Learners
• Do lot of work on training data	• Do less work on training data
• Do less work when test tuples are presented	• Do more work when test tuples are presented

K Nearest Neighbours(KNN)

- K-Nearest Neighbors is a simple, intuitive algorithm used for both classification and regression tasks in supervised learning. It works by finding the "k" nearest data points (neighbors) to a given data point and making predictions based on those neighbors.

Functionality of KNN

1. **Choose the Number of Neighbors (k):** You start by selecting the number of neighbors, k, that the algorithm will consider when making a prediction.
2. **Distance Metric:** For each data point in your dataset, KNN calculates the distance between the point you want to classify (or predict) and all other points in the dataset. Common distance metrics include Euclidean distance, Manhattan distance, and Minkowski distance.
3. **Find Nearest Neighbors:** Identify the k data points that are closest to the target point based on the chosen distance metric.
4. **Make a Prediction:**
    1. **For Classification:** KNN assigns the class label that is the most common among the k nearest neighbors. For example, if k=5 and 3 out of the 5 neighbors belong to Class A, the target point will be classified as Class A.
    2. **For Regression:** KNN predicts the target value by averaging the values of the k nearest neighbors.

Advantages of KNN

- **Simplicity:** It is easy to understand and implement.
- **No Training Phase:** KNN doesn’t require a training phase; it works directly with the data.

Disadvantages of KNN

- **Computational Complexity:** The algorithm can be slow, especially with large datasets, because it needs to calculate distances to all data points.
- **Memory Usage:** It requires storing the entire dataset, which can be problematic with large volumes of data.
- **Sensitivity to Irrelevant Features:** KNN can be affected by irrelevant or redundant features, making feature selection important.
- **Choice of k:** The choice of *k* can greatly influence the performance of the algorithm. A small k can make the model sensitive to noise, while a large *k*k can smooth out distinctions between classes or values.

How to choose the right value of k?

- The value of k is crucial and can be chosen using techniques like:
- **Cross-Validation:** Testing different values of k to see which one gives the best performance on a validation set.
- **Domain Knowledge:** Sometimes, domain-specific considerations can guide the choice of k.

Scaling and Normalization

- Since KNN relies on distance calculations, it's essential to normalize or standardize your data so that features contribute equally to the distance metrics. Features on different scales can disproportionately affect the results.

Working of KNN

- Step-1: Select the number K to decide the number of clusters.
- Step-2: Select random K points or centroids. (It can be other from the input dataset).
- Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.
- Step-4: Calculate the variance and place a new centroid of each cluster.
- Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.
- Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.
- Step-7: The model is ready.

Naïve Bayes

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Bayes' Theorem

Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
The formula for Bayes' theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Where,
P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
P(B) is Marginal Probability: Probability of Evidence.

Working of Naïve Bayes

- Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

  - Convert the given dataset into frequency tables.
  - Generate Likelihood table by finding the probabilities of given features.
  - Now, use Bayes’ theorem to calculate the posterior probability.

- **Problem:** If the weather is sunny, then shall the Player should play or not?

Naïve Bayes Example

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Naïve Bayes Example

- Frequency table for the Weather Conditions:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

Naïve Bayes Example

- Likelihood table weather condition:

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

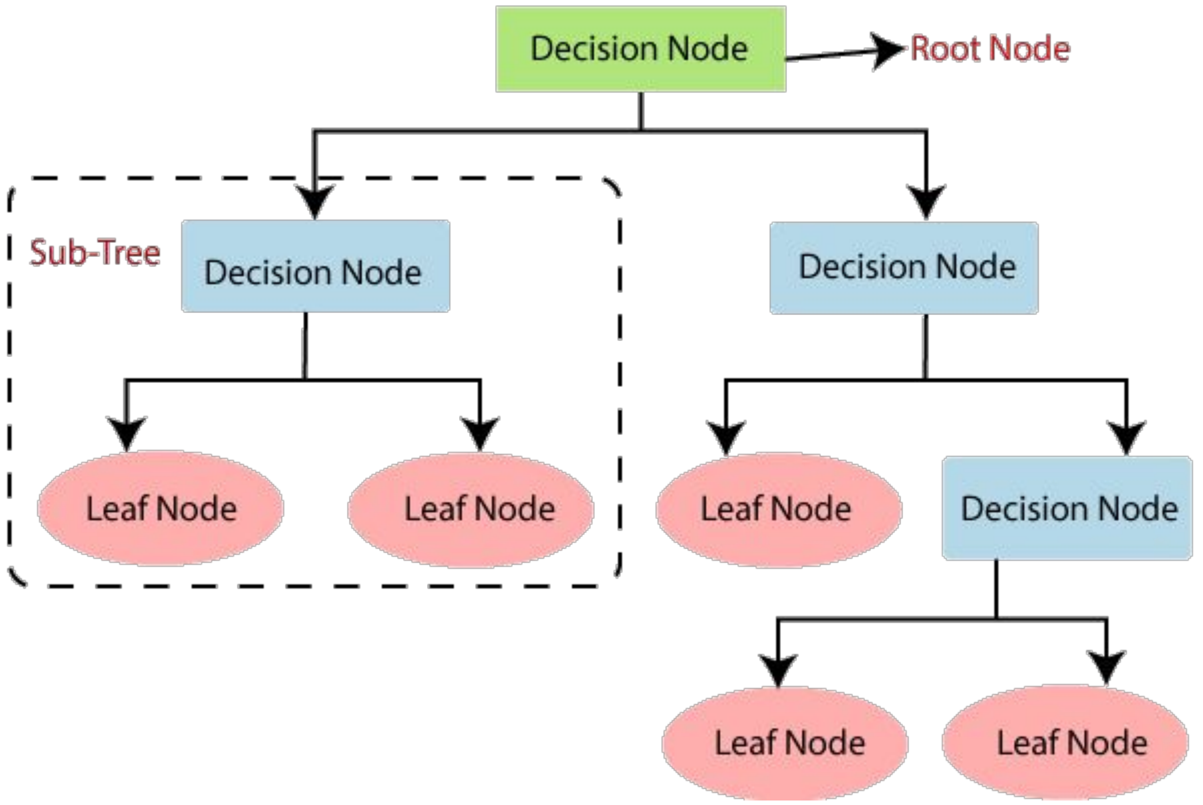
Naïve Bayes Example

- **Applying Bayes' theorem:**
- P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)
- P(Sunny|Yes)= 3/10= 0.3
- P(Sunny)= 0.35
- P(Yes)=0.71
- **So P(Yes|Sunny) = 0.3*0.71/0.35= 0.60**
- P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)
- P(Sunny|NO)= 2/4=0.5
- P(No)= 0.29
- P(Sunny)= 0.35
- **So P(No|Sunny)= 0.5*0.29/0.35 = 0.41**
- So as we can see from the above calculation that P(Yes|Sunny)>P(No|Sunny)
- Hence on a Sunny day, Player **can** play the game.

Decision Trees

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Decision Trees



Decision Tree Terminologies

- **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Pruning:** Pruning is the process of removing the unwanted branches from the tree.
- **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

Decision Tree Terminologies-Entropy

- Entropy is a measure of disorder or impurity in the given dataset.
- In the decision tree, messy data is split based on values of the feature vector associated with each data point. With each split, the data becomes more homogenous which will decrease the entropy. However, some data in some nodes will not be homogenous, where the entropy value will not be small. The higher the entropy, the harder it is to draw any conclusion. When the tree finally reaches the terminal or leaf node maximum purity is added.
- For a dataset that has C classes and the probability of randomly choosing data from class, i is Pi. Then entropy E(S) can be mathematically represented as

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

Decision Tree Terminologies-Information Gain

- The Information Gain measures the expected reduction in entropy. Entropy measures impurity in the data and information gain measures reduction in impurity in the data. The feature which has minimum impurity will be considered as the root node.
- Information gain is used to decide which feature to split on at each step in building the tree. The creation of sub-nodes increases the homogeneity, that is decreases the entropy of these nodes. The more the child node is homogeneous, the more the variance will be decreased after each split. Thus Information Gain is the variance reduction and can calculate by how much the variance decreases after each split.
- Information gain of a parent node can be calculated as the entropy of the parent node subtracted entropy of the weighted average of the child node.

Decision Tree Terminologies-Information Gain-Gini Index

- The Gini index can also be used for feature selection. The tree chooses the feature that minimizes the Gini impurity index. The higher value of the Gini Index indicates the impurity is higher. Both Gini Index and Gini Impurity are used interchangeably. The Gini Index or Gini Impurity favors large partitions and is very simple to implement. It performs only binary split. For categorical variables, it gives the results in terms of "success" or "failure".
- Gini Index can be calculated from the below mathematical formula where c is the number of classes and pi is the probability associated with the ith class.
- The Gini Index or Gini Impurity favors large partitions and is very simple to implement. It performs only binary split. For categorical variables, it gives the results in terms of "success" or "failure".

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

Decision Trees Terminologies-Pruning

- When the tree is fully grown up, it is liking to overfit data due to noise or outliers which can lead to anomalies in decision trees. Which in turn leads to poor accuracy. This can be handled by using pruning.
- Pruning is the process of removing redundant comparisons or removing subtrees. Pruning reduces unnecessary comparisons and achieves better performance. Pruned trees are less complex, smaller, and easy to understand. There are two approaches for pruning, the pre-pruning approach in which splitting or partition of the tree is halted at a particular node whereas in post-pruning approach removes subtree from the full tree. A subtree is pruned at a node. It is done by removing the branches at a node and replacing it with a leaf node.

Random Forests

- Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."
- Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.
- The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

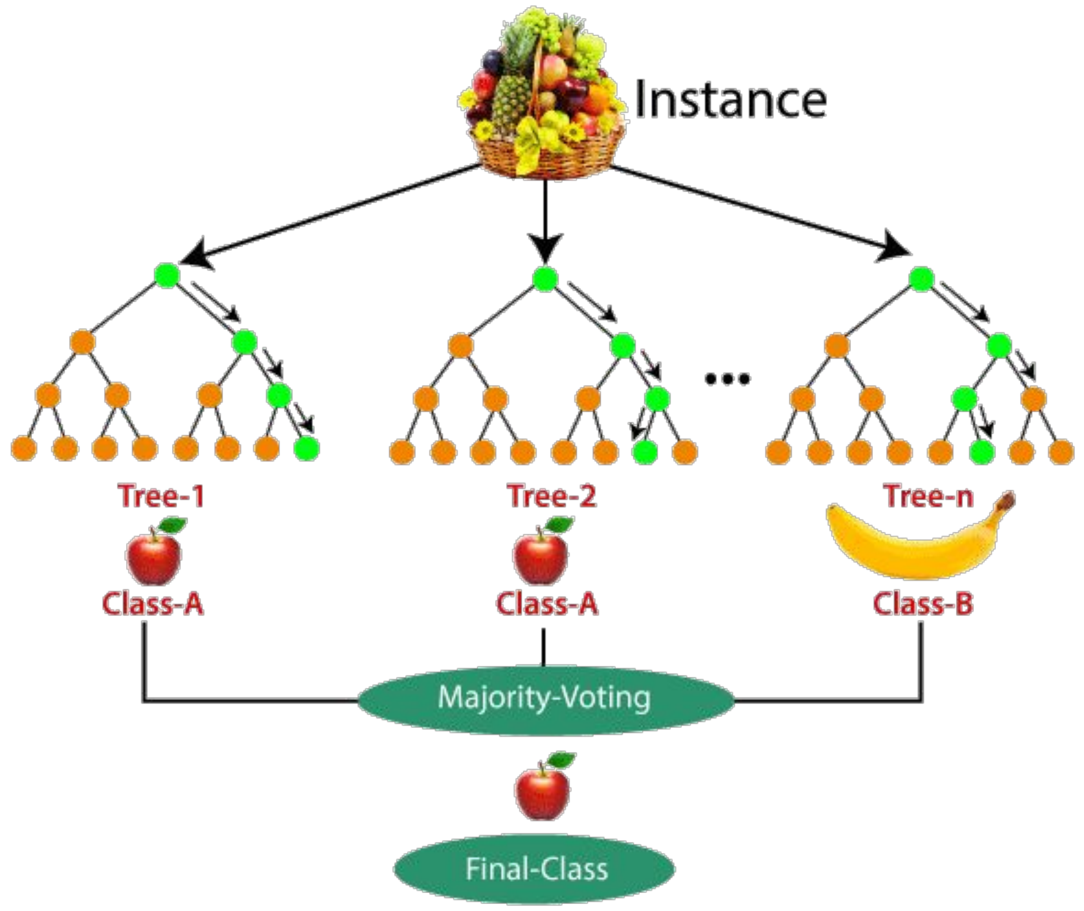
How does Random Forests work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

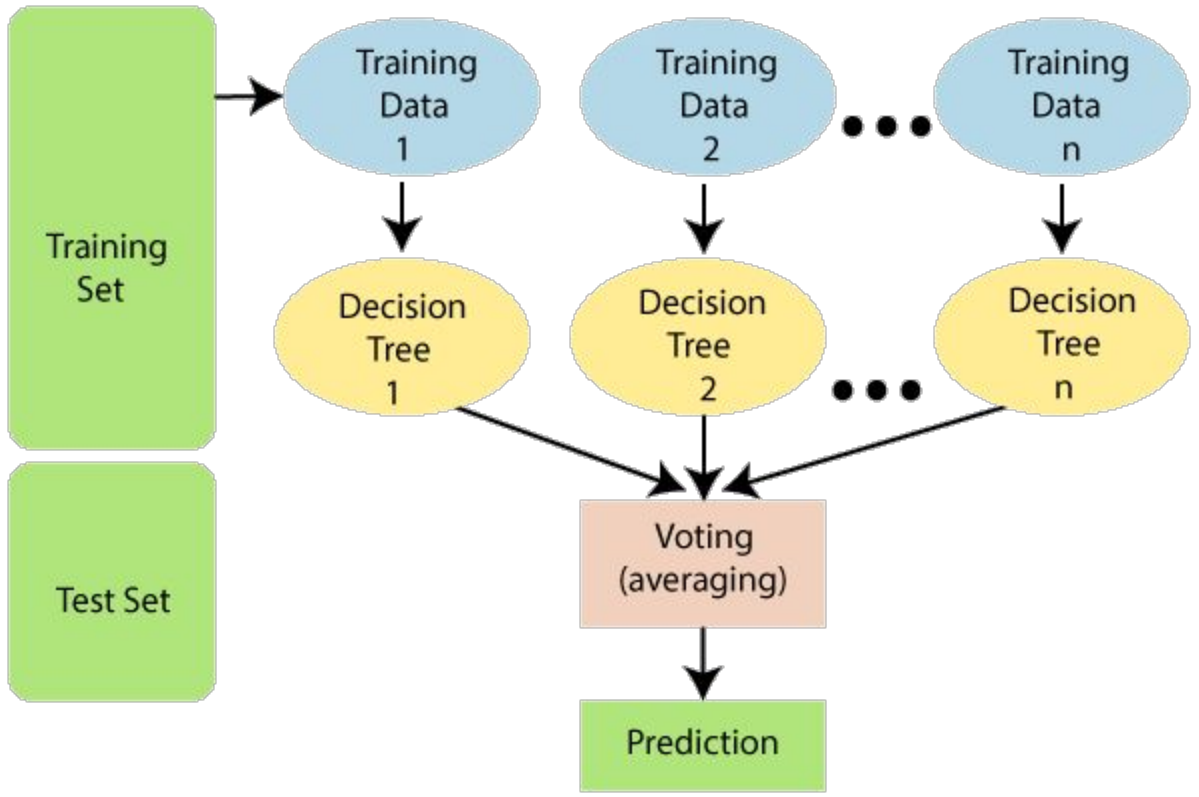
The Working process can be explained in the below steps and diagram:

- Step-1: Select random K data points from the training set.
- Step-2: Build the decision trees associated with the selected data points (Subsets).
- Step-3: Choose the number N for decision trees that you want to build.
- Step-4: Repeat Step 1 & 2.
- Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Random Forests



Random Forests

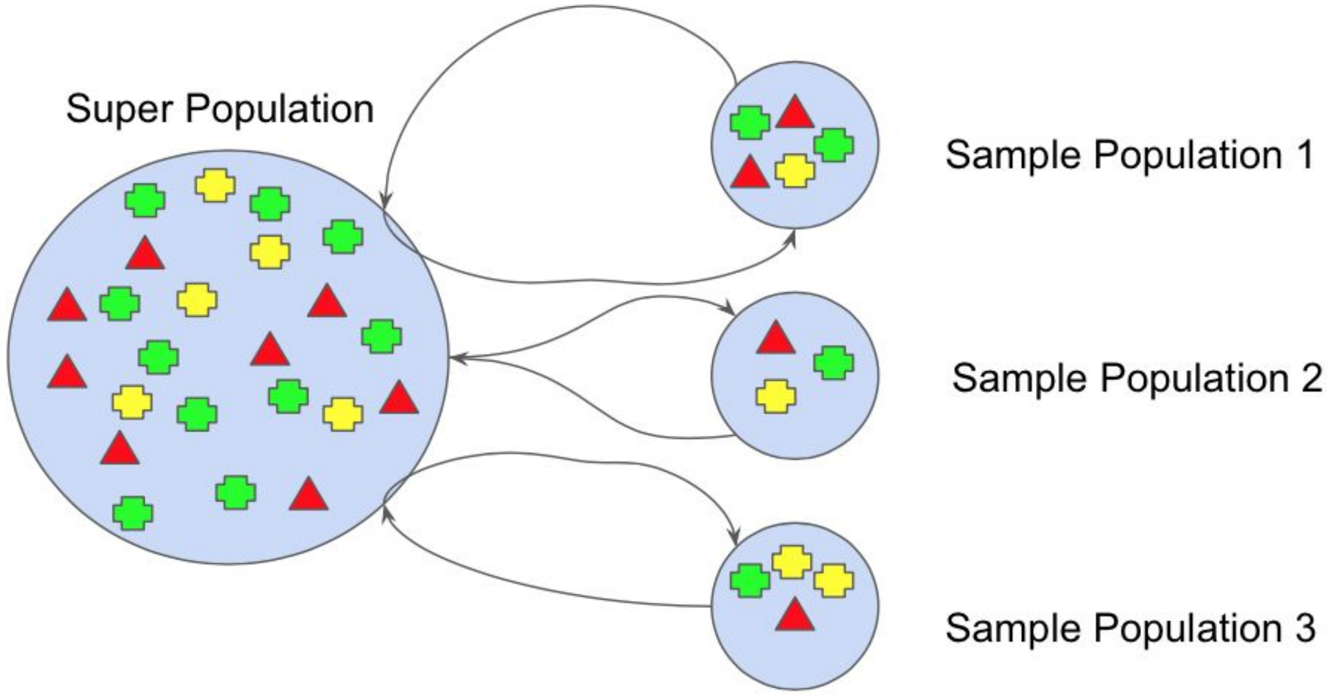


Bootstrapping in Random Forests

- Bootstrap refers to random sampling with replacement. Bootstrap allows us to better understand the bias and the variance with the dataset.
- So, Bootstrapping is a sampling technique in which we create subsets of observations from the original dataset with replacement. The size of the subsets is the same as the size of the original set.
- Bootstrap involves random sampling of small subset of data from the dataset. This subset can be replaced.
- The selection of all the example in the dataset has equal probability. This method can help to better understand the mean and standand deviation from the dataset.
- Let's assume we have a sample of 'n' values (x) and we want an estimate of the mean of the sample. We can calculate it as follows:

  mean(x) = 1/n * sum(x)
- Bootstrapping can be represented diagrammatically as follows:

Bootstrapping in Random Forests



Bagging in Random Forests

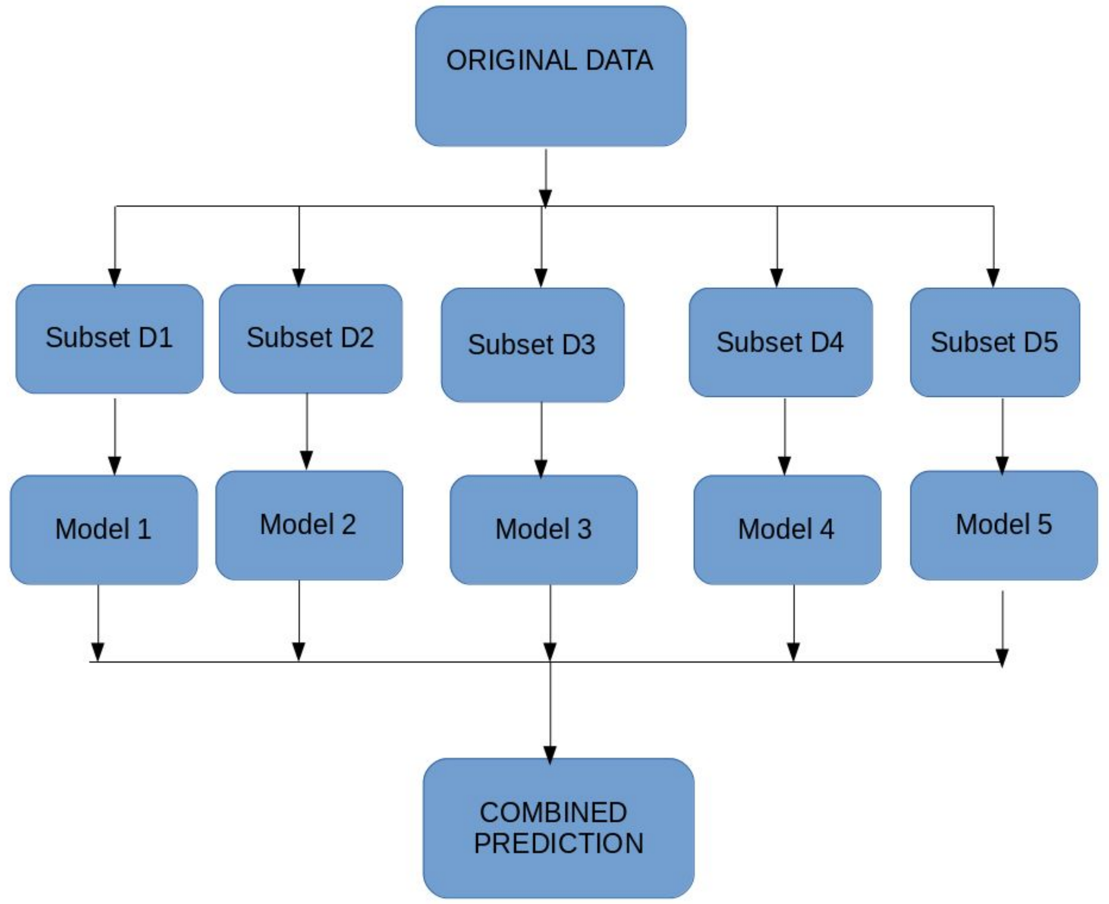
- Bagging (or Bootstrap Aggregation), is a simple and very powerful ensemble method. Bagging is the application of the Bootstrap procedure to a high-variance machine learning algorithm, typically decision trees.
- The idea behind bagging is combining the results of multiple models (for instance, all decision trees) to get a generalized result. Now, bootstrapping comes into picture.
- Bagging (or Bootstrap Aggregating) technique uses these subsets (bags) to get a fair idea of the distribution (complete set). The size of subsets created for bagging may be less than the original set.
- It can be represented as follows:

Bagging in Random Forests

Bagging works as follows:-

- Multiple subsets are created from the original dataset, selecting observations with replacement.
- A base model (weak model) is created on each of these subsets.
- The models run in parallel and are independent of each other.
- The final predictions are determined by combining the predictions from all the model

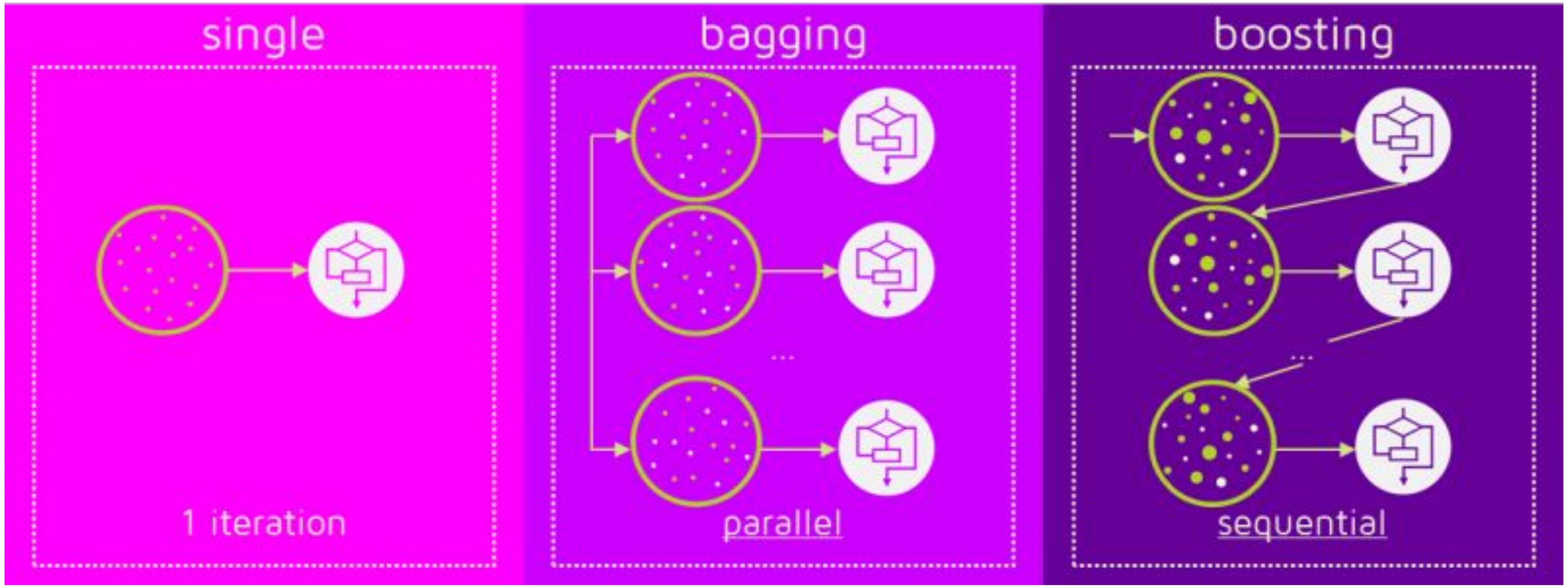
Bagging in Random Forests



Boosting in Random Forests

- Boosting is a sequential process, where each subsequent model attempts to correct the errors of the previous model. The succeeding models are dependent on the previous model.
- In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analyzing data for errors. In other words, we fit consecutive trees (random sample) and at every step, the goal is to solve for net error from the prior tree.
- When an input is misclassified by a hypothesis, its weight is increased so that next hypothesis is more likely to classify it correctly. By combining the whole set at the end converts weak learners into better performing model.
- Let’s understand the way boosting works in the below steps.
  - A subset is created from the original dataset.
  - Initially, all data points are given equal weights.
  - A base model is created on this subset.
  - This model is used to make predictions on the whole dataset.

Difference between Bagging & Boosting



Difference between Bagging & Boosting

- Bagging is the simplest way of combining predictions that belong to the same type while Boosting is a way of combining predictions that belong to the different types.
- Bagging aims to decrease variance, not bias while Boosting aims to decrease bias, not variance.
- In Bagging, each model receives equal weight whereas in Boosting models are weighted according to their performance.
- In Bagging,each model is built independently whereas in Boosting new models are influenced by performance of previously built models.
- In Bagging different training data subsets are randomly drawn with replacement from the entire training dataset. In Boosting every new subsets contains the elements that were misclassified by previous models.
- Bagging tries to solve over-fitting problem while Boosting tries to reduce bias.
- If the classifier is unstable (high variance), then we should apply Bagging. If the classifier is stable and simple (high bias) then we should apply Boosting.
- Bagging is extended to Random forest model while Boosting is extended to Gradient boosting.

Support Vector Machine

- A support vector machine (SVM) is a type of supervised learning algorithm used in machine learning to solve classification and regression tasks
- SVMs are particularly good at solving binary classification problems, which require classifying the elements of a data set into two groups.
- The aim of a support vector machine algorithm is to find the best possible line, or decision boundary, that separates the data points of different data classes. This boundary is called a hyperplane when working in high-dimensional feature spaces.
- The idea is to maximize the margin, which is the distance between the hyperplane and the closest data points of each category, thus making it easy to distinguish data classes.
- SVMs are useful for analyzing complex data that can't be separated by a simple straight line. Called nonlinear SMVs, they do this by using a mathematical trick that transforms data into higher-dimensional space, where it is easier to find a boundary.

Support Vector Machine

- The key idea behind SVMs is to transform the input data into a higher-dimensional feature space. This transformation makes it easier to find a linear separation or to more effectively classify the data set.
- To do this, SVMs use a kernel function. Instead of explicitly calculating the coordinates of the transformed space, the kernel function enables the SVM to implicitly compute the dot products between the transformed feature vectors and avoid handling expensive, unnecessary computations for extreme cases.

SVM Kernel

- SVMs can handle both linearly separable and non-linearly separable data. They do this by using different types of kernel functions, such as the linear kernel, polynomial kernel or radial basis function (RBF) kernel. These kernels enable SVMs to effectively capture complex relationships and patterns in the data.
- During the training phase, SVMs use a mathematical formulation to find the optimal hyperplane in a higher-dimensional space, often called the kernel space. This hyperplane is crucial because it maximizes the margin between data points of different classes, while minimizing the classification errors.

SVM Kernel

- The kernel function plays a critical role in SVMs, as it makes it possible to map the data from the original feature space to the kernel space. The choice of kernel function can have a significant impact on the performance of the SVM algorithm; choosing the best kernel function for a particular problem depends on the characteristics of the data.
- Some of the most popular kernel functions for SVMs are the following:
  - **Linear kernel:** This is the simplest kernel function, and it maps the data to a higher-dimensional space, where the data is linearly separable.
  - **Polynomial kernel:** This kernel function is more powerful than the linear kernel, and it can be used to map the data to a higher-dimensional space, where the data is non-linearly separable.
  - **RBF kernel:** This is the most popular kernel function for SVMs, and it is effective for a wide range of classification problems.
  - **Sigmoid kernel:** This kernel function is similar to the RBF kernel, but it has a different shape that can be useful for some classification problems.

Types of SVM

- Support vector machines have different types and variants that provide specific functionalities and address specific problem scenarios. Here are two types of SVMs and their significance:
- **Linear SVM:** Linear SVMs use a linear kernel to create a straight-line decision boundary that separates different classes. They are effective when the data is linearly separable or when a linear approximation is sufficient. Linear SVMs are computationally efficient and have good interpretability, as the decision boundary is a hyperplane in the input feature space.
- **Nonlinear SVM:** Nonlinear SVMs address scenarios where the data cannot be separated by a straight line in the input feature space. They achieve this by using kernel functions that implicitly map the data into a higher-dimensional feature space, where a linear decision boundary can be found. Popular kernel functions used in this type of SVM include the polynomial kernel, Gaussian (RBF) kernel and sigmoid kernel. Nonlinear SVMs can capture complex patterns and achieve higher classification accuracy when compared to linear SVMs.

SVM-Concepts

- **Decision boundary:** A decision boundary is an imaginary line or boundary that separates different groups or categories in a data set, placing data sets into different regions. For instance, an email decision boundary might classify an email with over 10 exclamation marks as "spam" and an email with under 10 marks as "not spam."
- **Hyperplane:** In n-dimensional space -- that is, a space with many dimensions -- a hyperplane is defined as an (n-1)-dimensional subspace, a flat surface that has one less dimension than the space itself. In a two-dimensional space, its hyperplane would be one-dimensional or a line.
- **Kernel function:** A kernel function is a mathematical function used in the kernel trick to compute the inner product between two data points in the transformed feature space. Common kernel functions include linear, polynomial, Gaussian (RBF) and sigmoid.
- **Regularization:** It is a technique used to prevent overfitting in SVMs. Regularization introduces a penalty term in the objective function, encouraging the algorithm to find a simpler decision boundary rather than fitting the training data perfectly.

SVM-Concepts

- In the context of Support Vector Machines (SVM) the parameter γ (gamma) can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.
- Intuitively, a low gamma value means that the influence of a single training example reaches far, affecting a larger region of the feature space.
- Conversely, a high gamma value means that the influence is close, affecting only the region near the training example
- Impact of Gamma on Model Complexity
- **When γ value is high:** When γ is large, the influence of each training example is limited to a small region. This means that the decision boundary will be very tight around individual data points. This can lead to overfitting, where the model captures noise in the training data, resulting in poor generalization to new, unseen data.
- **When γ value is low:** When γ is small, the influence of each training example reaches farther, resulting in smoother and less complex decision boundaries. This can lead to underfitting, where the model is too simple and fails to capture the underlying patterns in the data.