UNIT 4

TOPICS:

Introduction to Regression: Linear Regression, Polynomial regression. Metric for regression –mean square error.

Introduction to classification : decision tree, threshold for classification , metric for classification-accuracy, F1 score , confusion matrix. Type I and type 2 errors. Bias and variance , overfitting and under fitting in supervised algorithm

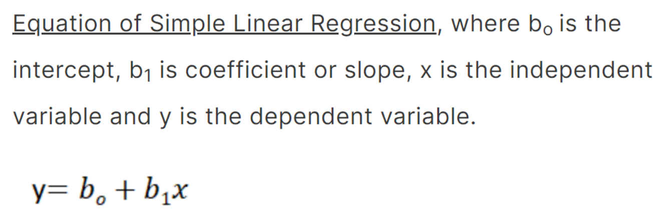
QUESTIONS:

1. Steps to solve id3 and sum
2. linear regression sum
3. What is confusion matrix? Give example and explain
4. Describe Type 1 and type 2 errors
5. Write a short note on types of regression polynomial/ logical? Give advantages disadvantages and application areas
6. simple python code data in list x and y just 5 - 6 line ka code in pracs
7. Differentiate between regression types
8. Differentiate between type 1 and type 2
9. Differentiate between supervised and unsupervised
10. Explain bias variance
11. Explainunder and overfitting
12. Explain Binary classification
13. Explain lazy learning
14. Explain eager learning

REGRESSION

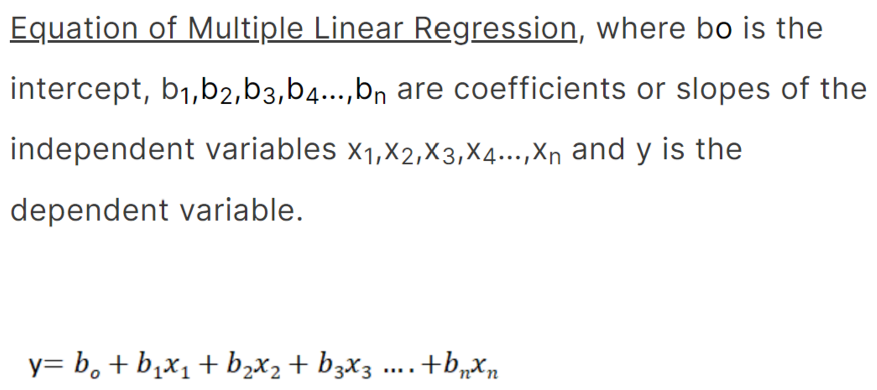
- regression is a powerful technique used in data science to understand the relationships between variables
- way to uncover how one thing (independent variable) affects another (dependent variable).
- done by fitting a model, typically a line or curve, to the data points.
- GOAL
    - aims to estimate the relationship between a dependent variable (what you're trying to predict) and one or more independent variables (what you're basing the prediction on).
    - widely used for tasks like forecasting future values, understanding trends, and even identifying causal relationships between variables.
    - different regression models, with linear regression being the simplest.
        - Linear regression fits a straight line to the data, while other models like polynomial regression can capture more complex relationships.

LINEAR REGRESSION

- Supervised learning model where
    - model finds the best fit linear line between the independent and dependent variable
    - i.e it finds the linear relationship between the dependent and independent variable.
- Two types:
    - Simple
        - only one independent variable is present and the model has to find the linear relationship of it with the dependent variable
    - Multiple
        - more than one independent variables for the model to find the relationship.
- Eqn:
    - 

      Equation of Simple Linear Regression, where b_o is the intercept, b_1 is coefficient or slope, x is the independent variable and y is the dependent variable.

      $$y = b_o + b_1 x$$



Equation of Multiple Linear Regression, where bo is the intercept, $b_1, b_2, b_3, b_4 \ldots, b_n$ are coefficients or slopes of the independent variables $x_1, x_2, x_3, x_4 \ldots, x_n$ and y is the dependent variable.

$$y = b_o + b_1x_1 + b_2x_2 + b_3x_3 \ldots . + b_nx_n$$

- A Linear Regression model's main aim is to find the best fit linear line and the optimal values of intercept and coefficients such that the error is minimized.
- Error => difference between the actual value and predicted value
  - the goal is to reduce this difference.
- Assumptions of linear regression
  - Linearity
    - It states that the dependent variable Y should be linearly related to independent variables.
  - Normality
    - The X and Y variables should be normally distributed.
  - Homoscedasticity
    - The variance of the error terms should be constant i.e the spread of residuals should be constant for all values of X.
  - Independence/No Multicollinearity:
    - The variables should be independent of each other i.e no correlation should be there between the independent variables
  - The error terms should be normally distributed
  - No Autocorrelation:
    - The error terms should be independent of each other. Autocorrelation can be tested using the Durbin Watson test.

L.R SUM

- Least Squares Regression Line, LSRL
- The calculation is based on the method of least squares.
- The idea behind it is to minimise the sum of the vertical distance between all of the data points and the line of best fit.

The equation of the least squares regression line is

$$\hat{y} = a + bx$$

where:

- $\hat{y}$ is the predicted value of y,
- $a = \bar{y} - b\bar{x}$,
- $b = \frac{S_{xy}}{S_{xx}} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2} = \frac{\sum(xy) - \frac{\sum x \sum y}{n}}{\sum(x^2) - \frac{(\sum x)^2}{n}}$,
- $\bar{x} = \frac{\sum x}{n}$,
- $\bar{y} = \frac{\sum y}{n}$,

EXAMPLE:

- Consider the example below where the mass, y (grams), of a chemical is related to the time, x (seconds), for which the chemical reaction has been taking place according to the table:

Time, x (seconds)	5	7	12	16	20
Mass, y (grams)	40	120	180	210	240

- find eqn

Start off by working out the mean of the independent and dependent variables.

$$\begin{aligned}\bar{x} &= \frac{\sum x}{n}\\ &= \frac{5+7+12+16+20}{5}\\ &= \frac{60}{5}\\ &= 12,\\ \bar{y} &= \frac{\sum y}{n}\\ &= \frac{40+120+180+210+240}{5}\\ &= \frac{790}{5}\\ &= 158.\end{aligned}$$

x_i	y_i	$x_i - \bar{x}$	$y_i - \bar{y}$	$(x_i - \bar{x})(y_i - \bar{y})$	$(x_i - \bar{x})^2$
5	40	$5 - 12 = -7$	$40 - 158 = -118$	$-7 \times -118 = 826$	$-7^2 = 49$
7	120	$7 - 12 = -5$	$120 - 158 = -38$	$-5 \times -38 = 190$	$-5^2 = 25$
12	180	$12 - 12 = 0$	$180 - 158 = 22$	$0 \times 22 = 0$	$0^2 = 0$
16	210	$16 - 12 = 4$	$210 - 158 = 52$	$4 \times 52 = 208$	$4^2 = 16$
20	240	$20 - 12 = 8$	$240 - 158 = 82$	$8 \times 82 = 656$	$8^2 = 64$
$\sum x = 60$	$\sum y = 790$			$\sum(x_i - \bar{x})(y_i - \bar{y}) = 1880$	$\sum(x_i - \bar{x})^2 = 154$

$$\begin{aligned} b &= \frac{S_{xy}}{S_{xx}} \\ &= \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2} \\ &= \frac{1880}{154} = 12.20779... \\ &= 12.208 \text{ (3.d.p.)} \end{aligned}$$

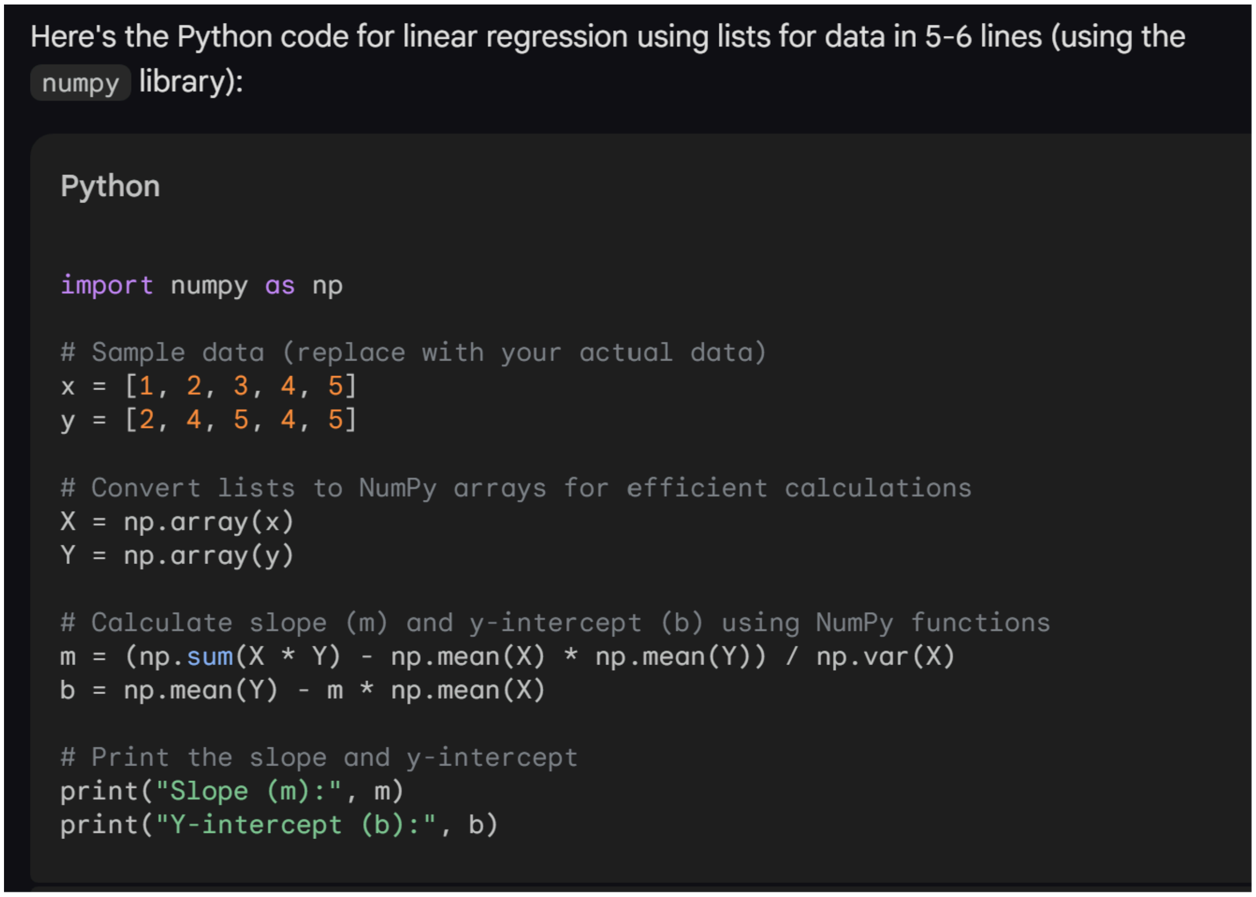
$$\begin{aligned} a &= \bar{y} - b\bar{x} \\ &= 158 - 12.208 \times 12 \\ &= 11.506... \\ &= 11.506 \text{ (3.d.p.).} \end{aligned}$$

-

POLYNOMIAL REGRESSION

- Polynomial Regression is a regression algorithm that models the relationship
  - between a dependent(y) and independent variable(x)
  - as nth degree polynomial
- Eqn:
  - $y= b_0+b_1x_1+ b_2x_1^2+ b_2x_1^3+...... b_nx_1^n$
- It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression.
- It is a linear model with some modification in order to increase the accuracy.
- The dataset used in Polynomial regression for training is of non-linear nature.
- It makes use of a linear regression model to fit the complicated and non-linear functions and datasets.
- Steps for Polynomial Regression:
- The main steps involved in Polynomial Regression are given below:
  - Data Pre-processing
  - Build a Linear Regression model and fit it to the dataset
  - Build a Polynomial Regression model and fit it to the dataset
  - Visualize the result for Linear Regression and Polynomial Regression model.
  - Predicting the output.

- Advantages:
    - Captures Non-Linear Relationships: Polynomial regression is beneficial when the data exhibits a curved or non-linear pattern that a straight line cannot capture effectively.
- Disadvantages:
    - Overfitting: Using high-degree polynomials can lead to overfitting, where the model memorizes the training data but fails to generalize well to unseen data.
    - Higher Variance: Complex polynomial models can have higher variance, meaning small changes in the data can lead to significant changes in the fitted curve.
    - Loss of Interpretability: As the degree of the polynomial increases, the model becomes more complex and the relationship between the variables becomes harder to interpret.



Here's the Python code for linear regression using lists for data in 5-6 lines (using the `numpy` library):

Python

```python
import numpy as np

# Sample data (replace with your actual data)
x = [1, 2, 3, 4, 5]
y = [2, 4, 5, 4, 5]

# Convert lists to NumPy arrays for efficient calculations
X = np.array(x)
Y = np.array(y)

# Calculate slope (m) and y-intercept (b) using NumPy functions
m = (np.sum(X * Y) - np.mean(X) * np.mean(Y)) / np.var(X)
b = np.mean(Y) - m * np.mean(X)

# Print the slope and y-intercept
print("Slope (m):", m)
print("Y-intercept (b):", b)
```

Linear Regression: The Simplest Storyteller

- Function: Models linear relationships between a dependent variable (y) and one or more independent variables (x). Imagine fitting a straight line to your data.
- Advantages:
  - Simplicity: Easy to understand and interpret.
  - Computational Efficiency: Fast to train and implement.
- Disadvantages:
  - Limited to Linear Relationships: Cannot capture complex, curved patterns.
- Applications:
  - Analyzing trends over time (e.g., stock prices vs. time).
  - Predicting continuous values based on a single variable (e.g., house price based on square footage).

Polynomial Regression: Embracing the Curves

- Function: Models non-linear relationships using polynomial equations. Think curvy lines instead of straight ones.
- Advantages:
  - Flexibility: Adapts to various curved patterns.
  - Interpretability (for lower degrees): Understandable relationships between variables (for simpler models).
- Disadvantages:
  - Overfitting: Prone to memorizing training data, hindering performance on new data (especially with high-degree polynomials).
  - Higher Variance: Sensitive to data changes, leading to unstable models with complex polynomials.
  - Interpretability Challenge (for higher degrees): Complex models become difficult to interpret.
- Applications:
  - Modeling growth trajectories (e.g., population growth over time).
  - Predicting trends with non-linear patterns (e.g., sales figures based on marketing spending).

Logistic Regression: Conquering the Binary World

- Function: Models the probability of a binary outcome (yes/no, pass/fail) based on one or more independent variables.
- Advantages:
  - Straightforward Interpretation: Estimates the probability of an event, making it easy to understand the impact of changes in variables.
  - Robustness: Handles outliers and non-linear relationships to some extent.
- Disadvantages:
  - Limited to Binary Outcomes: Not suitable for predicting continuous values.
  - Assumptions: Relies on assumptions about the data distribution, which may not always hold true.
- Applications:
  - Classifying emails as spam or not spam.
  - Predicting loan defaults (approve/deny).
  - Modeling customer churn (stay/leave).

Choosing Your Regression Champion

The best regression technique depends on your problem:

- For simple, linear relationships, linear regression reigns supreme.
- For non-linear continuous predictions, polynomial regression is your champion.
- But if you're dealing with binary outcomes, logistic regression is the hero you need.

REGRESSION METRIC

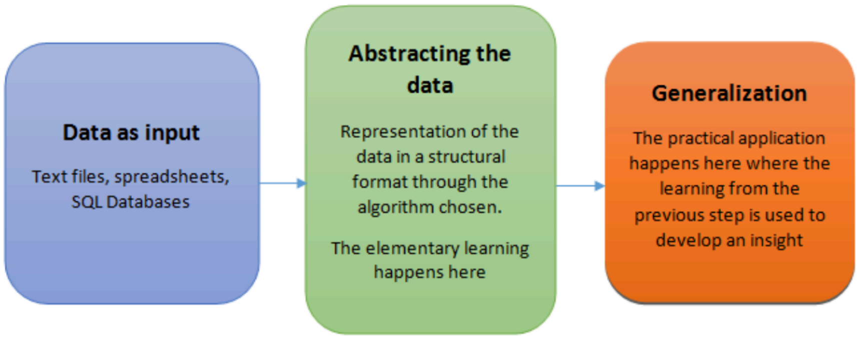
- Root Mean Squared Error (RMSE)
- RMSE stands for Root Mean Squared Error.
- metric in regression analysis and machine learning to measure the accuracy or goodness of fit of a predictive model, e
  - specially when the predictions are continuous numerical values.
- RMSE quantifies how well the predicted values from a model align with the actual observed values in the dataset. Here’s how it works:
  - Calculate the Squared Differences: For each data point, subtract the predicted value from the actual (observed) value, square the result, and sum up these squared differences.

    - Compute the Mean: Divide the sum of squared differences by the number of data points to get the mean squared error (MSE).
    - Take the Square Root: To obtain the RMSE, simply take the square root of the MSE.
- The formula for RMSE for a data with 'n' data points is as follows:
- $$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2}$$
- Where:
    - RMSE is the Root Mean Squared Error.
    - xi represents the actual or observed value for the i-th data point.
    - yi represents the predicted value for the i-th data point.
-

WHY DO WE NEED LEARNING?

- Learning: learning refers to the ongoing process of acquiring knowledge and skills that are essential for effectively working with data.
- Algorithms
- Super abilities of solving every task
- Experience
- AI
- Machine learning is programming computers to optimize a performance criterion using example data or past experience.
- There is no need to "learn" to calculate payroll
- Learning is used when:
    - Human expertise does not exist (navigating on Mars),
    - Humans are unable to explain their expertise (speech recognition)
    - Solution changes in time (routing on a computer network)
    - Solution needs to be adapted to particular cases (user biometrics)

How exactly do we teach machines


Data as input
Text files, spreadsheets, SQL Databases
Abstracting the data
Representation of the data in a structural format through the algorithm chosen.
The elementary learning happens here
Generalization
The practical application happens here where the learning from the previous step is used to develop an insight

-
- TYPES
  Supervised Learning (Learning by Example):
    - Imagine a teacher showing you labeled examples (data with answers).
    - You learn by looking at the inputs (e.g., pictures) and their corresponding outputs (e.g., labels saying "cat" or "dog").
    - Goal: Train a model to predict the correct output for new, unseen data.
    - Training data includes desired outputs

  Unsupervised Learning (Finding Hidden Patterns):
    - Imagine exploring a maze without a map.
    - You learn by observing the data itself, uncovering hidden patterns or structures.
    - Goal: Discover interesting groupings, trends, or relationships within the data.
    - Training data does not include desired outputs

  Semi-supervised Learning (Learning with a Little Help):
    - Imagine having a partially labeled map for the maze.
    - You learn from both labeled and unlabeled data, leveraging the labeled data to guide your exploration of the unlabeled data.
    - Goal: Improve the efficiency and accuracy of learning using a combination of labeled and unlabeled data.
    - Training data includes a few desired outputs

Reinforcement Learning (Learning by Trial and Error):

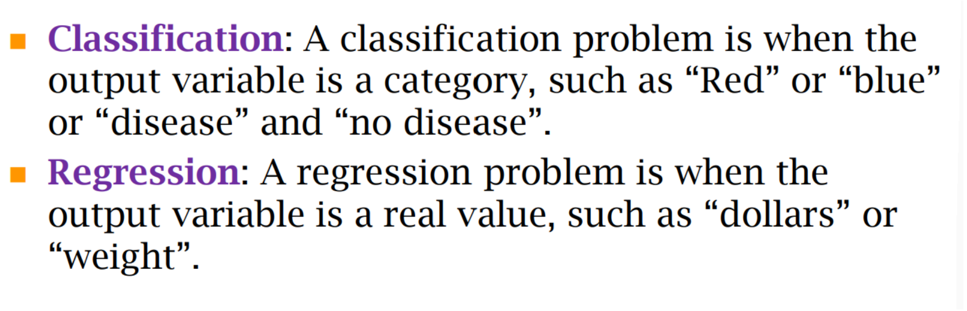
- Imagine learning a game through rewards and penalties.
- You take actions, receive rewards for good choices and penalties for bad choices, and learn to adjust your actions accordingly.
- Goal: Train a model to take optimal actions in an environment to maximize long-term rewards.
- Rewards from sequence of actions

SUPERVISED LEARNING

It is called supervised learning because the process of an learning(from the training dataset) can be thought of as a teacher who is supervising the entire learning process. Thus, the “learning algorithm” iteratively makes predictions on the training data and is corrected by the “teacher”, and the learning stops when the algorithm achieves an acceptable level of performance(or the desired accuracy).

STEPS:

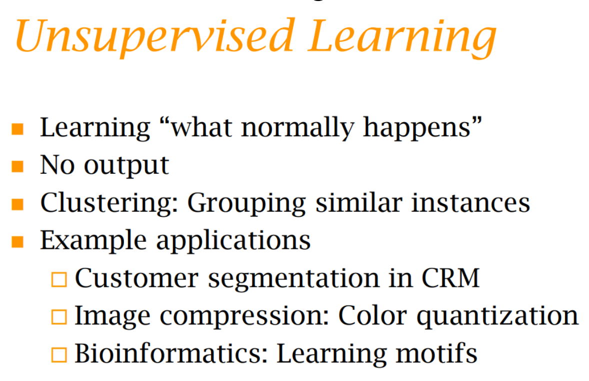
- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training dataset, test dataset, and validation dataset.
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set.
- If the model predicts the correct output, which means our model is accurate



- **Classification**: A classification problem is when the output variable is a category, such as “Red” or “blue” or “disease” and “no disease”.
- **Regression**: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

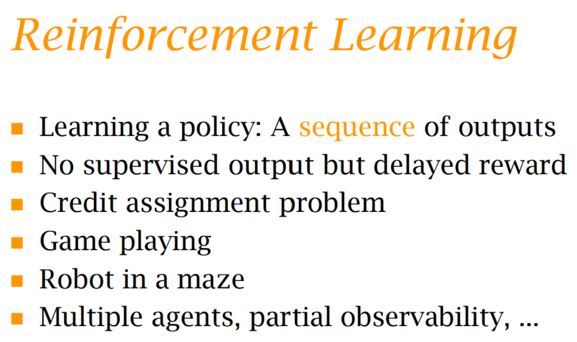
- **USES:**

- Prediction of future cases: Use the rule to predict the output for future inputs
- Knowledge extraction: The rule is easy to understand
- Compression: The rule is simpler than the data it explains
- Outlier detection: Exceptions that are not covered by the rule, e.g., fraud



Unsupervised Learning

- Learning "what normally happens"
- No output
- Clustering: Grouping similar instances
- Example applications
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - Bioinformatics: Learning motifs



Reinforcement Learning

- Learning a policy: A sequence of outputs
- No supervised output but delayed reward
- Credit assignment problem
- Game playing
- Robot in a maze
- Multiple agents, partial observability, ...

MODEL

- a system for mapping inputs to outputs
- represents a theory about a problem to predict house prices, we could make a model that takes in the square footage of a house andoutputs a price
- A model learns relationships between the inputs, called features, and outputs, called labels, from a training dataset
- A "model" in machine learning is the output of a machine learning algorithm run on data.
- A model represents what was learned by a machine learning algorithm.
- The model is the "thing" that is saved after running a machine learning algorithm on training data and represents the rules, numbers, and any other algorithm-specific data structures required to make predictions.
- The best analogy is to think of the machine learning model as a "program."

TRAINING DATA

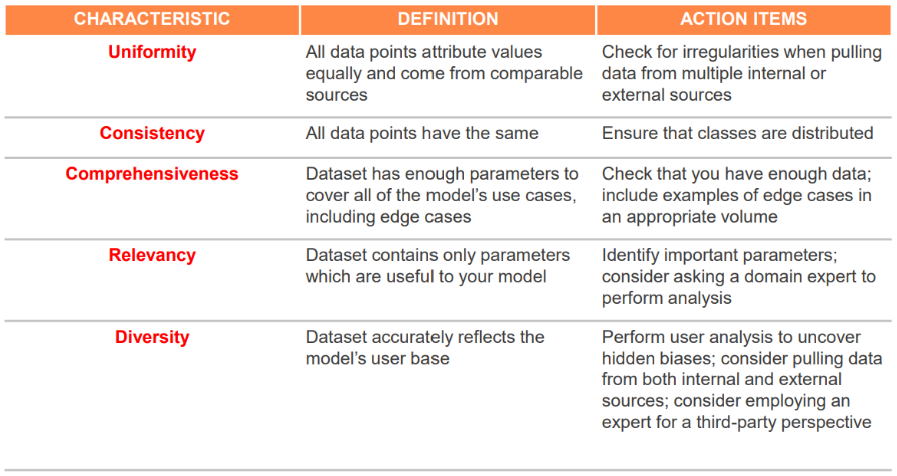
- Training data is used to help your machine learning model make predictions.
- It's the largest part of your dataset, forming at least 70-80% of the total data you'll use to build your model.
- This data is used exhaustively across multiple training cycles to improve the accuracy of your algorithm.

- Training data is different from validation and testing data in that its classes are often evenly distributed.
- Depending on your task, this might mean that the data doesn't accurately reflect its realworld use case

  - **How Much Training Data Do I Need?**
  - **Why is it Difficult to Estimate Dataset Size?**
  1. **Diversity of input**
  2. **Tolerance for errors**
  3. **Complexity of model**
  4. **Training method**

  The deciding factor for how much data you'll need is your project's unique requirements and goals. Each project requires a unique balance of all of these influencing factors, which you'll have to figure out for yourself when coming up with that target dataset size. Keeping this in mind, let's now dive into some of the ways that you can begin to figure out your data needs.
-

What is Quality?



CHARACTERISTIC	DEFINITION	ACTION ITEMS
Uniformity	All data points attribute values equally and come from comparable sources	Check for irregularities when pulling data from multiple internal or external sources
Consistency	All data points have the same	Ensure that classes are distributed
Comprehensiveness	Dataset has enough parameters to cover all of the model's use cases, including edge cases	Check that you have enough data; include examples of edge cases in an appropriate volume
Relevancy	Dataset contains only parameters which are useful to your model	Identify important parameters; consider asking a domain expert to perform analysis
Diversity	Dataset accurately reflects the model's user base	Perform user analysis to uncover hidden biases; consider pulling data from both internal and external sources; consider employing an expert for a third-party perspective

-

Underfitting and overfitting are common challenges encountered in machine learning, particularly when dealing with models that learn from data. Here's a breakdown of both concepts:

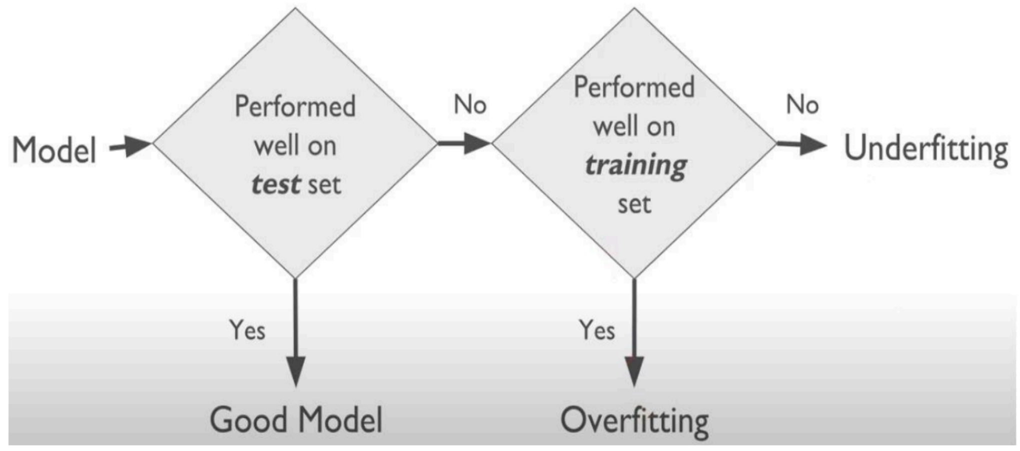
Underfitting:

- Imagine a student who memorizes only a few key facts for an exam but fails to grasp the underlying concepts.
- In machine learning, underfitting occurs when a model is too simple and fails to capture the important patterns in the training data. This results in:
  - High Bias: The model makes overly simplified predictions that are far from the actual values.
  - Poor Performance: The model performs poorly on both the training data and unseen data (generalization).
- Causes of underfitting:
  - Limited Model Complexity: Using a model that is too simple (e.g., a straight line for a complex dataset).
  - Insufficient Training Data: Not having enough data for the model to learn the underlying patterns effectively.

Overfitting:

- Imagine a student who memorizes every detail from their textbook but struggles to apply that knowledge to new situations.
- Overfitting happens when a model learns the training data too well, including the noise and irrelevant details. This leads to:
  - High Variance: The model's predictions are highly sensitive to small changes in the training data.
  - Poor Generalization: The model performs well on the training data but fails to generalize well to unseen data. It memorizes the specifics of the training data rather than learning the generalizable patterns.
- Causes of overfitting:
  - High Model Complexity: Using a model that is too complex for the data (e.g., a highly non-linear model for a simple dataset).
  - Limited Training Data: When the model complexity is high but the training data is limited, the model can overfit to the noise in the data.

Feature	Underfitting	Overfitting
Bias	High	Low
Variance	Low	High
Training Data Performance	Poor	Good
Generalization Performance	Poor	Poor
Model Complexity	Too simple	Too complex



CLASSIFICATION

Here's a breakdown of the core concepts:

- Supervised Learning: Unlike unsupervised learning (where patterns are discovered without labels), classification thrives on labeled data. Each data point has a pre-assigned class label (e.g., "spam" or "cat").
- The Model as a Classifier: The heart of classification is the model. This model, trained on the labeled data, learns to identify the characteristics that distinguish between different classes.
- Prediction Power: Once trained, the model can predict the class label for new, unseen data points. Imagine showing the model a new image, and it predicts whether it contains a cat or not.

Types of Classification Problems:

- Binary Classification: The simplest scenario, involving two classes (e.g., spam/not spam).
- Multi-Class Classification: More than two classes are involved (e.g., classifying handwritten digits as 0, 1, 2, etc.).

Common Classification Algorithms:

- Logistic Regression: A powerful technique for binary classification problems.
- K-Nearest Neighbors (KNN): Classifies a data point based on the majority class of its nearest neighbors in the training data.
- Support Vector Machines (SVM): Creates a hyperplane that best separates the data points of different classes.
- Decision Trees: Learns a tree-like structure where each branch represents a decision rule leading to a class label.
- Random Forests: Combines multiple decision trees for improved accuracy and robustness.

Applications of Classification are Vast:

- Spam Filtering: Classifying emails as spam or not spam.
- Image Recognition: Categorizing images based on their content (e.g., cats, dogs, cars).
- Fraud Detection: Identifying fraudulent transactions on credit cards.
- Medical Diagnosis: Classifying medical images to support diagnoses.
- Customer Segmentation: Grouping customers based on their characteristics for targeted marketing.

1. Binary Classification: A Two-Class Showdown
   - Imagine sorting emails into "spam" or "important." That's the essence of binary classification. The model is trained to distinguish between exactly two mutually exclusive classes.
   - Examples:
     - Spam filtering (spam/not spam)
     - Medical diagnosis (positive/negative test result)
     - Image classification (cat/dog)

How it Works:

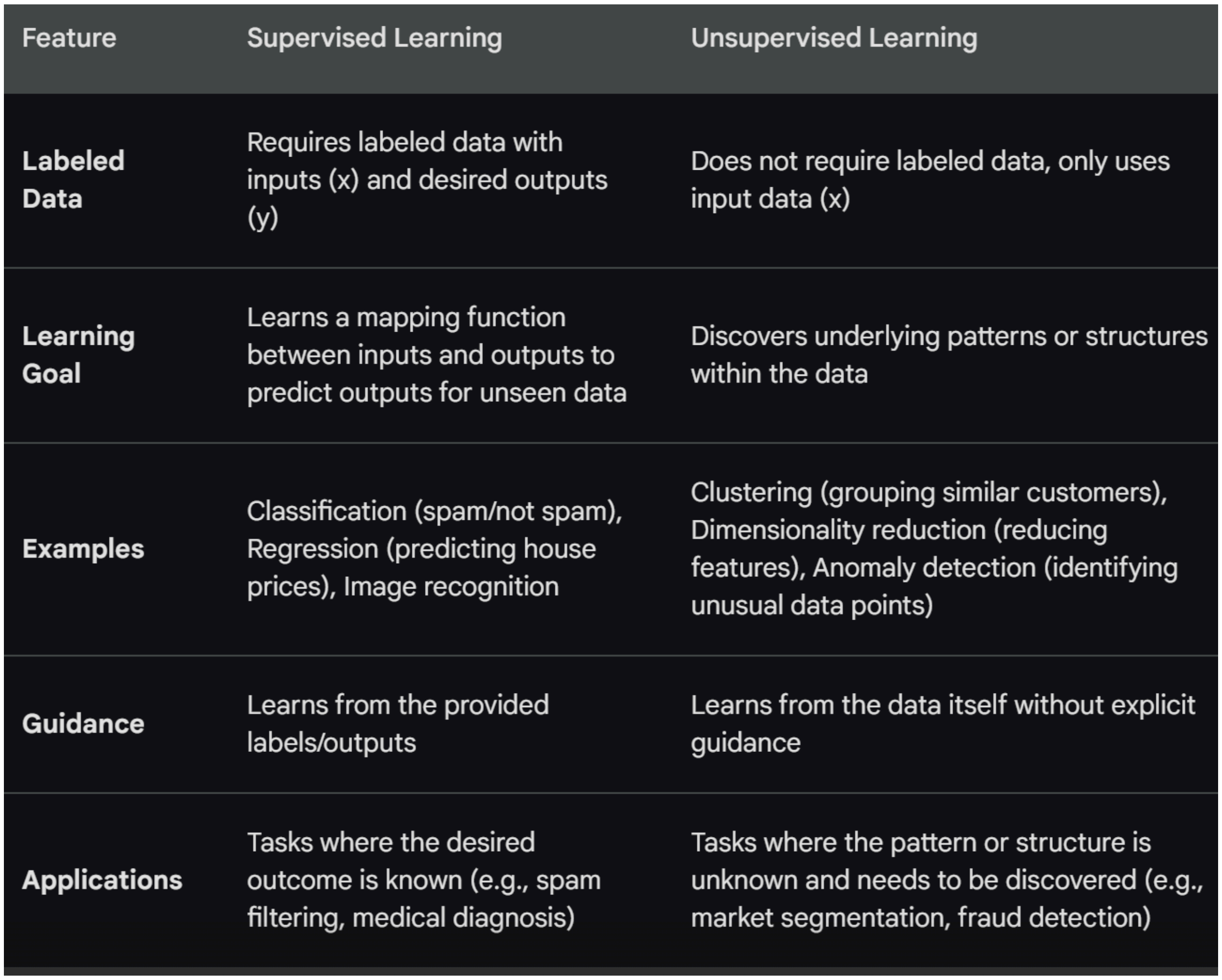
- The model learns a decision boundary that separates the data points belonging to the two classes in the feature space.
- Feature space: This refers to the space where each data point is represented by its features (characteristics).
- During prediction, the model takes a new, unseen data point and classifies it based on which side of the decision boundary it falls on.

2. Multi-Class Classification: Beyond the Binary Divide
   - Imagine classifying handwritten digits as 0, 1, 2, 3, etc. This is multi-class classification, where the model deals with more than two classes.
   - Examples:
     - Handwritten digit recognition (0, 1, 2, ..., 9)
     - Image classification (cat, dog, bird, ...)
     - Sentiment analysis (positive, negative, neutral)

How it Works:

- The model learns multiple decision boundaries or decision functions to separate the data points belonging to different classes.
- There are various techniques for multi-class classification, such as one-vs-rest (treating each class vs all others) or multi-class logistic regression.
- During prediction, the model assigns the new data point to the class with the highest probability or score based on the learned decision functions.

Feature	Binary Classification	Multi-Class Classification
Number of Classes	Exactly two	More than two
Decision Boundary	One decision boundary separates the classes	Multiple decision boundaries or functions separate the classes
Applications	Simpler classification tasks with two clear categories	More complex tasks with diverse categories



Feature	Supervised Learning	Unsupervised Learning
Labeled Data	Requires labeled data with inputs (x) and desired outputs (y)	Does not require labeled data, only uses input data (x)
Learning Goal	Learns a mapping function between inputs and outputs to predict outputs for unseen data	Discovers underlying patterns or structures within the data
Examples	Classification (spam/not spam), Regression (predicting house prices), Image recognition	Clustering (grouping similar customers), Dimensionality reduction (reducing features), Anomaly detection (identifying unusual data points)
Guidance	Learns from the provided labels/outputs	Learns from the data itself without explicit guidance
Applications	Tasks where the desired outcome is known (e.g., spam filtering, medical diagnosis)	Tasks where the pattern or structure is unknown and needs to be discovered (e.g., market segmentation, fraud detection)

Bias and variance
are two crucial concepts in machine learning, particularly when dealing with model generalization - the ability of a model to perform well on unseen data. Both bias and variance contribute to a model's overall error, and finding the right balance is essential for optimal performance.
Bias:

- Think of it as a systematic error. Imagine a student who consistently underestimates the answer on a test due to a misunderstanding of a key concept.
- In machine learning, bias refers to the systematic difference between the model's predictions and the actual values. This can happen due to:
    - Overly simplistic models: A model that is too simple may not be able to capture the complexity of the data, leading to consistently inaccurate predictions.

    - Incorrect assumptions: If the model's underlying assumptions about the data are wrong, it will lead to biased predictions.
    - Limited training data: If the model is trained on insufficient data that doesn't represent the entire population, it may learn biased patterns specific to that data.

High Bias Effects:

- The model consistently underestimates or overestimates the true value, regardless of the specific data point.
- The model performs poorly on both the training data and unseen data because it fails to learn the generalizable patterns.

Variance:

- Think of it as the model's sensitivity to changes in the training data. Imagine a student who gets excellent scores on practice tests from their textbook but struggles with slightly different questions on the actual exam.
- In machine learning, variance refers to the model's sensitivity to the specific training data it was exposed to. This can happen due to:
    - Overly complex models: A model with too many parameters can become overly sensitive to the specifics of the training data and fail to generalize well to unseen data.
    - High noise in the training data: If the training data contains a lot of noise or irrelevant information, the model may learn to fit that noise, leading to high variance.

High Variance Effects:

- The model's predictions can vary significantly depending on the specific training data it was trained on.
- The model performs well on the training data but fails to generalize well to unseen data, leading to poor performance on new data points.

LAZY LEARNING AND EAGER LEARNING

In machine learning, choosing the right learning approach is crucial for efficient and effective model training. Here's a breakdown of lazy learning and eager learning, two contrasting paradigms:

Lazy Learning (Instance-Based Learning):

- Think of it as a student who waits until the exam to cram all the information. Lazy learning models **postpone the heavy lifting of learning until prediction time.**

- Characteristics:
    - Stores training data: Lazy learners store the entire training data set without any upfront processing or model building.
    - Learning at prediction: When a new data point arrives for prediction, the model compares it to the stored training data to make a classification or prediction. This comparison typically involves finding the nearest neighbors (similar data points) in the training data.
    - Examples: K-Nearest Neighbors (KNN), Instance-based learning algorithms

Advantages of Lazy Learning:

- Simpler training phase: Requires less computational resources and time for training compared to eager learning.
- Adapts to new data: Can easily incorporate new data points into the training data without retraining the model.
- No model assumptions: Doesn't require prior assumptions about the underlying data distribution.

Disadvantages of Lazy Learning:

- Slower prediction: Predicting new data points can be computationally expensive as it involves searching through the entire training data.
- High memory usage: Storing the entire training data can require significant memory resources, especially for large datasets.
- Interpretability challenges: Can be difficult to interpret the reasoning behind a prediction as it relies on comparisons to training data points.

Eager Learning:

- Think of it as a student who diligently studies the entire semester to prepare for the exam. Eager learning models build a generalized model from the training data before encountering any new data for prediction.
- Characteristics:
    - Model building during training: Eager learners analyze and process the training data during the training phase to build a model that represents the relationships between features and outputs.
    - Prediction using the model: Once trained, the model can be used to make predictions for new, unseen data points by applying the learned relationships.
    - Examples: Decision Trees, Support Vector Machines (SVM), Linear Regression

Advantages of Eager Learning:

- Faster prediction: Predictions for new data points are typically faster compared to lazy learning as the model has already been built.
- Lower memory usage: Stores a compact model instead of the entire training data, reducing memory requirements.
- Interpretability: In some cases, the model's structure and decision-making process can be easier to understand and interpret.

Disadvantages of Eager Learning:

- Complex training phase: Training can be computationally expensive and time-consuming, especially for complex models and large datasets.
- Less adaptable to new data: Adding new data points typically requires retraining the entire model, which can be inefficient.
- Model assumptions: Relies on assumptions about the underlying data distribution, which may not always hold true.

Choosing the Right Approach:

The choice between lazy learning and eager learning depends on several factors, including:

- Dataset size: For large datasets, lazy learning's storage requirements might outweigh its benefits.
- Prediction speed: If real-time predictions are crucial, eager learning's faster prediction times may be preferred.
- Model interpretability: If understanding the model's reasoning is important, eager learning with interpretable models might be a better choice.
- Data access: If new data is constantly added, lazy learning's ease of incorporating new data points could be advantageous.

1. Decision Tree:
A decision tree is a supervised learning algorithm that works by recursively splitting the data based on features (attributes) to create a tree-like model for classification or regression tasks.

- Imagine a flowchart where each internal node represents a feature (e.g., income level) and each branch represents a possible value of that feature (e.g., high, low).
- The tree splits the data at each node based on a decision rule (e.g., if income is high, go left).
- Leaves at the end of the tree represent the predicted class (e.g., credit risk: high, low) or a continuous value in regression.

Advantages:

- Easy to interpret and visualize
- Handles both categorical and numerical features
- No need for feature scaling

Disadvantages:

- Prone to overfitting if not regularized
- Can be sensitive to small changes in the data

2. Threshold for Classification:
In binary classification problems (two classes), a threshold is a specific value used to classify data points based on the model's output.

- The model might output a probability (between 0 and 1) or a score indicating class membership.
- We set a threshold (e.g., 0.5) to categorize data points:
    - Above the threshold: Classified as class 1
    - Below the threshold: Classified as class 2

The choice of threshold can impact the model's performance and can be adjusted based on the cost of misclassification (e.g., a higher threshold for a critical medical diagnosis).

3. Metric for Classification - Accuracy:
Accuracy is a common metric for evaluating the overall performance of a classification model. It simply calculates the percentage of predictions that the model got correct.

- Accuracy = (True Positives + True Negatives) / Total Samples
- True Positives (TP): Correctly classified positive cases
- True Negatives (TN): Correctly classified negative cases

Limitations of Accuracy:

- Accuracy can be misleading in imbalanced datasets (unequal class distribution).
- A model might just predict the majority class all the time and still get high accuracy.

4. F1 Score:
The F1 score is a metric that addresses the limitations of accuracy by considering both precision and recall.

- Precision: Measures how many of the predicted positives were actually correct (TP / (TP + FP))
- Recall: Measures how many of the actual positives were identified by the model (TP / (TP + FN))
- F1 Score: Harmonic mean of precision and recall (2 * (Precision * Recall) / (Precision + Recall))

The F1 score provides a balanced view of a model's performance, especially for imbalanced datasets.
5. Confusion Matrix:
A confusion matrix is a table that visually summarizes the performance of a classification model. It shows the number of actual cases (rows) classified into each predicted class (columns).

- Example:

Predicted Class	Actual Class 1	Actual Class 2
Class 1	True Positives (TP)	False Positives (FP)

Class 2	False Negatives (FN)	True Negatives (TN)

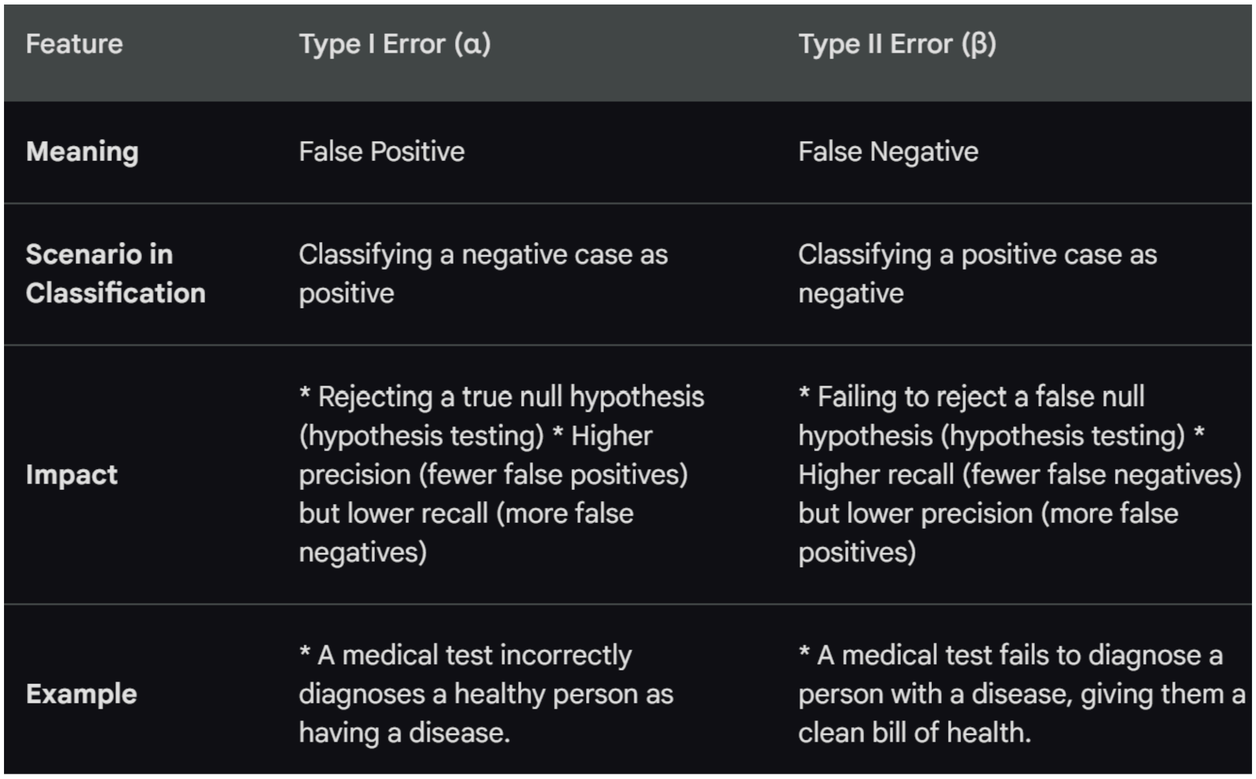
The confusion matrix helps identify potential issues like high false positives (classifying negative cases as positive) or high false negatives (missing positive cases).

6. Type I and Type II Errors:

These concepts are also known as alpha and beta errors and refer to the two possible errors in hypothesis testing:

- Type I Error (α) - False Positive: Rejecting a true null hypothesis. In classification, this translates to classifying a negative case as positive.
- Type II Error (β) - False Negative: Failing to reject a false null hypothesis. In classification, this translates to classifying a positive case as negative.

The choice of a threshold can influence the balance between these errors. A lower threshold might decrease Type I errors (fewer false positives) but increase Type II errors (more false negatives).



Feature	Type I Error (α)	Type II Error (β)
Meaning	False Positive	False Negative
Scenario in Classification	Classifying a negative case as positive	Classifying a positive case as negative
Impact	* Rejecting a true null hypothesis (hypothesis testing) * Higher precision (fewer false positives) but lower recall (more false negatives)	* Failing to reject a false null hypothesis (hypothesis testing) * Higher recall (fewer false negatives) but lower precision (more false positives)
Example	* A medical test incorrectly diagnoses a healthy person as having a disease.	* A medical test fails to diagnose a person with a disease, giving them a clean bill of health.

ID3

ID3 Algorithm

ID3 is a fundamental decision tree learning algorithm used for classification tasks. It works by recursively splitting a dataset based on features (attributes) to create a tree-like model that predicts the class of new, unseen data points.

Key Concepts:

- Top-Down Approach: ID3 starts at the root node and iteratively splits the data into subsets based on the feature that best separates the classes.
- Entropy: ID3 uses entropy, a measure of uncertainty in the data, to choose the splitting feature. The feature with the highest information gain (reduction in entropy) is chosen for the split.
- Information Gain: Information gain measures how much a specific feature helps in distinguishing between different classes.
- Stopping Criteria: ID3 stops splitting a node when it reaches a pure node (all data points belong to the same class), when there are no more features to split on, or when a predefined depth limit is reached.

ID3 Algorithm Steps:

1. Start with the entire training dataset.
2. Calculate the entropy of the target variable (class labels).
3. For each feature:
    - Calculate the entropy of the target variable after splitting the data based on the values of that feature.
4. Choose the feature that results in the highest information gain (greatest reduction in entropy).
5. Create a decision node in the tree for the chosen feature.
6. For each value of the chosen feature, create a branch in the tree and assign the corresponding subset of data to that branch.
7. Repeat steps 2-6 for each branch until a stopping criterion is met.
8. The resulting tree can be used to classify new data points by following the branches based on the data point's feature values.

Information gain is the statistical quantity measuring how well an attribute classifies the data. In other words in order to select the best attribute we need to first calculate the information gain for each attribute and then choose the attribute with the greatest information gain

Entropy, which measures information content of a *random process*.

https://towardsdatascience.com/decision-trees-for-classification-id3-algorithm-explained-89df76e72df1