Machine learning

AI and ML
A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if **its performance at tasks in T, as measured by P, improves with experience E**.
The field of study that gives computers the ability to **learn without explicitly being programmed**.
A decision rule is a logical statement or a set of conditions used to make decisions or predictions based on input data.

Data in ML

Types of Data

- **Training**: To train the model.
- **Validation**: To tune hyperparameters and evaluate performance during training, to select the most appropriate model.
- **Test**: Assess final performance after training.

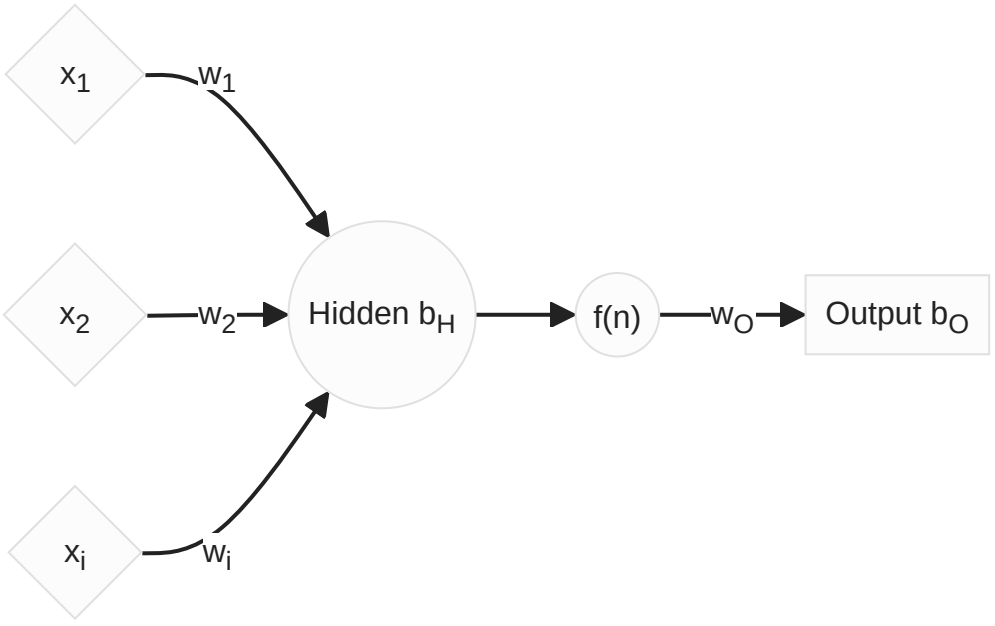
Evaluating performance

- **Cross Validation**: Assess how well a predictive model generalizes to an independent dataset. The data is divided into k subsets, and the model is trained k times, each time using a different subset as the test set and the remaining data as the training set.
- **Leave One Out**: A special case of k-fold cross-validation where k equals the number of instances in the dataset. Each instance is used as the test set once while the rest of the data is used for training.
- **Holdout**: Splitting the dataset into two subsets: one for training the model and the other for evaluating its performance. It's commonly used when there's a large amount of data available.
- **Resubstitution**: Testing it on the same data that was used for training.
  Precision: Percentage of detection that is correct.
  Recall: Percentage of true positives detected.
  -> Draw the curve between precision and recall, we can obtain more criteria, including ROC curves, average precision and so on.

Neural network

An artificial neural network (ANN) is a computational model inspired by the structure and functioning of biological neural networks, like the human brain. It's composed of interconnected nodes, called neurons, organized into layers. These layers typically include an input layer, one or more hidden layers, and an output layer.
The components of an artificial neural network typically include:

- **Neurons**: The fundamental building blocks of neural networks. Neurons receive inputs, process them using an activation function, and produce an output.
- **Connections**: Neurons are connected to each other via connections, which carry signals from one neuron to another. Each connection has an associated weight that determines its strength.
- **Layers**: Neurons are organised into layers. The input layer receives external data, the hidden layers process this data, and the output layer produces the final result.
- **Activation Function**: An activation function determines the output of a neuron based on its inputs. It introduces non-linearity into the network, enabling it to learn complex patterns.
- **Weights and Biases**: Weights represent the strength of connections between neurons, determining how much influence one neuron has on another. Biases allow neurons to adjust their output independently of the inputs.



$$Output = b_O + w_O \cdot f(b_H + \sum_i x_i w_i)$$

The output of a unit is a parameterized non-linear function of its inputs. Neural networks are universal approximators, that is, they can represent an approximation of any continuous function.
Learning a neural network: given a set of examples (training data), find parameter values such that the model fits the data.

Backpropagation learning

Inputs:

- A neural network architecture including all units and their connections
- Stopping criterion
- Learning rate (constant of proportionality of gradient descent search)
- Initial parameter values
- A set of training data with the ground truth labels about the classes
  Output: Updated parameter values

Algorithm

An iterative procedure combining feed-forward operation and back-propagation of the error. Repeat until the stopping criterion is met:

- Evaluate the network on each example given the current parameter settings.
- Determine the error derivative for each parameter.
- Update each parameter in proportion to its derivative.
  Gradient descent is a key optimization algorithm, adjusting the weights and biases of the network in order to minimize the loss function, which measures the discrepancy between the network's predictions and the actual target values.

1. **Compute Gradients**
2. **Update Weights and Biases**:

   Mathematically, the weight update for a particular weight parameter w is given by:

   $$w_{new} = w_{old} - learning\_rate \times gradient$$

   Similarly, for biases, the update is done similarly but with biases instead of weights.
3. **Iterative Process**: Until loss function reaches a satisfactory minimum or stabilizes.

Types of ML

Supervised:

- Labelled data, make predictions, decisions.
- Include regression (continuous) or classification (discrete).
- Common problems:
  - Overfitting: Well on training, not well on test.
  - Underfitting: Not well on training, well on test.
  - Curse of dimensionality: Challenges with high-dimensional spaces.

Generalization: Balance between overfitting and underfitting -> Good
Unsupervised:

- Unlabelled data, find hidden pattern/structure.
- Include dimensional reduction (continuous) or clustering (discrete).
- Common problems: Evaluating performance, Missing data, Number of features.

Reinforcement:

- Make sequential decisions by interacting with the environment and get feedback, i.e., rewards and penalties.
- Common problems: Exploration vs Exploitation, Reward system.

Discrete: Finite number of values, i.e., categories or integers.
Continuous: Infinite number of values within a certain range, i.e., real numbers.'

Tasks in ML

- Regression is about predicting a continuous outcome based on input variables. Ex: drawing a line through scattered points to predict where the next point will fall.
- Classification is like sorting objects into predefined categories based on their features or attributes. Ex: putting things into labelled boxes.
  - The perceptron is a type of neural network that takes input values, applies weights to them, and produces an output based on whether the weighted sum exceeds a certain threshold.
  - The perceptron cost function is a measure used to evaluate the performance of a perceptron model in binary classification tasks.
  - Decision boundary is a hypersurface that separates different classes in a classification problem.
  - Decision region is an area in the input space where all the points are assigned to the same class by the classification model.
- Clustering involves grouping similar objects together based on their characteristics or features. Ex: finding natural groupings in a pile of mixed items.
- Dimensionality reduction aims to simplify complex data by reducing the number of variables while preserving important information. Ex: summarizing a long story into a few key points.

Regression: Linear regression

> Model relationship between 1 or more independent variables and 1 dependent variables by fitting a linear equation to the observed data points.

Calculate the mean of the independent and dependent variables. Compute the slope and intercept of the line that best fit the data points, minimising the sum of squared differences

between the observed and predicted value. Use this equation to predict the dependent's value for new inputs.

Pros	Cons
Simple to understand	Assume linearity
Quick computation	Sensitive to outliers in data
Provide insights of variables' relationship	Cannot capture complex pattern

Classification: Decision trees

Decision tree: A tree-like structure where each internal node represents a "test" on an attribute (also known as a feature), each branch represents the outcome of the test, and each leaf node (or terminal node) holds the class label or regression value. Contains:

- Non-leaf nodes: Nodes that do not contain the final decision. They represent attributes or features on which the decision tree splits the data.
- Leaf nodes: Terminal nodes of the decision tree. They contain the final decision or the predicted class label/regression value.
- Splitting rules: True (go left), False (go right)

> Classify or predict the outcome of a target variable by recursively splitting the dataset into subsets based on the values of input features.

Start with the entire dataset and choose the best feature to split the data into two subsets. Repeat this process for each subset until all data points within each subset belong to the same class or until a stopping criterion is met. This creates a tree-like structure where each internal node represents a decision based on a feature, and each leaf node represents a class label or prediction.

Pros	Cons
Simple to understand	Overfitting
Handles both numerical and categorical data	Sensitive to small variations
Can capture non-linear relationships	Cannot capture subtle relationships between features

Number of bits: $\log_2 P(x)$

Given a distribution over a set of possible symbols, to identify a member in a set/sequence of symbols, the expected number of bits is: $-\sum_x P(x) \cdot \log_2 P(x)$

Classification: k-nearest neighbour

> Classify or predict the class of a new data point based on the majority class of its nearest neighbours in the feature space.

Calculate the distance between the new data point and all other points in the dataset using a distance metric such as Euclidean distance. Select the k nearest neighbours based on these distances. Finally, for classification, assign the most common class among the k neighbours to the new data point; for regression, take the average of the values of the k neighbours.

Pros	Cons
Simple to understand	Computational inefficiency for large datasets
No training phase	Sensitive to irrelevant features
Handle different types of data	Need to choose optimal k

Classification: Random forest

> Build an ensemble of decision trees that collectively make predictions or classifications

Select a subset of data from the dataset. Construct a decision tree using the selected subset by repeatedly splitting the data based on the features that provide the best separation. Repeat this process to create multiple decision trees. Finally, for a new data point, let each tree in the forest make a prediction, and then aggregate the predictions to determine the final output.

Pros	Cons
High accuracy	Computational inefficiency for large datasets
Robust to overfitting	Difficult to understand
Can handle large datasets with ease	Not suitable for imbalanced datasets

Pruning is to remove redundant leaf nodes to trade off the overfitting on the training data.
Feature bagging: randomly sample a subset of feature candidates for splitting.
Dataset bootstrap: create a bootstrapped data by sampling a subset of dataset. It is OK to repeat the same data when sampling.

Clustering: k-means

> Partition a dataset into k clusters based on similarity, aiming to minimize the within-cluster sum of squares.

Randomly initialize k cluster centers. Iteratively assign each data point to the nearest cluster center and update the cluster centers based on the mean of the points assigned to each cluster. Repeat these steps until convergence, typically defined by minimal change in cluster assignments or cluster centers.

Pros	Cons
Simple to understand	Need to choose optimal initial centroids
Computationally efficient for larger data sets	Dependent on number of clusters k
Can handle large datasets with ease	Assumption that clusters are spherical and similar in sizes

Clustering: DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

> Identify clusters of varying shapes and sizes in a dataset by grouping together points that are closely packed, while also marking outliers as noise.

Select a point in the dataset. If this point has a sufficient number of neighboring points within a specified distance (epsilon), it becomes a core point and forms a cluster. Expand this cluster by adding nearby points to it, recursively, until no more points can be added. Then, repeat this process for other points in the dataset, assigning them to existing clusters or marking them as noise if they don't belong to any cluster.

Pros	Cons
Robust to outliers in data	Difficult to choose parameters
Automatic determination of cluster numbers	Sensitivity to density
Handle clusters with different shapes and sizes	Computational inefficiency for large datasets

Clustering: AutoEncoder

> Learn a compressed representation of input data by training a neural network to reconstruct the input as accurately as possible.

An encoder network compresses the input data into a lower-dimensional representation, often called the latent space. Then, a decoder network reconstructs the original input from this compressed representation. During training, the AutoEncoder minimizes the difference between the input and the output, encouraging the model to learn meaningful features and capturing the essential information in the data.

Pros	Cons
Dimensionality reduction	Careful tuning of hyperparameters
Unlabelled data	Reconstruction loss
Extract relevant features from raw data	Limited interpretability

Dimension reduction: PCA (Principal Component Analysis)

> Reduce the dimensionality of a dataset while preserving most of its original variability.

Standardize the data to have a mean of 0 and a standard deviation of 1. Then, compute the covariance matrix of the standardized data. Next, calculate the eigenvectors and eigenvalues of the covariance matrix. Finally, select the top k eigenvectors corresponding to the largest eigenvalues to form the principal components, which represent the new lower-dimensional space.

Pros	Cons
Dimensionality reduction	Linearity assumption
Noise reduction	Loss of interpretability
Can handle large datasets	Sensitivity to scaling

Reinforcement learning: Q-learning

> Enable an agent to learn an optimal policy for decision making in a Markov Decision Process (MDP) through exploration and exploitation of actions.

Initialize a Q-table with all state-action pairs and their corresponding Q-values. Then, iteratively interact with the environment, selecting actions based on an exploration-exploitation strategy and updating Q-values using the Bellman equation. Repeat this process until convergence to find the optimal Q-values for each state-action pair.

Pros	Cons
Versatility	High memory required to store Q-vales
Suitable for unknown or complex environments	Slow convergence
Convergence to optimal policy	Exploration-exploitation tradeoff