Introduction to Machine Learning

1. Fundamentals of Machine Learning
2. Learning Paradigms and Machine Learning Tasks
3. Task vs Dataset - Learning the wrong task
4. Performance metrics in Machine Learning
5. Loss functions in Machine Learning

1. Fundamentals of Machine Learning

- **Learn from Data**. Computers can learn from data without being explicitly programmed for every specific task (rule-based → data-driven).

- **Identify Patterns**. Algorithms can automatically find patterns, relationships, and insights within data.

- **Generalization**. Ability to make predictions on new, unseen data

- **Algorithms and Models**. Algorithms (sets of instructions) are used to build models (representations of the learned patterns).

2. Learning Paradigms and Machine Learning Tasks

Machine learning tasks can be categorized according to the learning paradigm and the specific goal they pursue:

- Supervised Learning
- Unsupervised Learning
- Self-Supervised Learning
- Reinforcement Learning

Supervised Learning

- Each input data is associated with a known output or target variable.
- **Goal**: Learn a *mapping* from input to output data.

Some common tasks: **Classification** & **Regression**

Classification Task

- **Goal**: Assign data points to predefined categories or classes.
- The output variable is categorical.
  - **Binary Classification**: Predicting one of two classes.
  - **Multi-class Classification**: Predicting one of more than two classes.

- Examples:
  - Image Classification: Identifying objects in an image (e.g., cat, dog, car).
  - Spam Detection: Classifying emails as spam or not spam.
  - Medical Diagnosis: Determining if a patient has a certain disease based on symptoms and test results.
  - Fraud Detection: Identifying fraudulent transactions based on user behavior and transaction details.

A model that fits a type of task can be used regardless of what the inputs and outputs represent.

- If a model fits a binary text classification task, it can be used for:
  - Spam Detection: Spam ↔ Not Spam
  - Topic Classification: Relevant ↔ Irrelevant
  - Fake News Detection: Fake ↔ Real
  - Bot Detection in Social Media: Bot ↔ Human
  - Political Orientation Classification: right ↔ left political ideology

Regression Task

- **Goal**: Predict a continuous numerical value.
- The output variable is continuous.
  - **Linear Regression**: Linear relationship with the input features.
  - **Polynomial Regression**: Polynomial relationship with the input features.
  - ...
  - **Time Series Forecasting**: Predicting future values based on past values
- Examples:
  - Feature-based Estimations
    - Predict a house selling price given features like the size of the house, number of bedrooms, location, and age.
    - Insurance risk estimation based on customer demographics, driving history, and other risk factors
  - Feature & Time based Estimations (Time Series Forecasting)
    - Stock Market Prices estimation based on historical stock data, economic indicators, and company performance.
    - Energy demand prediction based on weather patterns, time of day, and historical demand.

Categorical vs discrete output variables

Output variable can be discrete, but not categorical:

- Sentiment Analysis, e.g. classifying text (e.g., reviews, tweets) as positive, negative, or neutral.
- Rating Prediction, e.g. predicting a song or movie rating on a scale of 1 to 5 stars

Different approaches:

- Use a regression model and discretize the estimation
- Use a classification model (with loss of ordinal information or ignoring numerical relationships)

Unsupervised Learning

- The algorithm learns from unlabeled data, without any explicit output or target variable.
- **Goal**: Discover hidden patterns or structure in the (input) data

Some common tasks: Clustering & Dimensionality Reduction (PCA, t-SNE, ...)

Self-Supervised Learning

- The algorithm learns useful representations of the data by creating "artificial" labels from the unlabeled data itself.
  - E.g. a system that creates a lower-dimensional representation of the (corrupted) input data and tries to reconstruct the original (clean) input data.
- **Goal**: Learn useful representations from the (input) data.

Some common tasks: Word embeddings

Word embeddings

- Low-dimensional (50-300) numerical representation of words
- Capture the semantic meaning and relationships between words
  - Words with similar meanings are close in the embedding space
- Mathematical operations reflect semantic relationships
  - `vector("king")` - `vector("man")` + `vector("woman")` ≈ `vector("queen")`
- Language dependent, already available for many languages

Reinforcement Learning

- The algorithm (*agent*) learns through trial and error by interacting with the environment and observing the consequences of its actions (rewards/penalties).
  - Like learning through experience
- **Goal**: Learn an optimal policy to maximize cumulative reward.

Some common tasks: Game Playing, Autonomous Driving, Chats

3. Task vs Dataset - Learning the wrong task

- The model solves the problem present in the dataset
  - Input data → Output data
  - The database may not represent the task it is intended to.
  - Dataset bias → learn **spurious correlations**
- Random sampling for unbiased datasets
  - No selection bias
  - Better reflects the true underlying distribution of the population you want your model to generalize to.
  - Machine learning statistical methods assume that the data is a random sample.
- In most cases (almost all), datasets are not created from random samples.

Some examples of dataset bias:

- Alzheimer detection from voice. Dataset: Recordings of people who have developed the disease and healthy people → voice disorder, age, channel...
- Language identification in phone calls. Dataset: phone calls → age, gender, channel...
- Fake news detection. Dataset: Fake news from Twitter vs news agencies (AP, Reuters...) → source format
- AI-generated text detection: Dataset some text created by humans and AIs → spelling error, rich vocabulary... (maybe not a spurious correlation? 😅)

Always question what task you are solving

Even more if your model results are surprisingly good...

4. Performance metrics in Machine Learning

- Get an estimation of the performance of the model.
- Depending on the concrete task, there are different metrics
  - Binary Classification
  - Multiclass Classification
  - Regression

Binary Classification - Confusion Matrix

- Visualizes the performance by comparing the predicted labels to the actual labels:

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (**TP**)	False Negative (**FN**)
Actual Negative	False Positive (**FP**)	True Negative (**TN**)

Binary Classification - Aggregate Metrics

- **Accuracy**. The most intuitive metric:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \in [0, 1]$$

- **Precision**. The ratio of TP instances to all the instances the model predicted as positive. Useful in false-positive high-cost scenarios.

$$Precision = \frac{TP}{TP + FP} \in [0, 1]$$

- **Recall (Sensitivity, True Positive Rate)**. The ratio of TP instances to all the actual positive instances. Useful in false-negative high-cost scenarios.

$$Recall = \frac{TP}{TP + FN} \in [0, 1]$$

Binary Classification - Aggregate Metrics

- **F1-Score**. Harmonic mean of precision and recall (penalizes extreme values).

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \frac{2 \cdot TP}{2 \cdot TP + FN + FN} \in [0, 1]$$

- **Specificity (Sensitivity, True Negative Rate)**. The ratio of TN instances to all the actual negative instances.

$$Specificity = \frac{TN}{TN + FP} \in [0, 1]$$

- **False Positive Rate**. The ratio of FP instances to all the actual negative instances.

$$FPR = \frac{FP}{TN + FP} = 1 - Specificity \in [0, 1]$$

- **False Negative Rate**. The ratio of FN instances to all the actual positive instances.

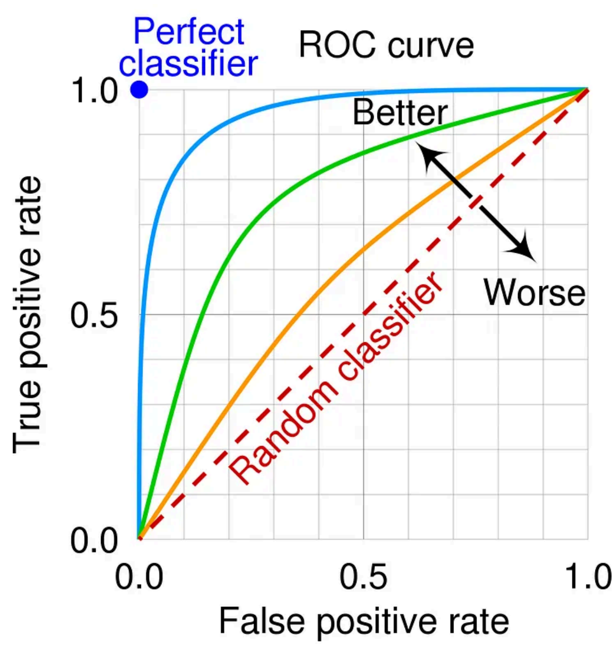
$$FNR = \frac{FN}{TP + FN} = 1 - Recall \in [0, 1]$$

Binary Classification - ROC curve

ROC Curve: Receiver Operating Characteristic Curve.

- Given input data x, binary classifiers tipically use a threshold over the posterior probability $P(y = 1|x)$ to decide positive/negative output.
  - If $P(y = 1 \mid x) \geq threshold$, predict the $y = 1$ positive class.
  - If $P(y = 1 \mid x) < threshold$, predict the $y = 0$ negative class.
  - Calibrated system and balanced error costs (FP vs FN) → $threshold = 0.5$
  - $threshold \approx$ **operation point**

- ROC curve plots the True Positive Rate (TPR) vs the False Positive Rate (FPR) over $threshold \in [0, 1]$
  - $threshold = 1$ → all negatives → $TPR = 1$ and $FPR = 1$
  - $threshold = 0$ → all positives → $TPR = 0$ and $FPR = 0$



Binary Classification - AUC

AUC: Area Under the (ROC) Curve

- Aggregate metric based on ROC
  - Perfect Classifier: $AUC = 1$
  - Random Classifier: $AUC = 0.5$
  - Better than Random Classifier: $AUC > 0.5$
  - Worse than Random Classifier: $AUC < 0.5$

Multiclass Classification - Confusion Matrix

- n-class Classification task → $n \times n$ confusion matrix

Actual \ Predicted	Class 1	...	Class n
Class 1	C_{11}	$\cdots$	C_{1n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$
Class n	C_{n1}	$\cdots$	C_{nn}

- **Accuracy**. The ratio of correctly classified samples to the total number of samples

$$Accuracy = \frac{trace(C)}{sum(C)} = \frac{\sum_{i=1}^{n} C_{ii}}{\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}} \in [0, 1]$$

- **Precision, Recall, and F1-Score** (Per Class and Averaged)

Regression - Performance Metrics

- **GOAL**: Quantify the difference between the predicted/actual values.

- **Mean Absolute Error (MAE)**: $\frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$
  - Average prediction error

- **Mean Squared Error (MSE)**: $\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
  - Sensitive to outliers (penalizes larger errors)
  - Squared units relative to target variable

- **Root Mean Squared Error (RMSE)**: $\sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$
  - Sensitive to outliers (penalizes larger errors)
  - Same units as target variable

5. Loss Functions in Machine Learning

Loss Function = **Cost** Function = **Objective** Function

- Quantifies the error/discrepancy between the predicted and the actual (true) output.

- The purpose of training is to **minimize the loss**.
- Intuitively, $performance\ metric\ \rightarrow\ loss\ function$
- Most optimization algorithms (gradient-based methods) impose some restrictions on loss functions:
  - Differentiability (at least sub-differentiability)
  - Convexity (preferred)
  - Lack of large flat regions or plateaus

- **Accuracy**, **Precision**, **Recall**, and **F1-score**
  - Step functions (Piecewise Constant). Based on discrete predictions (TP, FP, FN, TN))
  - Not differentiable and where differentiable, flat.
  - → **Cannot be used**

- **Mean Absolute Error (MAE)**
  - Differentiable (except at zero error)
  - Constant gradient (± 1) → Prone to oscillations around the minimum.
  - **Can be used** for gradient-based optimization.

- **Mean Squared Error (MSE)** and **Root Mean Squared Error (RMSE)**
  - Differentiable
  - **Well-suited** for gradient-based optimization.

Classification tasks - Loss Functions

- **Cross-Entropy**
  - The average number of bits needed to identify/represent an output data y given an input data x and the knowledge provided by the model.
  - The average amount of **knowledge** provided by the model.

$$L = \frac{1}{n} \sum_{i=1}^{n} -\log(p(y = y_i | \mathbf{x}_i))$$

  - Measured in nats (base e) or bits (base 2)
  - Perfect Classifier: $L = 0$
  - Non-Informative Classifier: $L = \log(N) \quad (N$: number of classes)
  - Better than Non-Informative Classifier: $AUC < \log(N)$
  - Worse than Non-Informative Classifier: $AUC > \log(N)$

- **Hinge Loss**

- **Squared Hinge Loss**

- **Kullback-Leibler Divergence**

- **Focal Loss**