DISCLAIMER

These notes are supposed to explain some major topics covered in the lessons.

While they may not provide an exhaustive overview, they strive to complement the information presented in the slides with some explanations taken from additional readings in various textbooks.

If someone finds some errors or have any suggestion, it's possible to contact me on mmeschini001@gmail.com

MACHINE LEARNING

Why machine learning? Many interesting problems are too complex to admit an algorithmic solution or even a complete description. For these problems only data are available. Machine learning is about using data to solve problems.

Machine learning jargon

MODEL: a mathematical or computational representation of the real world

TRAINING DATA: the portion of the dataset used to train the machine learning model

TEST DATA: the portion of the dataset that is used to evaluate the performance of a trained machine learning model.

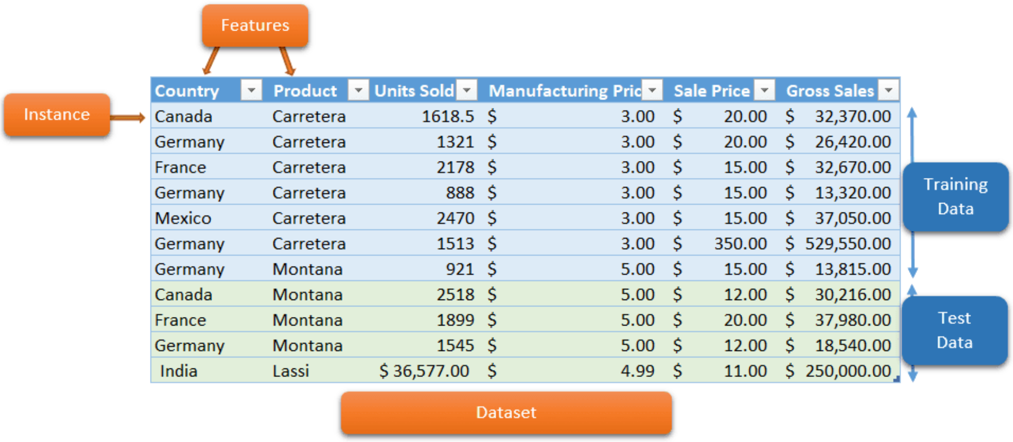
FEATURE/ATTRIBUTE/VARIABLES: an individual measurable property used to train the machine learning model

OBSERVATIONS/SAMPLES/ISTANCES: are individual data points that represent specific records. Typically they correspond to a single entry.

CLASS/LABELS: The output or target variable that a machine learning model is trying to predict.

OVERFITTING: When a machine learning model learns the training data too well, capturing noise and irrelevant details, which can lead to poor generalization to new data.

UNDERFITTING: The opposite of overfitting, where a model is too simple to capture the underlying patterns in the data, resulting in poor performance.



Country	Product	Units Sold	Manufacturing Pric	Sale Price	Gross Sales
Canada	Carretera	1618.5	$ 3.00	$ 20.00	$ 32,370.00
Germany	Carretera	1321	$ 3.00	$ 20.00	$ 26,420.00
France	Carretera	2178	$ 3.00	$ 15.00	$ 32,670.00
Germany	Carretera	888	$ 3.00	$ 15.00	$ 13,320.00
Mexico	Carretera	2470	$ 3.00	$ 15.00	$ 37,050.00
Germany	Carretera	1513	$ 3.00	$ 350.00	$ 529,550.00
Germany	Montana	921	$ 5.00	$ 15.00	$ 13,815.00
Canada	Montana	2518	$ 5.00	$ 12.00	$ 30,216.00
France	Montana	1899	$ 5.00	$ 20.00	$ 37,980.00
Germany	Montana	1545	$ 5.00	$ 12.00	$ 18,540.00
India	Lassi	$ 36,577.00	$ 4.99	$ 11.00	$ 250,000.00

There are three different types of machines learning:

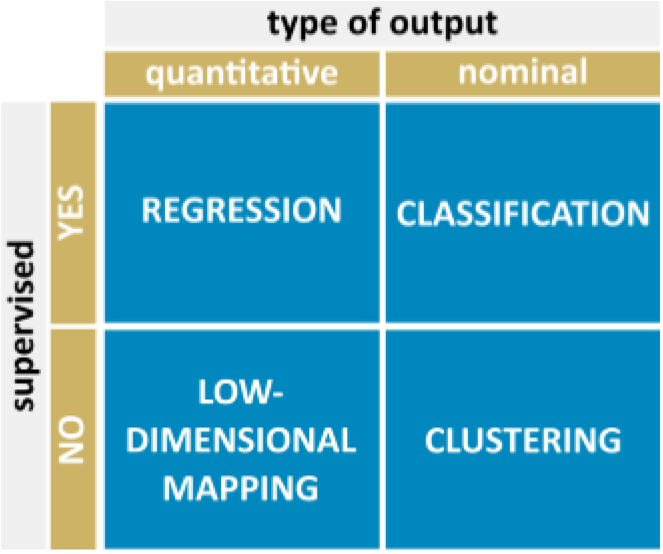
- SUPERVISED LEARNING: the algorithm "learns" from the training dataset by iteratively making predictions on the data and adjusting for the correct answer. While supervised learning models tend to be more accurate than unsupervised learning models, they require upfront human intervention to label the data appropriately.
- UNSUPERVISED LEARNING/CLUSTERING: Unsupervised learning models, in contrast, work on their own to discover the inherent structure of unlabelled data. Note that they still require some human intervention for validating output variables.

- REINFORCEMENT LEARNING algorithms work by training the agent through interaction with the environment. They learn to optimize their policy over time, improving their decision-making capabilities.

The types of quantity our model want to learn can be of two different types:

- Real values
- Categorical/nominal values: e.g. colour, name. (there isn't a natural ordering)

Depending on the model an the output variables we can distinguish:



		type of output	
		quantitative	nominal
supervised	YES	REGRESSION	CLASSIFICATION
	NO	LOW-DIMENSIONAL MAPPING	CLUSTERING

Possible scenarios

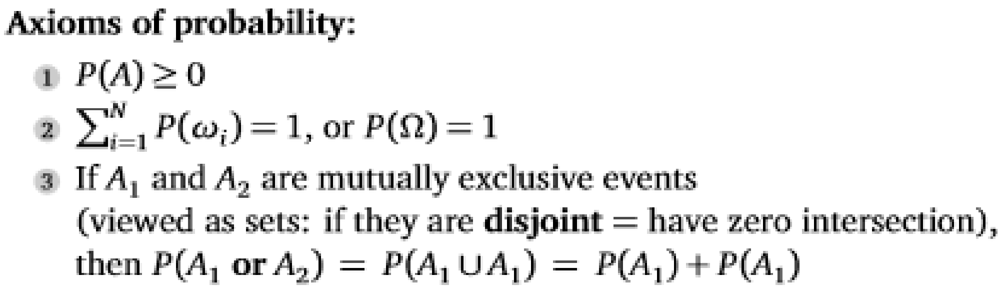
There are three possible scenarios:

1) Useful mostly for reasoning: we have a **learning machine** that is **fixed** and also the **data** are **fixed** (we have only the training set). The objective is to find the correct learners.
2) Not realizable: we **know the model that generates data** (the probabilities are known), so the objective is to find the optimal learner. Useful for theory
3) Our usual situation: **Data** are **stochastic**, the **learner** is **fixed** and chosen in advance. The objective is to find the best configuration of learning machine which is correct for any realization of the data

Probability

The probability quantifies the chance of an event or outcome occurring. It provides a way to express uncertainty to which something is likely to happen. Something that may or may not happen is called an **outcome (ω)**. All possible outcomes constitute the **sample space (Ω)**. An **event** refers to a specific outcome or a subsets of $\boldsymbol{\Omega}$**.**

We defined as P(A) the probability of event A.



Axioms of probability:

1. $P(A) \geq 0$
2. $\sum_{i=1}^{N} P(\omega_i) = 1$, or $P(\Omega) = 1$
3. If A_1 and A_2 are mutually exclusive events (viewed as sets: if they are **disjoint** = have zero intersection), then $P(A_1 \textbf{ or } A_2) = P(A_1 \cup A_1) = P(A_1) + P(A_1)$

A **random variable (X)** is a numerical variable that doesn't have a fixed value but changes according to a given probability law. The **Cumulative distribution function (CDF)** of a random variable X is the function given by:

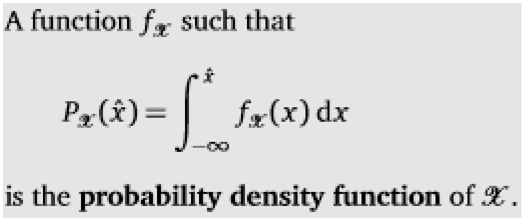
$$F_X(x) = \mathrm{P}(X \leq x) \quad \textbf{(Eq.1)}$$

Where the right-hand side represents the probability that the random variable X takes on a value less than or equal to x.

The **probability mass function** is a function that gives the probability that a **discrete** random variable is exactly equal to some value.

$$p_X(x) = P(X = x)$$

The **probability density function** is a function whose value at any given sample in the sample space can be interpreted as providing a relative likelihood that the value of the random variable would be equal to that sample.



A function $f_{\mathcal{X}}$ such that

$$P_{\mathcal{X}}(\hat{x}) = \int_{-\infty}^{\hat{x}} f_{\mathcal{X}}(x)\, \mathrm{d}x$$

is the **probability density function** of $\mathcal{X}$.

We define the **conditional probability P(E | F)** as the probability of an event E given the knowledge that another event F has occurred.

Two events are **independent** if the outcome of one event does not influence the outcome of the second event.

Bayesian decision theory

Bayesian decision theory is a fundamental statistical approach to the problem of pattern classification. This approach is based on quantifying the trade-offs between various classification decisions using probability and the costs that accompany such decisions. It makes the assumption that the decision problem is posed in probabilistic terms, and that **all the relevant probability values are known.**

We define as **state of nature (ω)** *(t in professor slide)*t he different possible states or scenarios that a system could be in. (e.g. salmon and sea bass are two different state of nature). We define **a priori probability** refers to the probability of an event occurring based on prior knowledge. It is represented by the likelihood in the occurrence of an event before any new evidence is taken into account. Then, is important to define the **class-conditional probability density function** expressed as **P(x|ω)** that represents the distribution of the random variable x depending on the state of nature.

The probability density of finding a pattern that is in category $\boldsymbol{\omega_j}$ **and** has a feature value x can be written in two ways: $\boldsymbol{P(\omega_j, x) = P(\omega_j|x) * P(x) = P\left(x\middle|\omega_j\right) * P(\omega_j)}$ rearranging these leads us to the **Bayes' formula:**

$$\boxed{P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_j)}{p(x)},} \qquad posterior = \frac{likelihood \times prior}{evidence}.$$

Bayes' formula shows that by observing the value of x we can convert the prior probability $\boldsymbol{P(\omega_j)}$ to the **a posteriori probability $\boldsymbol{P(\omega_j|x)}$,** that is the probability of the state of nature being ω_j given that the feature value x has been measured.

Oss: the evidence can be viewed as a scale factor that guarantees that the posterior probability sum to one.

In this context, is useful to introduce the **decision rule** that represents a criterion (a set of boundaries) used to minimize the cost of our decision. To evaluate each decision, we use the **expected loss,** also called **risk.** The **loss function (λ)** states exactly how costly each action is and is used to convert a probability determination into a decision. For a generic decision y_i the risk is:

$$R(y_i) = \sum_{j=1}^{c} \lambda(y_i, t_j) P(t_j)$$

Finally, the **conditional risk** represent the expected loss associated a decision under a specific state of nature.

$$R(y_i\,|\,\mathbf{x}) = \sum_{j=1}^{c} \lambda(y_i, t_j) P(t_j\,|\,\mathbf{x})$$

$R(y_i\,|\,\mathbf{x})$ is the **conditional risk** of decision y_i
when we have the **experimental observation** $\mathbf{x}$

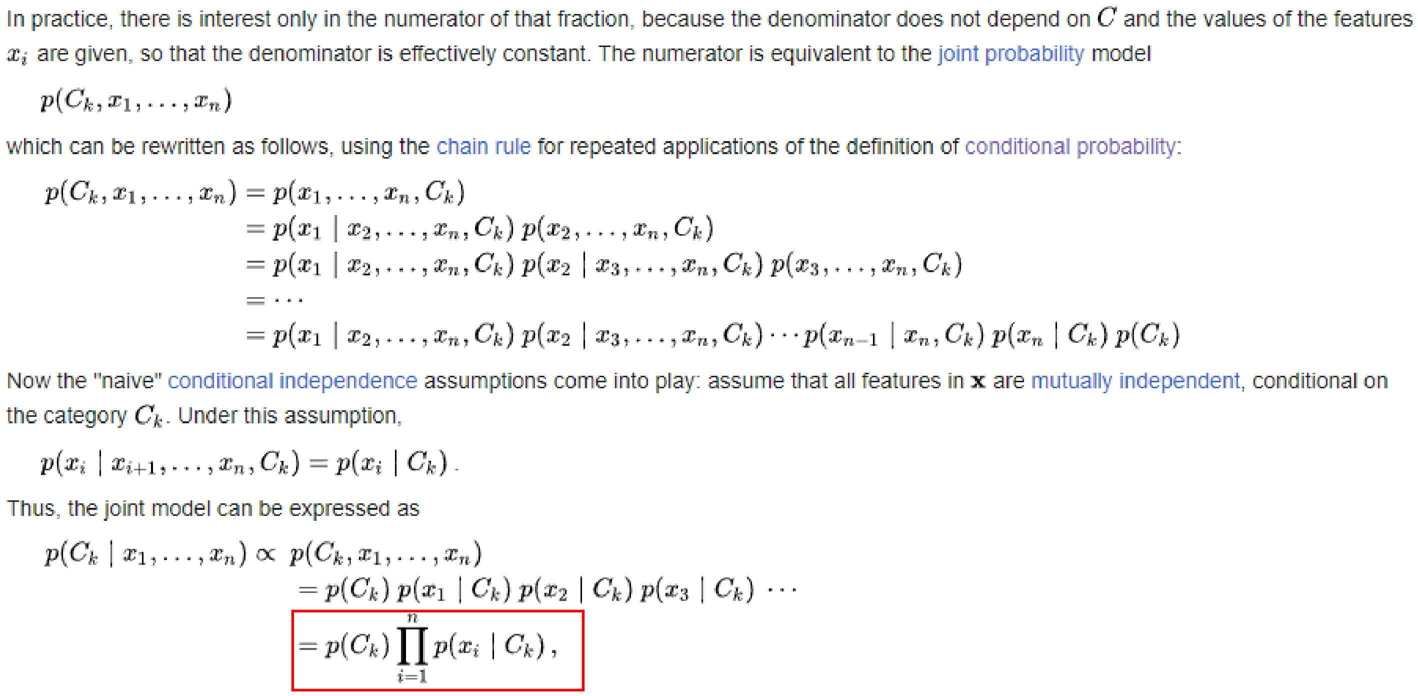
Classification is a decision problem where there is no decision to take and we want only to recognize the state of nature. In short, the model tries to predict the correct class of a given input data. A classifier is a rule y() that **receives an observation x** and **output a class y(x).**

Naive Bayes classifier

A Naive Bayes classifier is a simple and probabilistic machine learning algorithm that is often used for classification tasks. It is based on Bayes' theorem and makes a "naïve" assumption that features used to describe data are conditionally independent.

Naïve Bayes is a conditional probability model: it assigns probabilities $P(C_k|x_1, \dots, x_n)$ for each of the K possible classes C_k (= state of natures) given a problem instance to be classified represented by a vector $\boldsymbol{x} = (x_1, \dots, x_n)$ encoding some n features. From the Bayes' theorem

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$



In practice, there is interest only in the numerator of that fraction, because the denominator does not depend on C and the values of the features x_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model

$$p(C_k, x_1, \ldots, x_n)$$

which can be rewritten as follows, using the chain rule for repeated applications of the definition of conditional probability:

$$\begin{aligned} p(C_k, x_1, \ldots, x_n) &= p(x_1, \ldots, x_n, C_k) \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k)\ p(x_2, \ldots, x_n, C_k) \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k)\ p(x_2 \mid x_3, \ldots, x_n, C_k)\ p(x_3, \ldots, x_n, C_k) \\ &= \cdots \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k)\ p(x_2 \mid x_3, \ldots, x_n, C_k) \cdots p(x_{n-1} \mid x_n, C_k)\ p(x_n \mid C_k)\ p(C_k) \end{aligned}$$

Now the "naive" conditional independence assumptions come into play: assume that all features in $\mathbf{x}$ are mutually independent, conditional on the category C_k. Under this assumption,

$$p(x_i \mid x_{i+1}, \ldots, x_n, C_k) = p(x_i \mid C_k)\,.$$

Thus, the joint model can be expressed as

$$\begin{aligned} p(C_k \mid x_1, \ldots, x_n) &\propto\ p(C_k, x_1, \ldots, x_n) \\ &= p(C_k)\ p(x_1 \mid C_k)\ p(x_2 \mid C_k)\ p(x_3 \mid C_k)\ \cdots \\ &\boxed{= p(C_k) \prod_{i=1}^n p(x_i \mid C_k)\,,} \end{aligned}$$

The decision rule for minimizing the probability of error is:

$$\text{Decide } \omega_1 \ if\ P(\omega_1|\ x) > P(\ \omega_2|x); otherwise\ decide\ \omega_2$$

The loss function for this classifier il **zero-one:** $\lambda(y\,|\,t) = \begin{cases} 0 & \text{if } y = t \\ 1 & \text{if } y \neq t \end{cases}$

So, all types of errors have the same cost (=1) and correct classifications don't have a cost.

Linear Regression

When only data are available is possible to find the correct output using a regression model. **Regression** means approximating a functional dependency based on measured data.

It is a supervised problem given that we have observations and target

$$\underset{\text{Observations}}{\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \vdots \\ \mathbf{x}_N \end{pmatrix}} \quad \underset{\text{Target}}{\begin{pmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \mathbf{t}_3 \\ \vdots \\ \mathbf{t}_N \end{pmatrix}}$$

We are looking for a linear model y(x) that predicts t given x.
Since the model is linear the form will be: $y(x) = w_0 + wx$

From the training set is possible to estimate the parameters w. Difficulty is possible to find values that are good for all points, so the solution is to find two values that minimize the average error. The most popular estimation method is the **square error loss**, in which we pick the coefficients w to minimize the following loss function:

$$\lambda_{\mathrm{SE}}(y,t) = (y-t)^2 \xrightarrow{\text{So our objective function will be}} J_{\mathrm{MSE}} = \frac{1}{N}\sum_{l=1}^{N}(y_l - t_l)^2$$

- This function when building a model (=training) is a function of the **parameters** of the model and the data are fixed.
- When using a model (=inference) is function of the **data**, while the parameters are now fixed.

The least squares solution of the one-dimensional linear regression problem (for the model $y(x) = wx$) is:

$$w = \frac{\sum_{l=1}^{N} x_l t_l}{\sum_{l=1}^{N} x_l^2}$$

While for the model $y(x) = w_0 + w_1 x$ where is also present the offset is:

$$\bar{x} = \frac{1}{N}\sum_{l=1}^{N} x_l \qquad \bar{t} = \frac{1}{N}\sum_{l=1}^{N} t_l$$

Note:

w_1 is called: slope, gain

w_0 is called intercept, offset, bias

$$w_1 = \frac{\sum_{l=1}^{N}(x_l - \bar{x})(t_l - \bar{t})}{\sum_{l=1}^{N}(x_l - \bar{x})^2}$$

$$w_0 = \bar{t} - w_1\bar{x}$$

Multidimensional linear regression problem

The data is now composed of d-dimensional vectors $\mathbf{x}_1 = [\, x_{1,1}, x_{1,2}, \ldots, x_{1,d}]$ and we can organize them into a N x d matrix. Now we have also d parameters:

$$X = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \vdots \\ \mathbf{x}_N \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ x_{3,1} & x_{3,2} & \cdots & x_{3,d} \\ & & \vdots & \\ x_{N,1} & x_{N,2} & \cdots & x_{N,d} \end{pmatrix} \qquad \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_d \end{pmatrix}$$

The linear model takes the d inputs of each observation and combines them by using the d parameters to produce one output:

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_N \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ x_{3,1} & x_{3,2} & \cdots & x_{3,d} \\ & & \vdots & \\ x_{N,1} & x_{N,2} & \cdots & x_{N,d} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_d \end{pmatrix} = \begin{pmatrix} x_{1,1}w_1 + x_{1,2}w_2 + & \cdots & +x_{1,d}w_d \\ x_{2,1}w_1 + x_{2,2}w_2 + & \cdots & +x_{2,d}w_d \\ x_{3,1}w_1 + x_{3,2}w_2 + & \cdots & +x_{3,d}w_d \\ & & \vdots \\ x_{N,1}w_1 + x_{N,2}w_2 + & \cdots & +x_{N,d}w_d \end{pmatrix} = X\mathbf{w}$$

The affine case let us incorporate the additive parameter w_0 by adding one constant column to the data matrix

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_N \end{pmatrix} = \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ 1 & x_{3,1} & x_{3,2} & \cdots & x_{3,d} \\ & & & \vdots & \\ 1 & x_{N,1} & x_{N,2} & \cdots & x_{N,d} \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_d \end{pmatrix} = \begin{pmatrix} w_0 + x_{1,1}w_1 + x_{1,2}w_2 + & \cdots & +x_{1,d}w_d \\ w_0 + x_{2,1}w_1 + x_{2,2}w_2 + & \cdots & +x_{2,d}w_d \\ w_0 + x_{3,1}w_1 + x_{3,2}w_2 + & \cdots & +x_{3,d}w_d \\ & & \vdots \\ w_0 + x_{N,1}w_1 + x_{N,2}w_2 + & \cdots & +x_{N,d}w_d \end{pmatrix} = X\mathbf{w}$$

Our final goal is to make this model's prediction $\boldsymbol{y}$ as similar as possible to the measured outputs for each observation

$$\mathbf{t} = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \\ \vdots \\ t_N \end{pmatrix}$$

The solution in this case is: $\mathbf{w} = (X^{\intercal}X)^{-1}X^{\intercal}\mathbf{t}$