Fundaments of Machine learning for and with engineering applications

Reidar B. Bratvold, Enrico Riccardi[1]

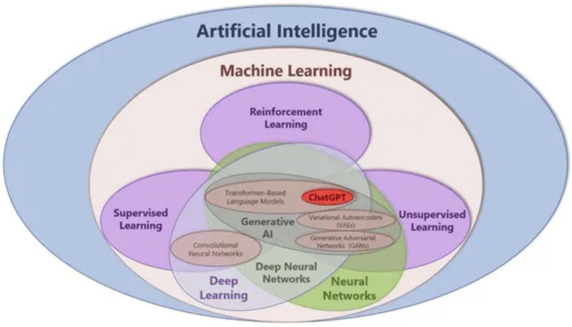
Department of Energy Resources, University of Stavanger (UiS).[1]

Feb 6, 2024

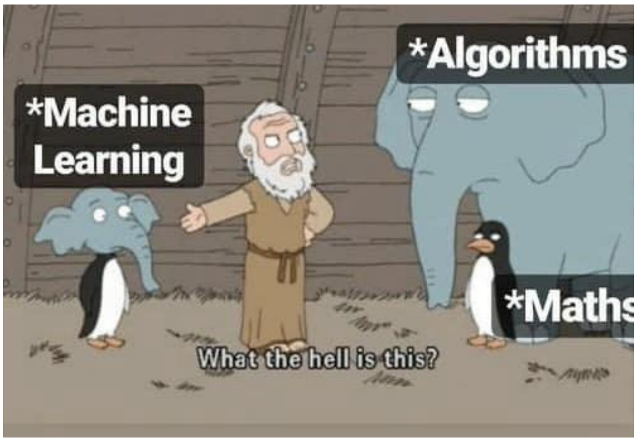
University of Stavanger

Statistics, Machine learning or Artificial intelligence?

What is the main difference between the three fields?



How Machine Learning Started?



Statistics

Let's start from the definition

- Statistics (origin "description of a state/country") is the discipline that concerns the collection, organization, analysis, interpretation, and presentation of data.
- It is conventional to begin with a statistical population or a statistical model to be studied. Populations can be diverse groups of people or objects such as "all people living in a country" or "every atom composing a crystal".
- Statistics deals with every aspect of data, including the planning of data collection in terms of the design of surveys and experiments.[Wikipedia]

Machine learning

Definitions:

- Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. [IBM]
- Machine learning (ML) is a field of study in artificial intelligence concerned with the development and study of statistical algorithms that can learn from data and generalize to unseen data, and thus perform tasks without explicit instructions. [WIKI]
- Machine learning is a subfield of artificial intelligence that uses algorithms trained on data sets to create models that enable machines to perform tasks that would otherwise only be possible for humans, such as categorizing images, analyzing data, or predicting price fluctuations. [Coursera]

Machine learning

One technical definition

Machine learning is a set of computer based statistical approaches that aim to minimise the loss function to maximise inference accuracy. [Enrico, 5.2.2024]

The loss function is the actual engine in machine learning.

Loss function

It quantifies the difference between the predicted outputs of a machine learning algorithm and the actual target values.

Artificial intelligences

And more definitions:

- Artificial intelligence is the intelligence of machines or software, as opposed to the intelligence of humans or other animals. It is a field of study in computer science that develops and studies intelligent machines. [WIKI]
- Artificial intelligence (AI) is the theory and development of computer systems capable of performing tasks that historically required human intelligence, such as recognizing speech, making decisions, and identifying patterns [Coursera]
- It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable. [IBM]

Data properties

- All starts from data: what are data-properties?

- Are there such things as good data and bad data?

Main lesson (Exam question)

- Data DO NOT always have value.

- TRASH in TRASH out

MetaData properties

Data without metadata are just numbers (i.e. if they are integers, they are still good to play lottery)

Metadata can be pretty much anything. Depending on the application, we can distinguish between:

1. Descriptive : used for discovery and identification. It includes elements such as title, abstract, author, and keywords.
2. Structural : describe how compound objects are put together. It describes the types, versions, relationships, and other characteristics of digital materials.
3. Administrative : to help manage a resource, like resource type, permissions, and when and how it was created.
4. Reference : to indicate the information about the contents and quality of statistical data.
5. Statistical : (or process data), may describe processes that collect, process, or produce statistical data.
6. Legal : creator, copyright, licensing.

MetaData aim

These characteristics shall all be considered when constructing data repositories.

They are a must for:

- Code repositories
- Data repositories

Please click on the link and explore. Those are just some of the open (scientific) repositories. The aim/hope is to allow people to extract information. How to make value from that information... is another story.

MetaData for sharing and re-use

More considerations:

- Metadata is more and more important in a digital open world.
- Researchers and automatic algorithms would benefit from importing data directly.
- FAIR research is an important part of Open Science revolution (Findable, Accessible, interoperable, Reusable)
- New applications, business, discoveries can be thus enabled.
- ChatGPT, Bard, Gemini, and all the LLMs are functional only thanks to this!

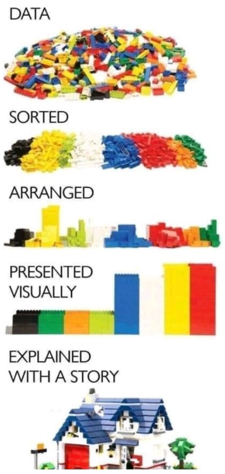
Super controversial

- Who would be responsible for them then?
- What is the advantage for who releases the data?
- Who gets the money for what?
- Copyright for data and/or for data processing?

Good examples

- Norwegian offshore directorate
- Norway Statistics
- World statistics
- Code repositories
- Data repositories

Data



Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Comparability

Same meaning **represenations** for different objects (inputs).

Discussion point!

How do we compare two wind turbines accounting for the 5 variables previously introduced?

Data properties

- All starts from data: what are data-properties?
- Are there such things as good data and bad data?

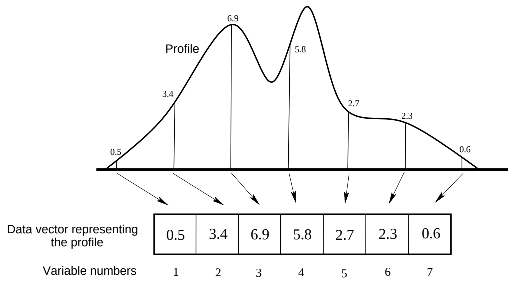
Life lesson (or exam question, same thing ;))

- Data DO NOT always have value.

- TRASH in TRASH out

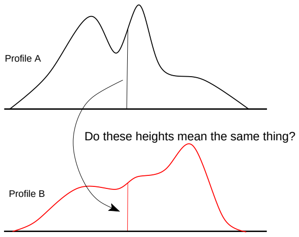
Sampling point representation (SPR)

- An intuitive way to represent curves and spectra is the **sampling point representation**.
- We sample at regular intervals where each sample point is represented by a variable



Sampling point representation (SPR)

- SPR is useful until point *i* in a curve has the same meaning of the point *i* in another curve.



- Which parts of the profiles or shapes are comparable, i.e. have the same meaning?

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Spatial and Temporal Data

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Data collection

On the data sources side

- Confidence intervals
- Relevance
- Significance
- Correlation
- Causation
- Data Filters
- Biases identification

Actual data

The data matrix is an extremely common data structure.

$$X = \begin{bmatrix} 95 & 89 & 82 \\ 23 & 76 & 44 \\ 61 & 46 & 62 \\ 49 & 2 & 79 \end{bmatrix}$$

In python these can be saved as

- lists (vanilla python)
- numpy.arrays
- pandas dataframes

Nomenclature

There are different conventions. Commonly we will construct data matrix such that:

- Rows are called instances, objects or samples.
- Columns are called features, variables.

One can think of each row to be an experiment, and the rows its properties. Each row (experiment, object, sample, ...) is thus a list of values, one for property.

Note

Mathematically speaking, this is just a notation. As long as one keeps track and is consistent, columns can be used as rows and vice versa.

A quick example

Environmental measurements of rivers. The features (properties) can be:

- pH
- Temperature
- Concentration of pollutants
- Flow rate
- water speed

The experiments/observations/sample can be:

- Po
- Danube
- Rio delle Amazzoni
- Sjoa
- Atna

Hard and soft modelling

Models allow us to predict 'the future', or describe the past and present (what is the present...?)

> Last life lesson for today
>
> Models are always wrong, but some are useful. (George Box)

Three main families:

1. Hard models (physics)
2. Soft models (statistic)
3. Machine learning

Hard modelling

- Based on an accurate physical description of the system and mathematical modelling (e.g. differential equations).

Hard models are often deterministic.

- Hard-modelling methods usually use optimization methods to find out the best values for the parameters of the model.
- Hard-modelling is preferable in laboratory experiments, where all the variables are controlled and the physicochemical nature of the dynamic model is known and can be fully described using a known mathematical model.
- Hard-modelling, if successful, usually gives better understanding of a system and better extrapolations. Wrong assumptions often leads to nonsense results.

Soft modelling

Characteristics:

- Soft-modelling describes systems without the need of an *a priori* physical or (bio)chemical model postulation. They are **data driven** models.
- Soft models are much easier to make than hard models.
- Soft modelling can be used to understand complex relationships.
- Soft modelling needs (much) more data than hard-modelling.
- Soft models have a poor extrapolating capabilities (compared with hard-modelling)

How to create hard models?

After understanding the problem to be solved we need to:

1. Link mathematics to physics.
2. Define boundary conditions and constitutive equations.
3. Make tons of assumptions.
4. Solve the constitutive equation in space and time.
5. CHeck solution stability and sensitivity analysis.
6. A long set of judicious approximations have to be taken.
7. It is hard (but we are engineers!).
8. Get quite some money for the awesome job.

How to create soft models?

After understanding the general problem to be solved we need to:

- Determine a suitable **numerical description** .
- Choose a suitable **model** to which parameters are fitted.
- Train, test, validate the model.
- Perform **data analysis** with chosen method(s).
- Link predictions with expectations.

Data collection

> On the data sources side

- Confidence intervals
- Relevance
- Significance
- Correlation
- Causation
- Data Filters
- Biases identification

Models and Methods

> On the modelling side

- Regression
- Clustering
- Principal components
- Decision tree (random forests)
- Neural network
- Performance metrics

Spatial and Temporal Data

Remember the definition?

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Spatial and Temporal Modelling

It is a branch of statistical analysis and model that uses spatial and time dependent data.

- Only a subset of statistical models can be fed with time dependent data (most standard statistical method assume independent, identically distributed, data)
- Spatial and time related data come at a different range of scales. Data collection can be dependent of time and space, resulting in different representativity of a sample.

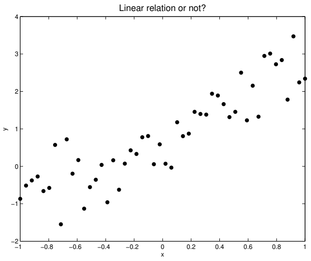
Fields of application

Tipical time and spatial data are:

- Spatial estimation of energy and mineral resources
- Weather modelling: from aviation to agriculture
- Maintenance forecasting
- Commodity, currency, stock and financial markets
- Market analysis
- Risk analysis
- ... and much much more!

Finding a suitable model

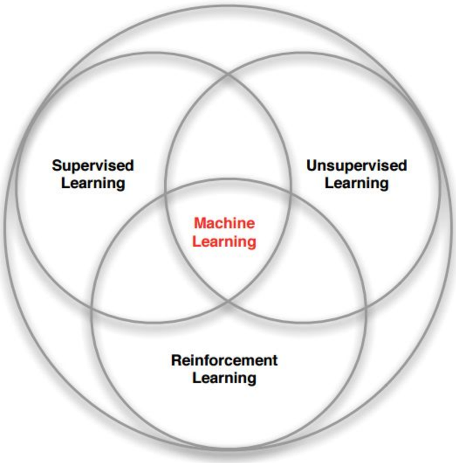
Soft modeling is in most cases based on **multivariate statistical methods**. Many of these methods may be viewed as sophisticated ways of performing curve fitting to data.



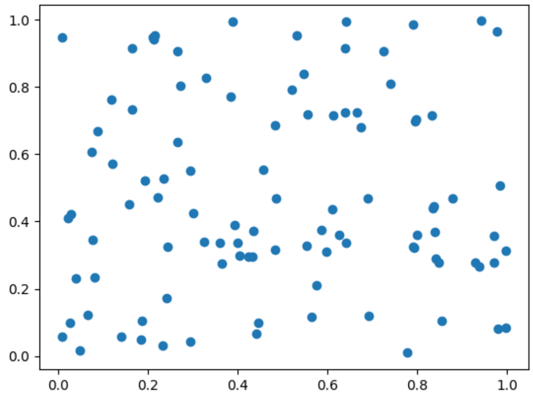
What would be the best model?

- Straight?: $y(x) = ax + b$
- Parabolic?: $y(x) = ax^2 + bx + c$
- Trigonometric?: $y(x) = asin(x) + bcos(x)$

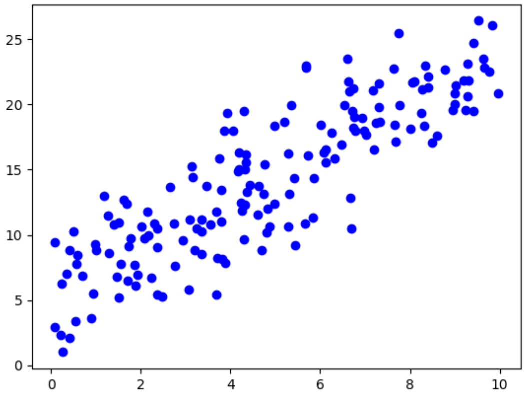
Families of Machine learning



What can we do with that?



What about in this case?



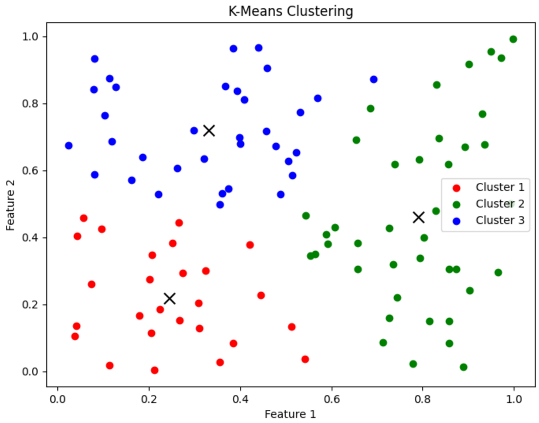
Python Source code 1

```
import numpy as np
import matplotlib.pyplot as plt

# Generate some sample data
data = np.random.rand(100, 2)  # 100 data points with 2 features

plt.scatter(data[:, 0], data[:, 1])
plt.show()
```

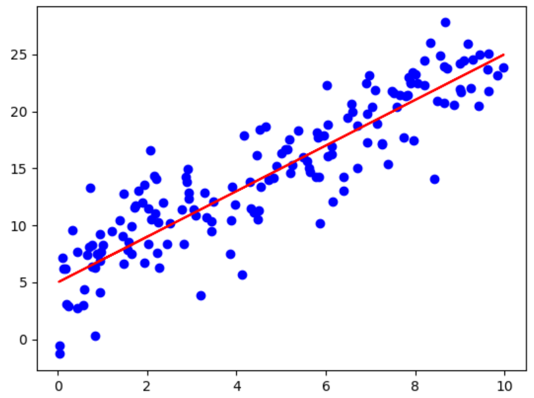
Python Source code 2

```
import numpy as np
import matplotlib.pyplot as plt

def generate_linear_data(n_random_points, noise=16):
    x = np.random.rand(n_random_points) * 10

    # Make 'perfect' data
    true_slope,  true_intercept = 2, 5
    y = true_slope * x + true_intercept

    # Add noise
    y += np.random.randn(n_random_points)*noise

    return x, y, true_slope, true_intercept

# Use the function to generate data
x, y, true_slope, true_intercept = generate_linear_data(
        n_random_points=166,
        noise=3)

# Plot all
plt.scatter(x, y, color='blue', label='Data Points')
plt.show()
```

Unsupervised learning



Supervised learning



The data decides

This is why we focus so much on the data type.

The data properties dictate what statistical model can be adopted.

An statistical model has leverages our understanding of the data structure to improve its **predictions** (inference).

The numerical recipe that we used to generate the data is defined the **truth**

Psychology or data science?

Most Machine learning tools are aimed to find the truth. In most cases, we are happy to not find lies.

Unsupervised learning

Unsupervised learning, a term that resonates with the autonomy of machine intelligence, operates on the principle of identifying patterns and structures in datasets without labelled responses.

This branch of machine learning is distinguished by its lack of explicit guidance, where algorithms are tasked with uncovering hidden structures from unlabeled data.

The most common clustering strategies are :

- filtering
- clustering
- dimensionality reduction
- association learning

Application of unsupervised learning

It is a bit of a holy grail: a computer that finds patterns without guidance. (Yes, it doesn't work, most of the time)

Still, it has been shown efficient for:

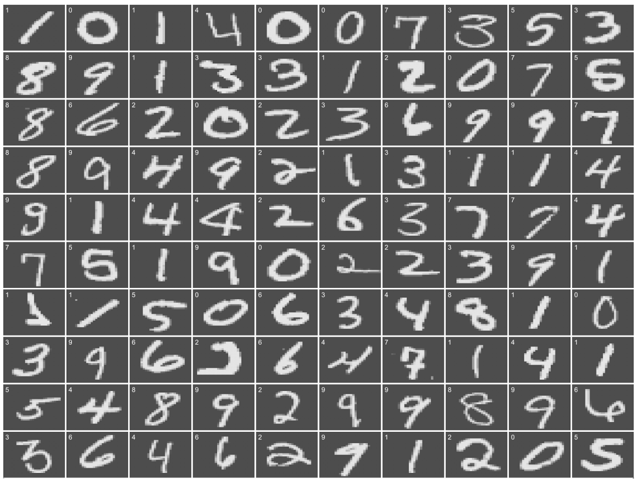
- Computer vision
- Anomaly detection
- Exploratory data analysis

Main challenge

The right result is quite undefined, Uncertain goal.

We will demonstrate it with a famous problem.

Uncertain goal



Wak supervised learning

A less popular type of machine learning problem is when labels are assigned to groups of instances.

The group of instances is called **bag**.

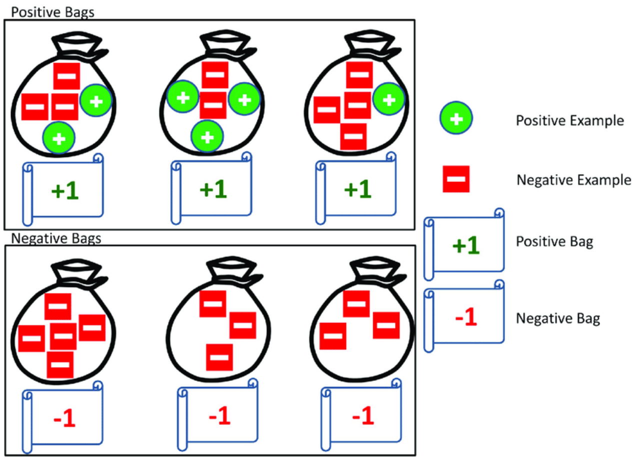
The question is, what is the level of a previously unforeseen bag?

This data structure and question type request a hybrid treatment between supervised and supervised learning.

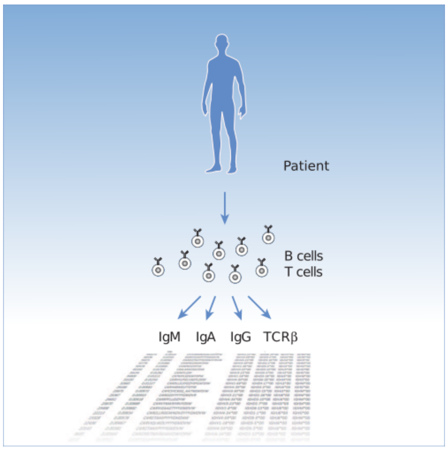
Multiple instance learing

Multiple instances are needed to learn (quite clear name)

Weak Supervised learning



Weak Supervised learning



Reinforcement learning

Finally, there is a further approach.

Reinforcement learning (RL)

It aims to train an intelligent agent to take actions in a dynamic environment in order to maximise the cumulative reward.

It learns from outcomes and decides which action to take next. After each action, the algorithm receives feedback that helps it determine whether the choice it made was correct, neutral or incorrect.

It is a self-teaching system that essentially learns by trial and error.

It is a dependable tool for automated decision making.

Generative AI

A generative AI model is a type of artificial intelligence that is designed to generate new content, based on the data it has been trained on.

It started in 1932, with the **mechanical brain** by Georges Artsrouni that was suppoused to translate automatically between languages,

Here a nice recaps of Generative AI and its storyline

Generative AI

Key characteristics of generative AI models include:

1. Learning from Data: They are trained on large datasets, enabling them to learn patterns, styles, or features inherent in the data.
2. Generating New Content: Generative models can create new data instances. For example, a model trained on a dataset of paintings can generate new images in the style of those paintings.

Trained generative models are thus able to input information at a low resolution/dimension and give output with a much greater dimensionality.

Applications

Here a list of possible applications:

- Images/video: Image generation, Super-resolution, Deep fakes.
- Music: noise filter, voice and music generation, voice deep fake.
- Text(LLM): chatGPT, bard, Gemini, etc.
- Chemistry: DeepMind (Alphafold).
- Coding (co-pilot)
- Speech
- Attacks and Hacking (Security testing)
- Generating training sets
- And many more

Science fiction?

This is scary:

1. Virtual best friends
2. Medical images to show diseases consequences
3. Synthetic data for digital twins
4. Preemptive suggestions (e.g. driving)
5. Matrix

Problems (currently)

New possibilities do not come with side effects.

1. Lack of transparency: how the output is generated, and why?
2. Accuracy: a lot of hallucinations
3. Bias: human biases are kept, supported and eventually increased
4. Intellectual properties (IP): who owns what is produced=
5. Cybersecutiry and frauds: mass cyber attacks can be created
6. Sustainability: massive quantity of electricity is used
7. Responsibility (who to blame?): Will AI get citizenship everywhere?

Where generative AI is ?

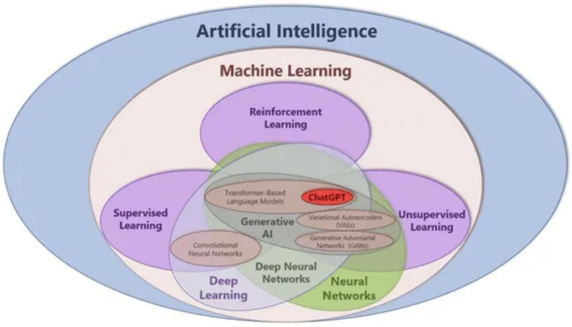


Image: https://iot-analytics.com

Structure of generative AI

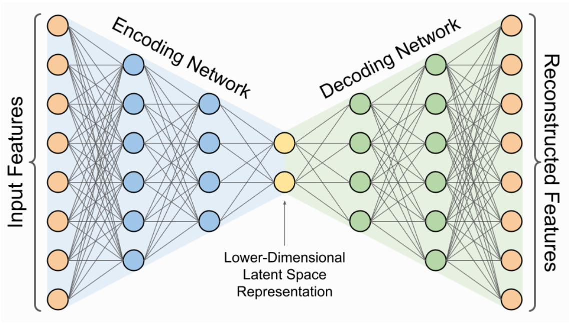


Image: https://www.rapidops.com

A new field?

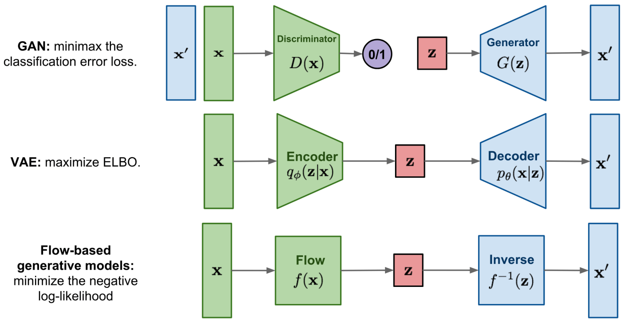
Generative AI is actually a new evolution.

It is based on Neural Network, and in comprises a set of advanced tools (numerical recepites):

1. Generative Adversarial Networks
2. Generative Pre-trained Transformers
3. Variational Autoencoders
4. Conditional Variational Autoencoders
5. Autoencoders

Types of generative AI

It is quite an advanced technique


GAN: minimax the classification error loss.
$\mathbf{x}'$
$\mathbf{x}$
Discriminator $D(\mathbf{x})$
0/1
$\mathbf{z}$
Generator $G(\mathbf{z})$
$\mathbf{x}'$
VAE: maximize ELBO.
$\mathbf{x}$
Encoder $q_\phi(\mathbf{z}|\mathbf{x})$
$\mathbf{z}$
Decoder $p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
Flow-based generative models: minimize the negative log-likelihood
$\mathbf{x}$
Flow $f(\mathbf{x})$
$\mathbf{z}$
Inverse $f^{-1}(\mathbf{z})$
$\mathbf{x}'$

Source: Lilian Weng

A **linear model**

Considering a univariate case, we have:

$$q = f(x)$$

which relates the **independent variable** x to the **true dependent variable** q.

!vspace1em

Assuming a linear model

$$q = \beta_0 + \beta_1 x$$

where β_i are the arbitrary selected coefficients.

Model set-up

For a given x we do not know the true response q, only the measurementa y_i for experiment i.

We have that:

$$y_i = q_i + \epsilon_i$$

NOTE: do not proceed if you do not fully understand this equation.

which is

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Discussion point

Does ϵ_i matter? And why so?

Estimated model paramters

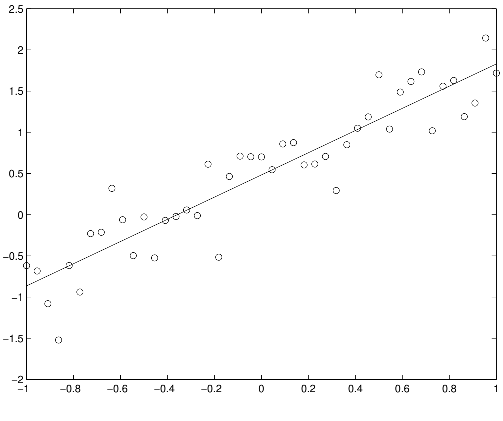
The model parameters β_0, β_1 are unknown, but they can be **estimated**. To distinguish estimates from true model parameters we call them b_0, b_1. These estimates are calculated such that the model

$$\hat{y} = b_0 + b_1 x$$

fits the n different experimental observations as well as possible.

Linear model example

We would like to find the TRUTH (What it it?)



Python Example

```
import numpy as np
import matplotlib.pyplot as plt

def generate_linear_data(n_random_points, noise=16):
    x = np.random.rand(n_random_points) * 10

    # Make 'perfect' data
    true_slope,  true_intercept = 2, 5
    y = true_slope * x + true_intercept

    # Add noise
    y += np.random.randn(n_random_points)*noise

    return x, y, true_slope, true_intercept

# Use the function to generate data
x, y, true_slope, true_intercept = generate_linear_data(
        n_random_points=166,
        noise=3)

# Plot all
plt.plot(x, true_slope*x + true_intercept,
         color='red', label='Truth Line')
plt.scatter(x, y, color='blue', label='Data Points')
plt.show()
```

Linear model(s)

Does a linear model mean only straight lines (or hyperplanes in general)?

That answer to this is *no.* different. In general for a model f to be defined linear, it has to be linear with respect to the unknown parameters $\beta_0, \cdots, \beta_n$. The general linear model is

$$q = \beta_0 + \beta_1 f_1(x_1) + \beta_2 f_2(x_2) + \cdots + \beta_n f_n(x_n)$$

where $f_i(x_i)$ may be non-linear functions. It is the **form** of the equation which makes it linear, i.e. that $f_i(x_i)$ does not depend on the parameters β_i.

Linear model(s)

Consider the following model example - is it linear?

$$q = \beta_0 + \beta_1 x_1^2 + \beta_2 x_2^{-1} + \beta_3 \mathrm{log} x_3$$

The answer is *yes* because by simple substitution it is possible to convert this formula into a linear form.

With $h_1 = x_1^2$, $h_2 = x_2^{-1}$ and $h_3 = \mathrm{log} x_3$, then we can formulate the new model:

$$q = \beta_0 + \beta_1 h_1 + \beta_2 h_2 + \beta_3 h_3$$

which is in the standard linear form.

Curvilinear models

A special class of linear models which we will investigate later are those which are expressed in terms of *polynomials* (here in only 1D):

$$q = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_m x^m = \sum_{i=0}^{n} \beta_i x^i$$

For n-D case there are a wide range of interaction terms and combinations, that can still be converted to standard linear form.

Such models are sometimes referred to as **curvilinear** instead of non-linear

Nonlinear models

But there are many other models that cannot be substituted to such a form. For instance:

$$q = \beta_0 + \log(x - \beta_1)$$

No substitution can transform this equation to the linear form.

That is the case for all the model that:

$$q \sim f(x, \beta)$$

Another example is:

$$f(x, \beta) = \frac{x\beta}{x + \beta}$$

Estimation of linear regression parameters

For the 1-dimensional problem, we have

$$\hat{y} = b_0 + b_1 x$$

where $\hat{y}$ is the estimated y-value from the approximate model that has been generated from a set of measurements (x_i, y_i). We aim to find the b_i parameters such that the regression line fits the observed data as well as possible.

This means we want to minimise the residuals

$$e_i = y_i - \hat{y}_i$$

Estimation of linear regression parameters

- Cannot sum e_i values since they might be positive and negative and thus cancel
- Could use e.g. $\sum_{i=1}^{n} |e_i|$, but is mathematically more difficult to handle
- Residual"smallness" measured by $\sum_{i=1}^{n} e_i^2$.

Thus, we find the linear regression coefficients by *minimising*

$$R = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

How?

Regression!

Let's recaps

Before to continue, let's make sure to have all the main elements clear:

1. Splitting the data in dependent and independent variables
2. Assumption of a linear model between them
3. Recognise the difference between the truth and the estimation
4. Aiming to *minimize* the residuals

Discussion

What happen when the sun of residual is 0 ?

What happens where the data is heavily correlated?

Regression

To minimize the sum of the square residuals, we can try to solve the following equations:

$$\begin{aligned}\frac{\partial R}{\partial b_0} &= 0\\ \frac{\partial R}{\partial b_1} &= 0\end{aligned}$$

where:

$$R = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad =$$

$$\sum_{i=1}^{n}(y_i - (b_0 + b_1 x_i))^2 = \sum_{i=1}^{n} u_i^2$$

Regression

Skipping the math (but you are more than welcome to try), here are the results:

$$\begin{aligned}b_0 &= \bar{y} - b_1\bar{x}\\ b_1 &= \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}\end{aligned}$$

Straightforward derivation becomes very cumbersome for multiple variables. Thus, a different approach must be used. Yet, it is important to understand that there is an analytical solution (even if not all the time).

Many variable equation

The general equation would look like:

$$\hat{y} = b_0 + b_1x_1 + b_2x_2 + \cdots + b_nx_n = \sum_{j=0}^{m} x_{ij}b_j$$

where we will have to solve all the $n+1$ equations (called the **normal equations**) of the form:

$$\frac{\partial R}{\partial b_j} = 0 \quad \forall j \in [0, n]$$

Is there a way for us to simplify this?

We can use vector and matrix algebra.

Regression via Matrix operation

Remember that

$$R = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

then we can define the vector **e**:

$$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}}$$

thus

$$\boldsymbol{e}^T = [(y_1 - \hat{y}_1)\ (y_2 - \hat{y}_2)\cdots\ (y_N - \hat{y}_N)]$$

and can then write

$$R = \boldsymbol{e}^T\boldsymbol{e}$$

Regression via Matrix operation

Remember that

$$R = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

then we can define the vector **e**:

$$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}}$$

thus

$$\boldsymbol{e}^T = [(y_1 - \hat{y}_1)\ (y_2 - \hat{y}_2)\ \cdots\ (y_N - \hat{y}_N)]$$

and can then write

$$R = \boldsymbol{e}^T\boldsymbol{e}$$

From the following equation:

$$\hat{y}_i = b_0 + \sum_{j=1}^{m} x_{ij}b_j = \sum_{j=0}^{m} x_{ij}b_j$$

where $x_{i0} = 1$
we make the matrix equation:

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{b}$$

where the first column in **X** consists of ones only.

Residual

$$\begin{aligned}R &= \boldsymbol{e}^T\boldsymbol{e}\\ &= (\boldsymbol{y} - \hat{\boldsymbol{y}})^T(\boldsymbol{y} - \hat{\boldsymbol{y}})\\ &= (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b})\\ &= (\boldsymbol{y}^T - \boldsymbol{b}^T\boldsymbol{X}^T)(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b})\\ &= \boldsymbol{y}^T\boldsymbol{y} - \boldsymbol{y}^T\boldsymbol{X}\boldsymbol{b} - \boldsymbol{b}^T\boldsymbol{X}^T\boldsymbol{y}\\ &+ \boldsymbol{b}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b}\end{aligned}$$

All the parts of this equation are scalar values. This means e.g. that

$$\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{b} = \boldsymbol{b}^T\boldsymbol{X}^T\boldsymbol{y}$$

This gives

$$R = \boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{b} + \boldsymbol{b}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b}$$

Residual

But how can we now compute $\frac{\partial R}{\partial b_j}$ more efficiently in matrix form?

Vector differentiation ! Let

$$y = \boldsymbol{a}^T\boldsymbol{x} = a_1x_1 + \cdots + a_nx_n$$

If

$$\frac{\partial y}{\partial \boldsymbol{x}} = \begin{bmatrix}\frac{\partial y}{\partial x_1}\\ \frac{\partial y}{\partial x_2}\\ \vdots\\ \frac{\partial y}{\partial x_n}\end{bmatrix} = \begin{bmatrix}a_1\\ \vdots\\ a_n\end{bmatrix} = \boldsymbol{a}$$

and $y = \boldsymbol{x}^T\boldsymbol{a}$, then:

$$\frac{\partial y}{\partial \boldsymbol{x}} = \boldsymbol{a}$$

General solution

In general, when $y = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$, then

$$\frac{\partial y}{\partial \boldsymbol{x}} = 2\boldsymbol{A}\boldsymbol{x}$$

if **A** is symmetric
(check *Matrix calculus* for more properties)

We can use this to compute

$$\frac{\partial R}{\partial \boldsymbol{b}}$$

General solution

We have from above:

$$\begin{aligned} R &= \boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{b} \\ &+ \boldsymbol{b}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b} \end{aligned}$$

Vector differentiation gives

$$\frac{\partial R}{\partial \boldsymbol{b}} = 0 - 2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b} = 0$$

Solving this for $\boldsymbol{b}$ we get:

$$\begin{aligned} \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b} &= \boldsymbol{X}^T\boldsymbol{y} \\ (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b} &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \\ \boldsymbol{b} &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \end{aligned}$$

Multiple linear regression

Previous equation make the solution of MLR rather obvious!

When we have a matrix of y-variables $\mathbf{Y}$:

$$\boldsymbol{B} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$$

in the equation:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{B}.$$

These equations give us the **multiple linear regression** (MLR) solution.