Multivariate Analysis

BAW 2024

Rishabh Mehta

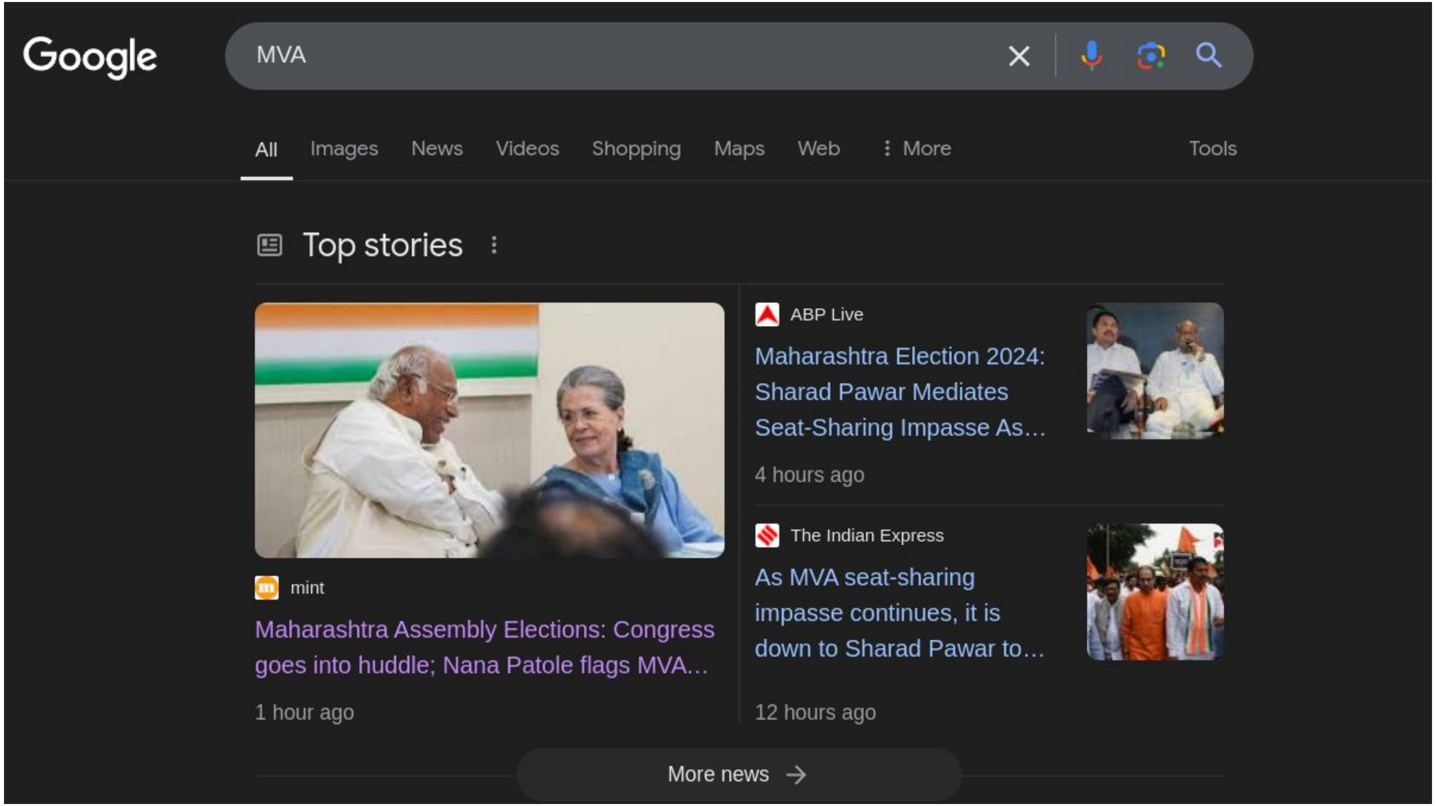
Disclaimer

This tutorial is not meant to teach you machine learning algorithms. Rather, I will focus on making you all comfortable with the idea of it, so that you can any of these tools intuitively with minimal efforts.

Some of these slides have been taken from some other tutorials in the past, and I am thankful to those.

MVA: Let me google that for you!

MVA: Let me google that for you!



MVA: What wikipedia says?

Multivariate analysis [edit]

See also: Univariate analysis

Multivariate analysis (**MVA**) is based on the principles of multivariate statistics. Typically, MVA is used to address situations where multiple measurements are made on each experimental unit and the relations among these measurements and their structures are important.[1] A modern, overlapping categorization of MVA includes:[1]

- Normal and general multivariate models and distribution theory
- The study and measurement of relationships
- Probability computations of multidimensional regions
- The exploration of data structures and patterns

Multivariate analysis can be complicated by the desire to include physics-based analysis to calculate the effects of variables for a hierarchical "system-of-systems". Often, studies that wish to use multivariate analysis are stalled by the dimensionality of the problem. These concerns are often eased through the use of surrogate models, highly accurate approximations of the physics-based code. Since surrogate models take the form of an equation, they can be evaluated very quickly. This becomes an enabler for large-scale MVA studies: while a Monte Carlo simulation across the design space is difficult with physics-based codes, it becomes trivial when evaluating surrogate models, which often take the form of response-surface equations.

MVA: What wikipedia says?

Multivariate analysis [edit]

See also: Univariate analysis

Multivariate analysis (**MVA**) is based on the principles of multivariate statistics. Typically, MVA is used to address situations where multiple measurements are made on each experimental unit and the relations among these measurements and their structures are important.[1] A modern, overlapping categorization of MVA includes:[1]

- Normal and general multivariate models and distribution theory
- The study and measurement of relationships
- Probability computations of m
- The exploration of data

Multivariate analysis can be … late the effects of variables for a hierarchical "s… late analysis are stalled by the dimensionality of the problem. These … ugh the use of surrogate models, highly accurate approximations of the physics-based code. Since surrogate models take the form of an equation, they can be evaluated very quickly. This becomes an enabler for large-scale MVA studies: while a Monte Carlo simulation across the design space is difficult with physics-based codes, it becomes trivial when evaluating surrogate models, which often take the form of response-surface equations.

Some useless sophisticated jargon…

In simpler terms…

- MVA is the analysis of relations between multiple different kinds of observations to infer/predict some properties of the underlying event.
- More simpler terms: Machine learning.

Activity: Guess the object

Observer 1: It’s white.

Activity: Guess the object

Observer 1: It’s white.

Observer 2: It has 4 extensions.

Activity: Guess the object

Observer 1: It’s white.

Observer 2: It has 4 extensions.

Physicist: It is spherical.

Activity: Guess the object

Observer 1: It’s white.

Observer 2: It has 4 extensions.

Physicist: It is spherical.

Observer 4: It says moo.

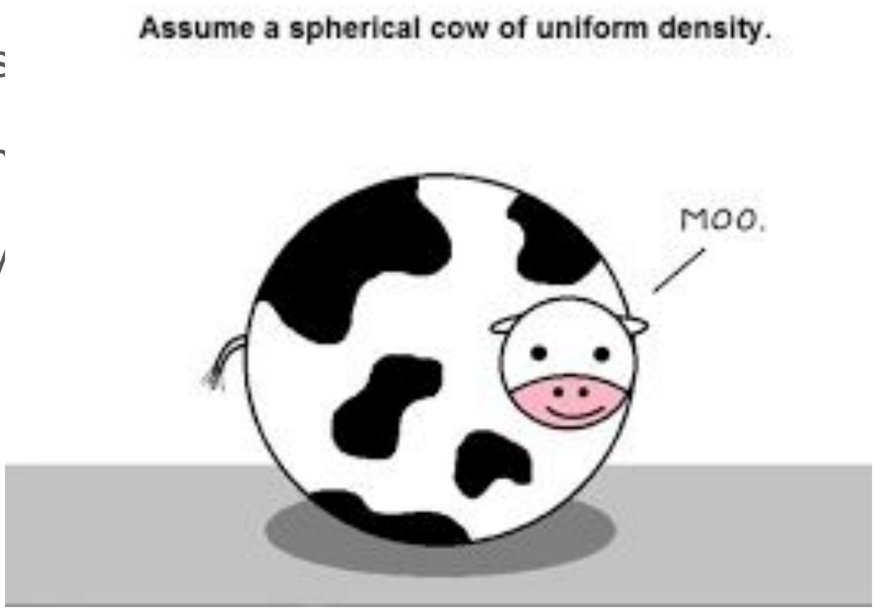
Activity: Guess the object

Observer 1: It's white.

Observer 2: It has

Physicist: It is sph

Observer 4: It say



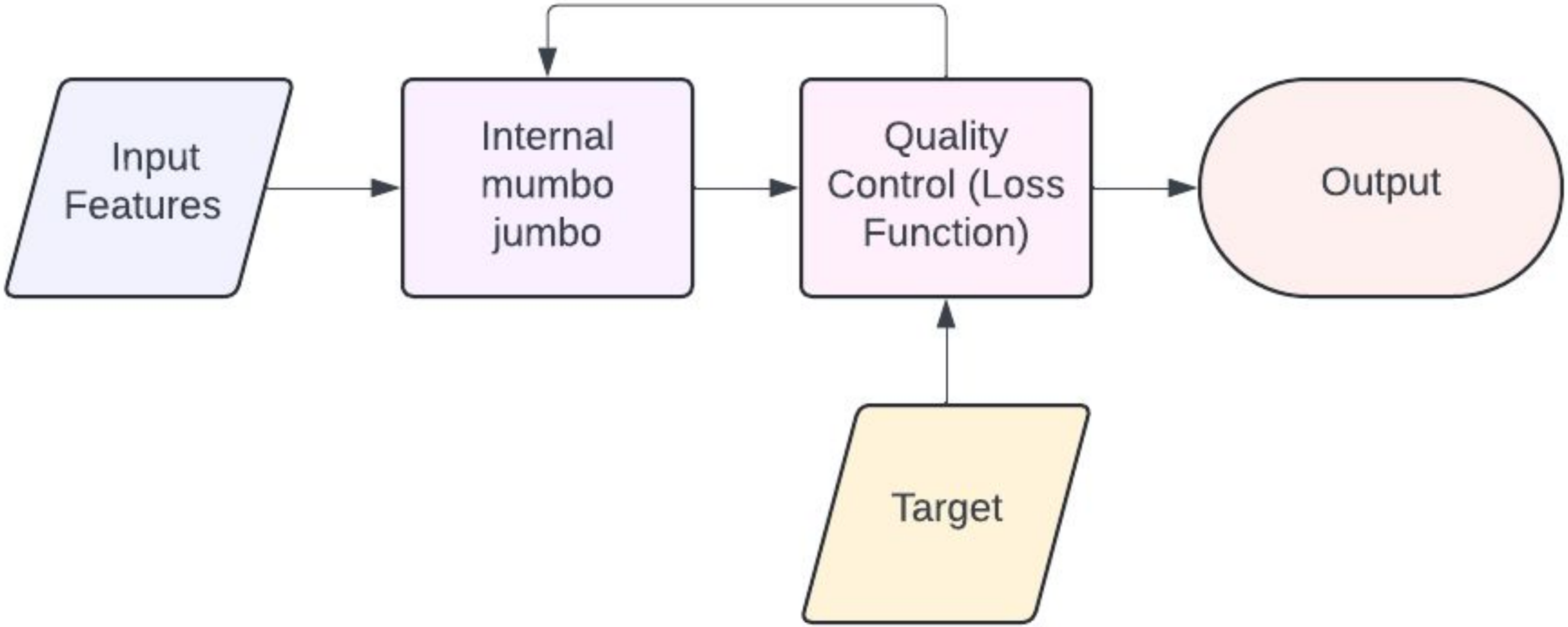
MVA: understanding the goal

- MVAs are tools to solve a particular ***problem***.
- It is VERY important to understand and analyse your problem for structures and patterns that can be exploited for solving it. (See Rahul’s talk on continuum suppression).
- We consider two types of problems.:
    - Classification (inference)
    - Regression (prediction)
- From hereon, we will use the term MVA to refer to any of the machine learning models used to solve these two problems.

ML: details

- Art of creating statistical model of data with predictive power over quantities of interest.
- This is basically equivalent to fitting a dataset with some function of input variables with a fixed number of parameters ($f(X, \omega)$).
  - X is the set of input features, ω is the set of model parameters.
- *Universal approximation theorem*: For each function there exists a neural network that can predict that function.
  - Guarantees existence, not discovery.

MVA structure



Input features

- This is the information which is always known to you, both while training and application.
- The MVA finds patterns within these features to predict the target variable.
- Choice of input feature varies according to your goal and constraints (eg. correlation with other variables etc.)

Loss Function

- This judges the quality of the output from the model and helps direct the model towards to improve its prediction.
- More formally, the model attempts to find parameters that minimise the loss function.
- The average loss, $R(\omega)$, defines a “landscape” in the parameter space of the model $f(X, \omega)$.
- The Goal: find the lowest point in the landscape defined by an infinite amount of data by navigating the landscape defined by a finite amount of data.

Minimizing the Average Loss

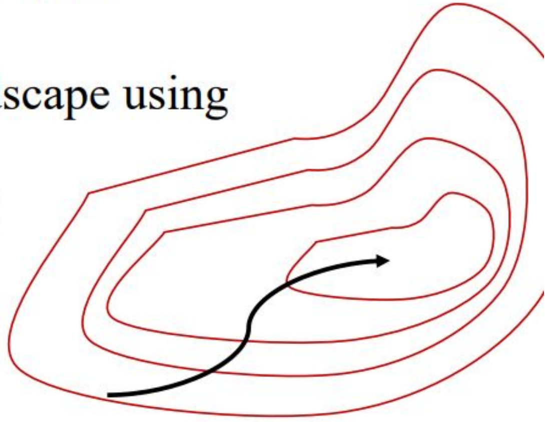
This is typically done by moving in the direction of steepest descent using **Stochastic Gradient Descent**.

At every step:

1. Compute the local gradient of $R(\omega) = \frac{1}{n}\sum_{i=1}^{n} L(t_i, f_i)$ using a *batch* of training data with $\boldsymbol{n} \ll \boldsymbol{N}$.

2. Move to the next position in the landscape using

$$\omega_{j+1} = \omega_j - \eta \nabla R$$



Credits: Harrison B. Prosper

Minimizing the Average Loss

Why does this algorithm

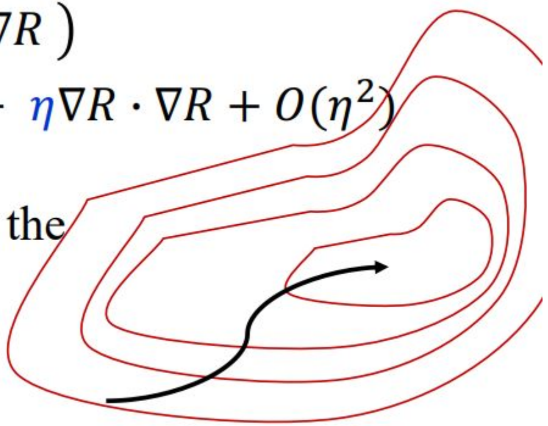
$$\omega_{j+1} = \omega_j - \eta \nabla R$$

work?

Here's why:

$$R(\omega_{j+1}) = R(\omega_j - \eta \nabla R)$$
$$= R(\omega_j) - \eta \nabla R \cdot \nabla R + O(\eta^2)$$

If the $O(\eta^2)$ can be neglected, and since the $O(\eta)$ term is always negative, then $R(\omega_{j+1}) < R(\omega_j)$.



Credits: Harrison B. Prosper

Common loss functions

Quadratic loss: $L(t, f) = (t - f)^2$

Exponential loss:

$$L(y, f) = \exp(-wtf/2)$$

Binary cross entropy loss:

$$L(y, f) = -[t \log f + (1 - t) \log(1 - f)]$$

Different Models

ML Models

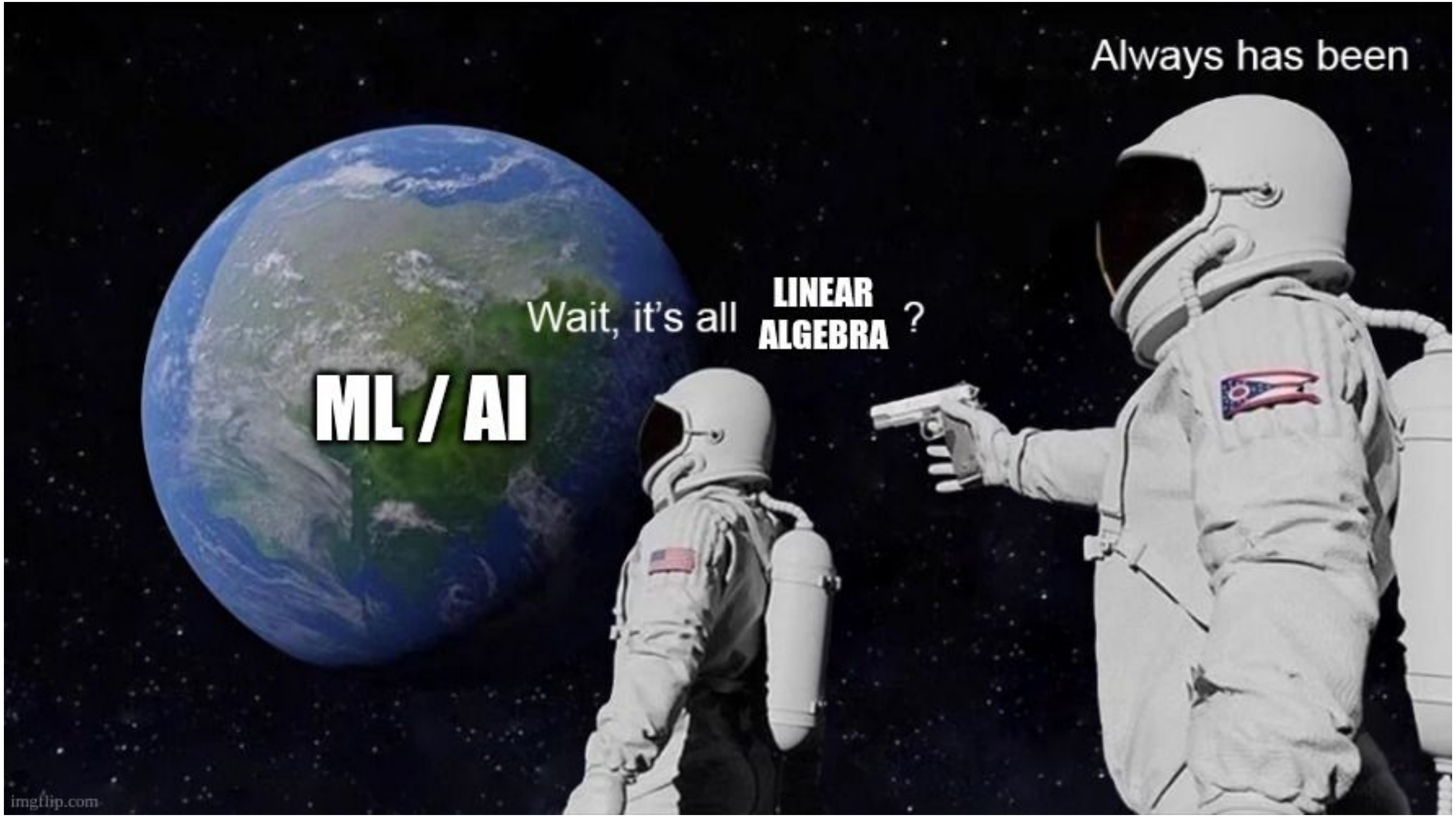
Decision Trees

- Random Forest
- Boosted DTs

Neural Nets

- DNN
- GNN
- CNN
- RNN
- Transformers

Neural Nets


Always has been
Wait, it's all LINEAR ALGEBRA ?
ML / AI
imgflip.com

Linear Algebra basics

- Multiplication of two matrices $M1(n,m)$ and $M2(m,p)$ is $M3(n,p)$.
- Addition of two matrices $M1(n,m)$ and $M2(n,m)$ is element-wise addition $M3(n,m)$.
- Transpose of a matrix $M1(n,m)$ is $M2(m,n)$: $M1(i,j) = M2(j,i)$
- Aggregation: reduction of matrix size.

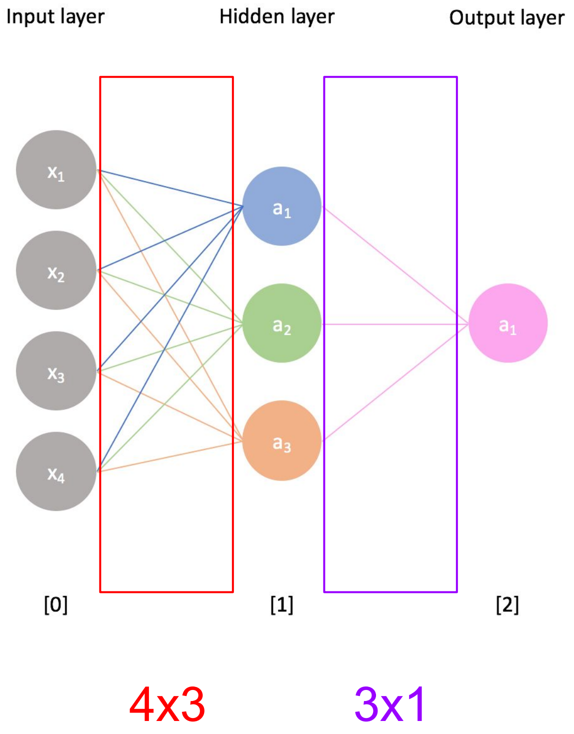
Linear Algebra basics

- Multiplication of two matrices M1(n,m) and M2(m,p) is M3(n,p).
- Addition of two matrices M1(n,m) and M2(n,m) is element-wise addition M3(n,m).
- Transpose of a matrix M1(n,m) is M2(m,n): M1(i,j) = M2(j,i)
- Aggregation: reduction of matrix size.

Congratulations! You now know all the maths behind Machine Learning!

Neural Net from scratch

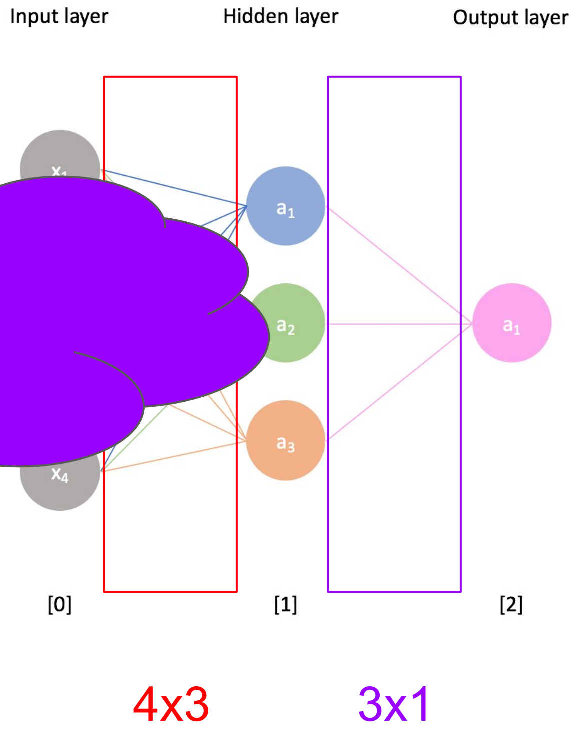
- Input is a vector (1xn)
- Create a series of matrices to multiply to the input.
- The last matrix is an aggregator (mx1) which gives the output value.
- Profit??



Neural Net from scratch

- Input is a vector (1xn)
- Create a series of matrices to multiply to the input.
- The last matri
(mx1) whi
- Profit??

Is it this simple?



It can’t be that simple!

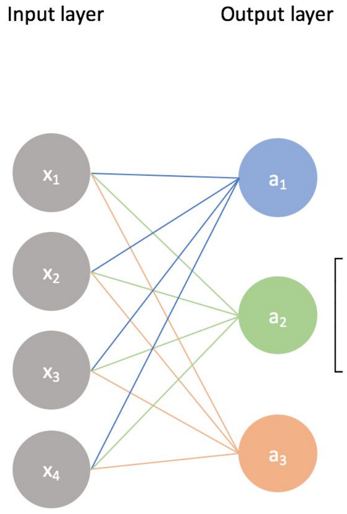
- Well, yes and no.

It can’t be that simple!

- Well, yes and no.
- What will happen if all the input features are 0?

It can't be that simple!

- Well, yes and no.
- What will happen if all the input features are 0?
    - Solution: add a bias term.

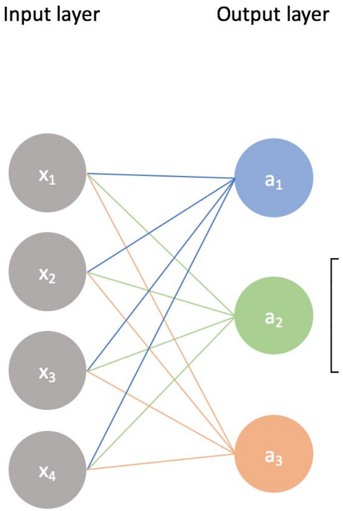


A simple neural network

$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \boxed{\begin{bmatrix} b \\ b \\ b \end{bmatrix}} = \begin{bmatrix} w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \end{bmatrix} \xrightarrow{activation} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

It can’t be that simple!

- Well, yes and no.
- What will happen if all the input features are 0?
  - Solution: add a bias term.
- Everything is linear.

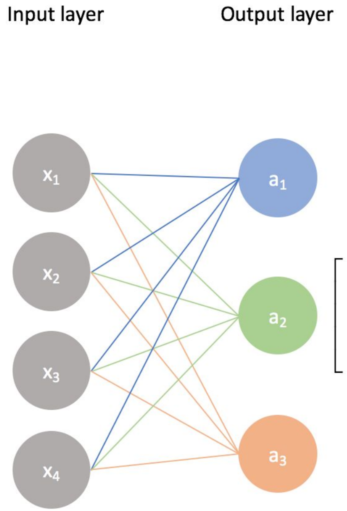


A simple neural network

$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix} = \begin{bmatrix} w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \end{bmatrix} \xrightarrow[activation]{} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

It can’t be that simple!

- Well, yes and no.
- What will happen if all the input features are 0?
  - Solution: add a bias term.
- Everything is linear.
  - Solution: activation function.



A simple neural network

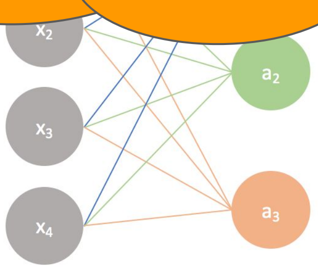
$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}+\begin{bmatrix} b \\ b \\ b \end{bmatrix}=\begin{bmatrix} w_1x_1+w_2x_2+w_3x_3+w_4x_4+b \\ w_1x_1+w_2x_2+w_3x_3+w_4x_4+b \\ w_1x_1+w_2x_2+w_3x_3+w_4x_4+b \end{bmatrix}\boxed{\underset{activation}{\rightarrow}}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

It can’t be that simple!

- Well, yes and no.
- What will happen if all the input features are 0?
  - Solution: add a bias term.
- Everything is linear
  - Solution: activat

le neural network

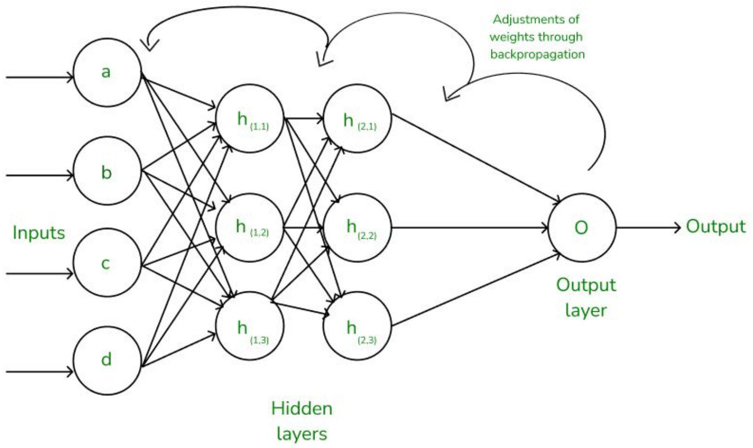
Congratulations, you have a random garbage generator.



$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix} = \begin{bmatrix} w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \end{bmatrix} \xrightarrow{activation} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

But how does it learn?

- What we have seen so far is a feed forward neural network.
- Loss function: the ***brain*** of any neural network!
- Back propagation: Iteratively propagate error and update weights throughout the network via SGD.



Hands On Session

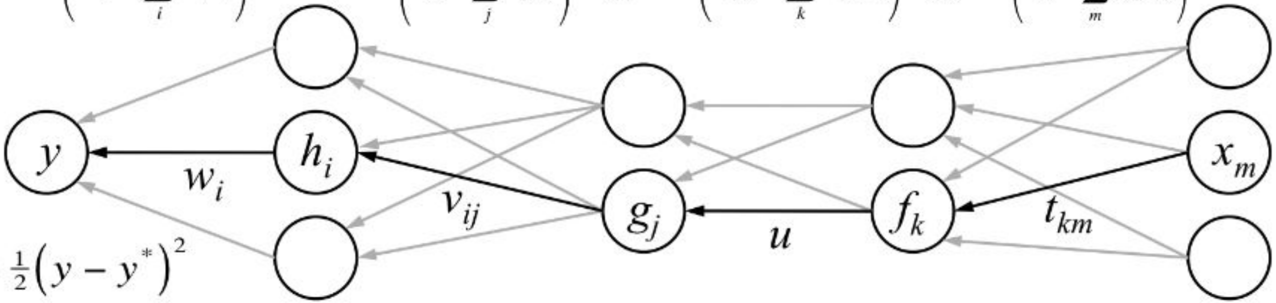
- Train a classifier for continuum suppression using xgboost.
- Train a classifier for continuum suppression using Neural Networks.

Challenge

- Train the best model with the given dataset with < 15% correlation with both Mbc and deltaE.
- The model will be evaluated by us in an unknown dataset. Winner gets a prize!

Back-propagation (formally)

$$y = \sigma\left(w_0 + \sum_i w_i h_i\right) \qquad h_i = \sigma\left(v_{i0} + \sum_j v_{ij} g_j\right) \qquad g_j = \sigma\left(u_{j0} + \sum_k u_{jk} f_k\right) \qquad f_k = \sigma\left(t_{k0} + \sum_m t_{km} x_m\right)$$



$$E = \tfrac{1}{2}\left(y - y^*\right)^2$$

$$\frac{\partial E}{\partial h_i} = \left(y - y^*\right) y(1-y) w_i$$

$$\frac{\partial E}{\partial g_j} = \left(y - y^*\right) y(1-y) \sum_i w_i h_i(1-h_i) v_{ij} = \sum_i h_i(1-h_i) v_{ij} \frac{\partial E}{\partial h_i}$$

$$\frac{\partial E}{\partial u} = \left(y - y^*\right) y(1-y) \sum_i w_i h_i(1-h_i) v_{ij} g_j(1-g_j) f_k = g_j(1-g_j) f_k \frac{\partial E}{\partial g_j}$$