# Machine Learning (2016.10.12)

* Neural Nets
  + Family of function
  + To learn features automatically
* Machine Learning
  + Learn “rules” from data,
  + Adapt to changes, and
  + Improve performance with experience
* AI / data science bottleneck
  + Your idea – data (feature/design) – useful information 🡪(machine learning) 🡪 your market
* Representation learning
  + Figure out which features are good for learning
* Two layer neural network means two regressions
* Why go deep?
  + 2- layer Neural Nets are already universal function approximators
    - No matter what function is pulled out, I can express the function as the linear combination of phi
* Why Nueral?
  + Need fewer parameter and still in the universal function approximators
* Different Levels of Abstraction
  + Processing
    - Pixels 🡪 1st layer “Edges” 🡪 2nd layer “Objective Parts” 🡪 3rd layer “Objects”
* How to constrain parameter
  + Adding layer, exponential increase of dimension
  + ML is about optimization
    - Most of the time
    - Min (1/n) sum(l(f(x),y) 🡪 in some function in the w\_dimension
  + We can search for the richer family of the neural network
* Deep Network Training
  + Problem : Nonconvexity
    - Z = xy is non convex function.
    - What is not convex function of 🡪 z is function of x or z is function of y, then it’s linear function and convex. But when we put them together, then nonconvex
  + Stochatic Gradient Descent for nonconvexity
    - Look closely and see how to precede
    - This can be very complex (think as climbing up the mountain)
    - Can be very slow
* How to create neural network?
  + Use chain rule of the derivative
  + Pre-forward the layer 🡪 look at the loss, and compute the gradient from the loss 🡪 get the gradient near the top
  + Using back propagation 🡪 blaming the layer? Or function above to make more progress
* Problem: Vanishing Gradients
* Why still using neural network?
  + Work heuristically
* Using unlabeled data
  + Good for learning good features
* Unsupervised pre-training
  + How to use unlabeled data to train?
    - Self-supervision, have some input. I can reconstruct that input.
    - Take some input, reconstruct like x\_hat = g(f(x)), where x is input
    - When we train, we need to constrain the function correctly
    - We want (number of layer) D < M (number of input) because we want to constrain and if D is bigger, need to cut down as we go further
    - Loss = (x – DECODER(ENCODER(x)))