Deep Learning Overview

Spencer Lyon
UCF MSDA Big Data Seminar
July 24, 2020

Table of Contents

*: Examples in these slides come from multiple sources including “Deep Learning” by Ian Goodfellow and Yoshua Bengio and Aaron Courville

ML Limits

ML Applications

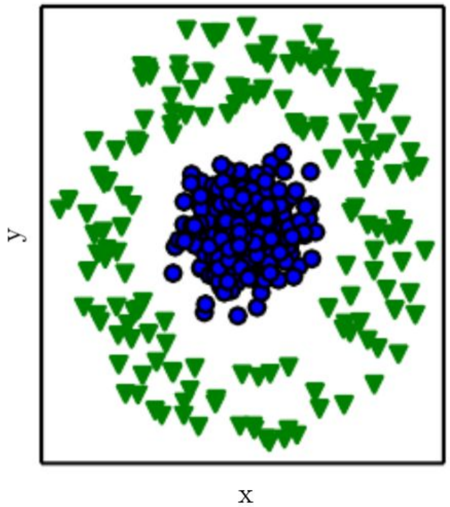
- Incredible results have come from relatively simple algorithms
- I’m sure we all know of examples… sharing time!

Representation: A Common Pre-Problem

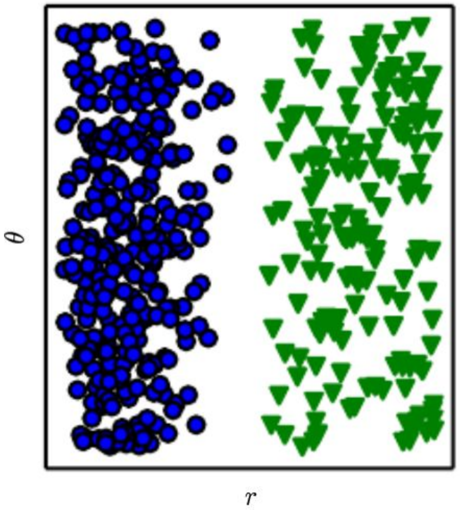
- Often raw input data needs to be transformed, processed, or altered before it can be useful in ML systems
- Example: Medical imaging
  - Goal: use logistic regression to identify presence of tumor in MRI
  - Raw data is pixels from MRI
  - Just ingesting raw pixel values, logistic regression would almost surely fail
    - Each pixel has near zero correlation with the existence of a tumor
  - Passing a formalized report from a radiologist would yield better reports
- In imaging example, the radiologist was feature engineer
- Almost always requires a subject matter expert, combine with algorithm expert
- This is known as **representation or feature engineering**

Representation example

- Example: classify as blue circle or green triangle given (x, y) coordinate

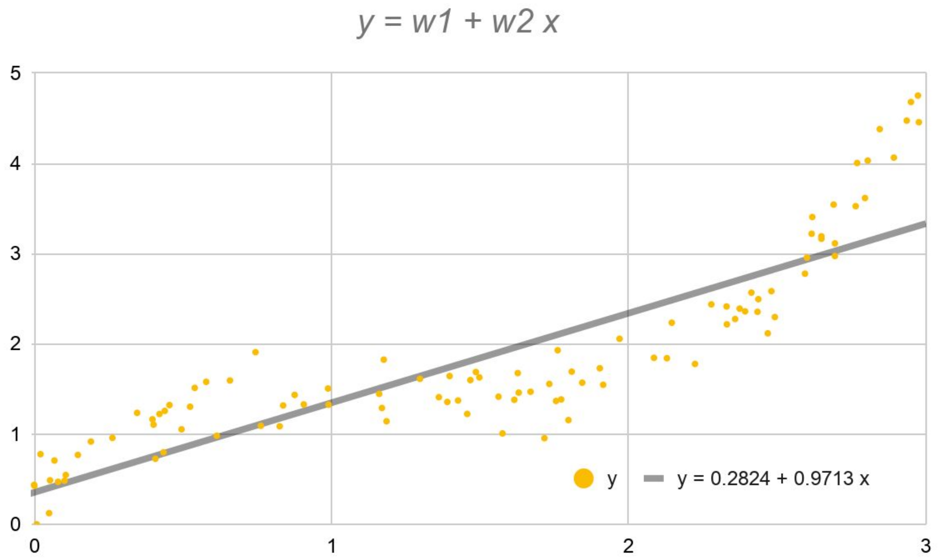


- Alternative representation: use polar coordinates (angle theta and radius r)



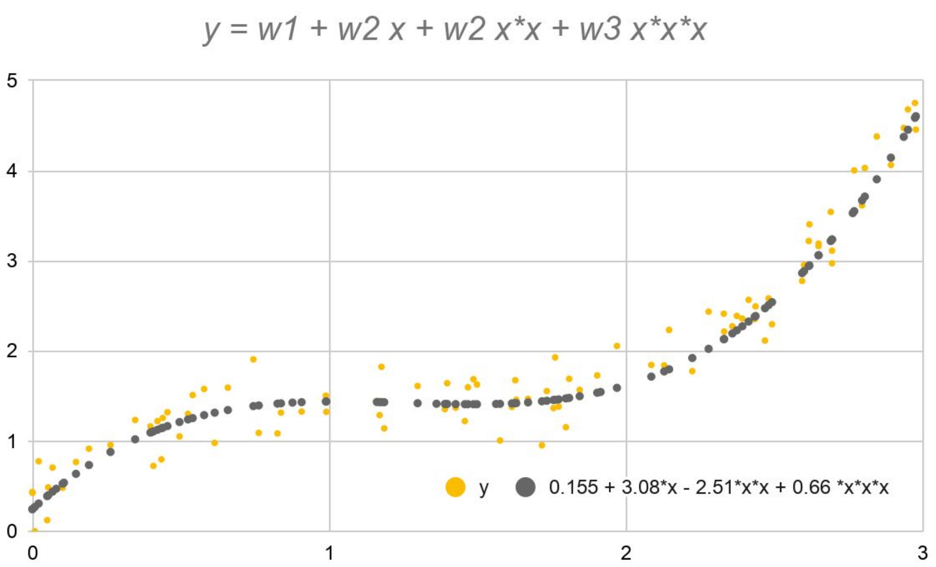
Source

Feature Engineering Example: Polynomial Regression



- Two dimensional dataset
- Goal: linear model that maps x to y
- Notice fit is “fine”, but we could do better…

… Continued



- Instead of using a linear model in x, we use a 3rd order polynomial in x
- Fit is now much better
- The model is linear in weights (w), but non-linear in input features (x)
- In this case we did feature engineering to arrive at a better model

Thoughts

- The data from the previous example was contrived
- To build it, I made up a cubic polynomial, evaluated it at random x points, then added some noise
- There is no surprise that a cubic polynomial regression fit the data well
- But… it required that we recognize that the data looks cubic (domain expertise) and that we alter the x points (feature engineering)
- In more realistic examples, both these parts are much harder

Deep Learning to the Rescue!

Deep Learning

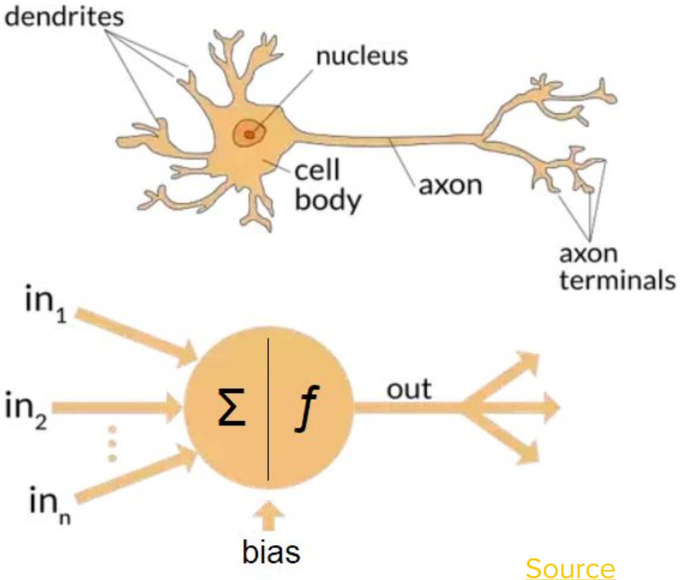
- One benefit of Deep Learning is that is capable of “automatic feature engineering”
- Demo

Deep Learning Architectures

Artificial Neural Network

An artificial neural network (ANN) is a specific mathematical model that bears resemblance to the neural structure of neurons and axons (connections between neurons)

ANNs started with the notion of the **perceptron**



Source

Multi-layer Perceptron

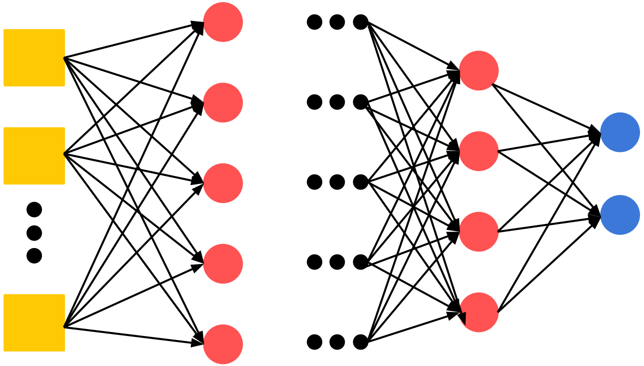
The multi-layer perceptron extends this concept

Stacks multiple perceptrons on top of one another

Layers multiple stacks of perceptrons, one after another

Mathematically, this is nested linear regression separated by activation functions:

$$y = f_{\dots}\left(f_2(f_1(Xw_1 + b_1)w_2 + b_2)w_{\dots} + b_{\dots}\right)w_{N+1} + b_{N+1}$$

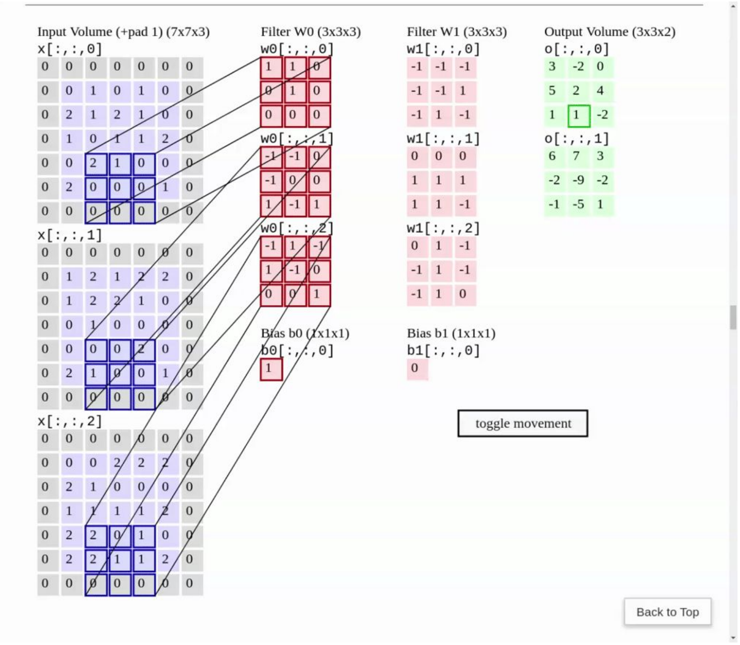


CNN

Another type of neural network is a convolutional neural network, or CNN

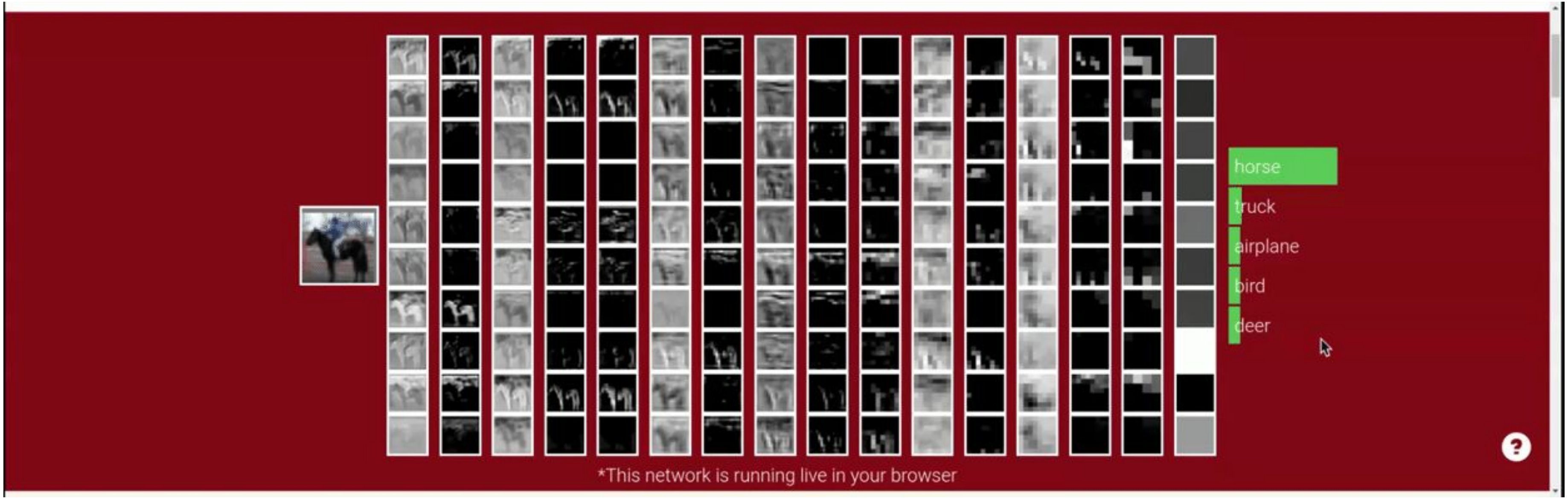
The core operation of a perceptron is matrix multiplication

The core operation of a CNN is the convolution



Source

CNN Example



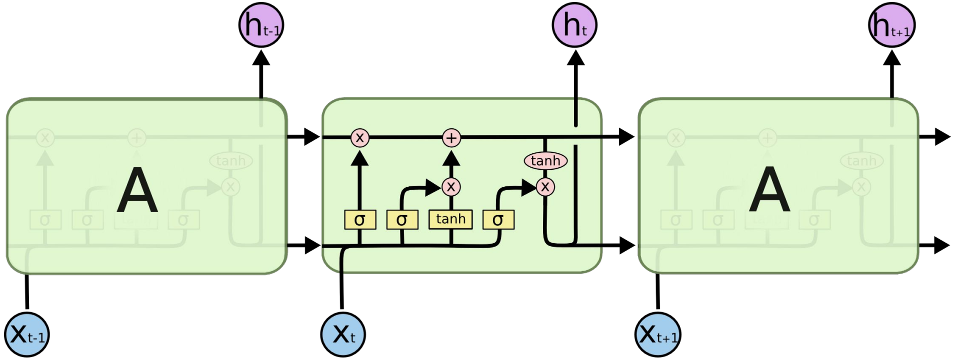
Source

RNN

The last main type of ANN is the recursive neural network or RNN

The RNN comes in multiple flavors, but the core idea is that it can retain a processed knowledge of previous observations

This is accomplished by the input layer being a concatenation of raw data and the RNN’s output from the previous observation



Source

What Can Go Wrong?

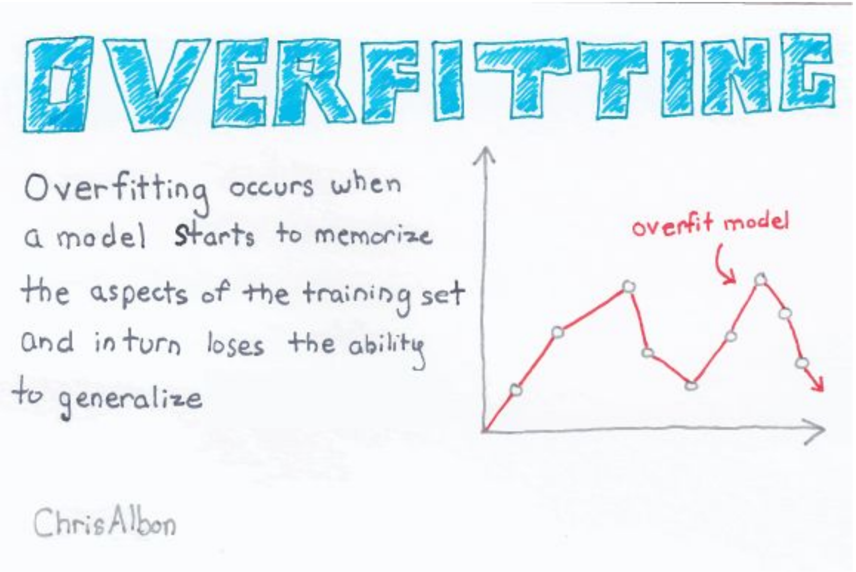
Common Pitfalls

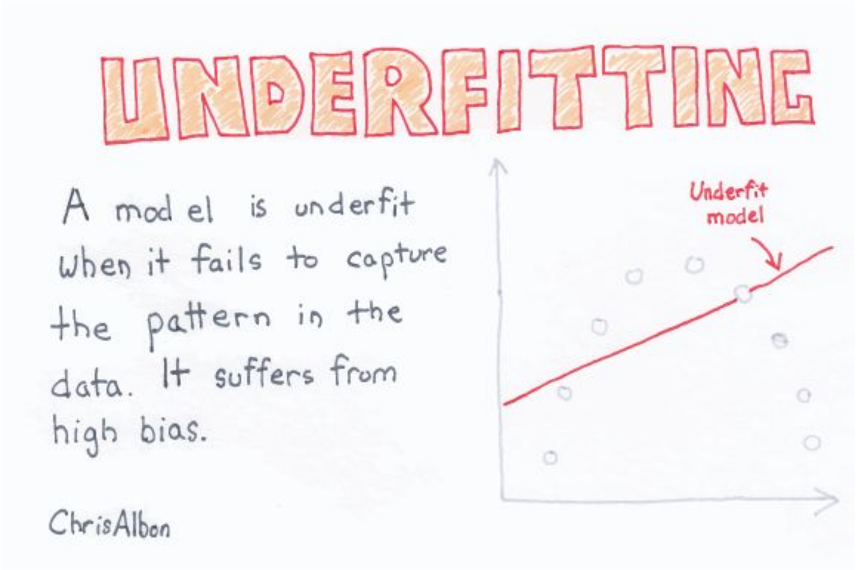
There are a number of common pitfalls that occur when doing ML work

Some of these only apply to a specific class of model or algorithm

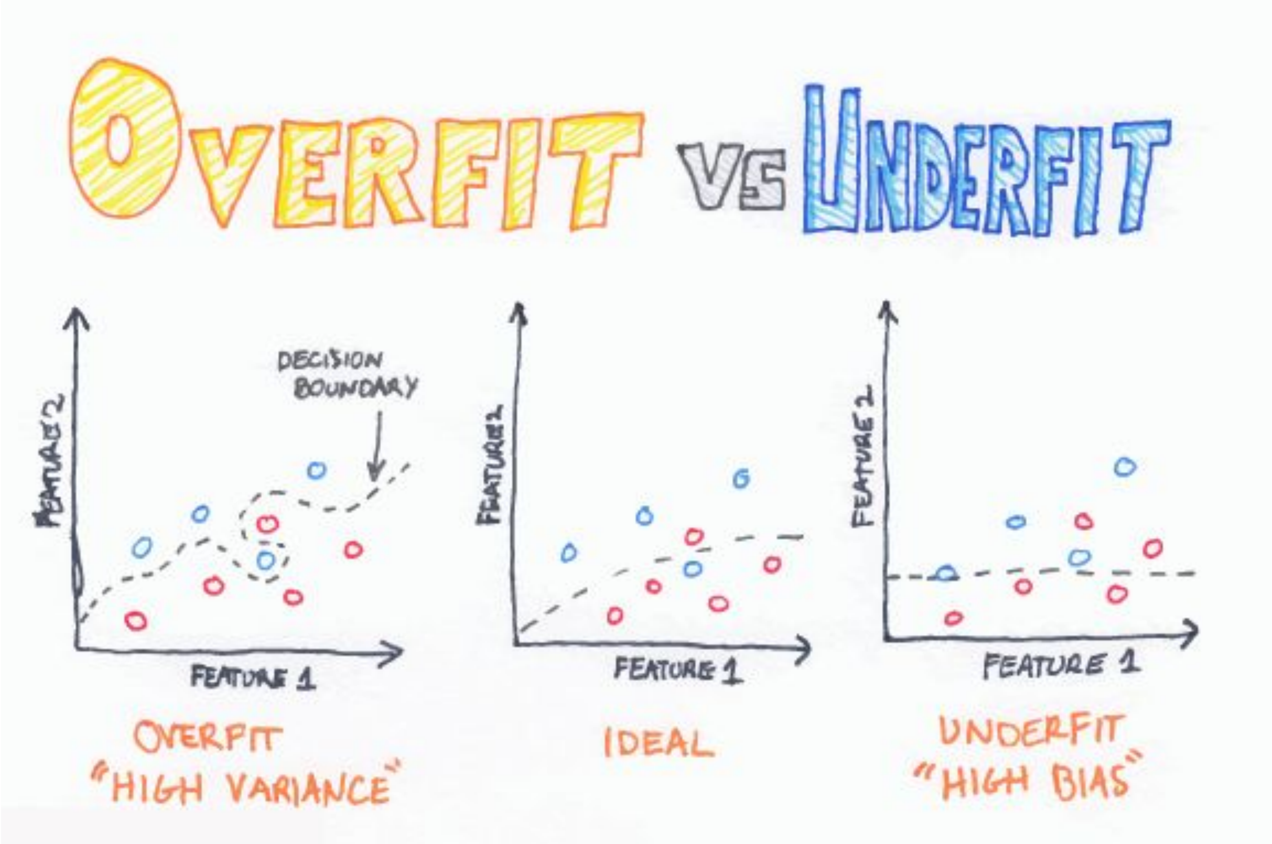
Most of them are more extreme with deep learning, so we'll focus our discussion there

Overfitting and Underfitting





Underfitting/Overfitting Tradeoff



Regularization

- To overcome underfitting:
    - Use more data
    - Use a model with higher complexity
- To overcome overfitting:
    - Use more data
    - Use a model with lower complexity
    - Regularization
        - L1 and L2 penalties (as in elastic net, ridge, or lasso regression)
        - Dropout
        - Early stopping
        - Data Augmentation

Model Selection

- Each mathematical structure and set of hyperparameters defines a different model
- Difficult to know a-priori which model will work best for application
- Would like to use data to select **best** model, but risk of overfitting increases
- Need to use a validation procedure:
  - Determine pool of candidate models
  - Split model into (at least) training and validation sets
  - Train all models on training sets, evaluate goodness of fit on validation sets
  - Choose model that does **best** on validation set according to predetermined metric