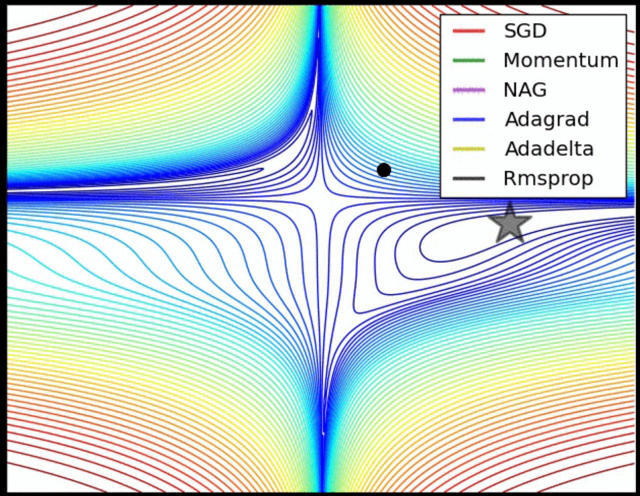
Booz Allen



http://sebastianruder.com/optimizing-gradient-descent/

Machine Learning Fundamentals through the Lens of TensorFlow: A Calculus of Variations for Data Science

Kirk Borne

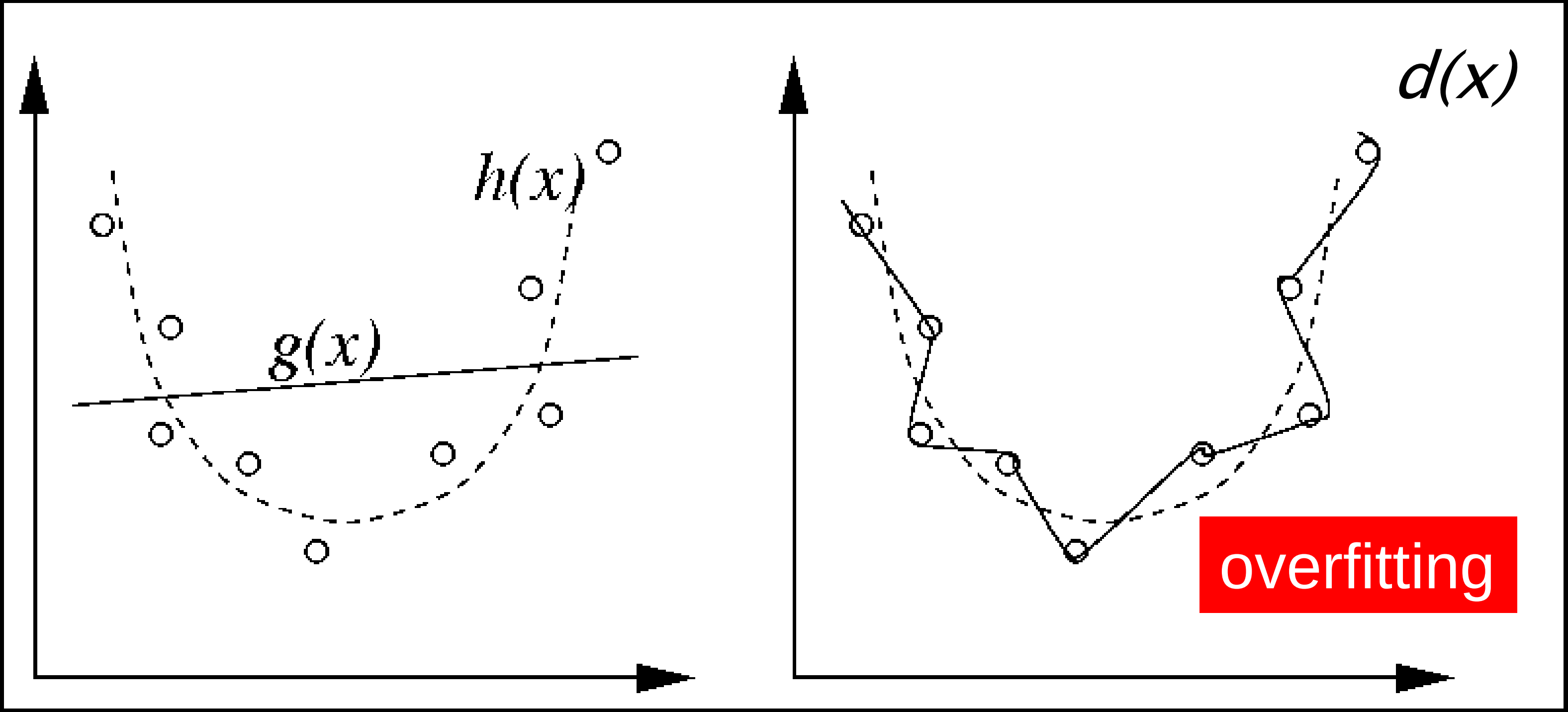
@KirkDBorne

Principal Data Scientist

Booz-Allen Hamilton, Strategic Innovation Group

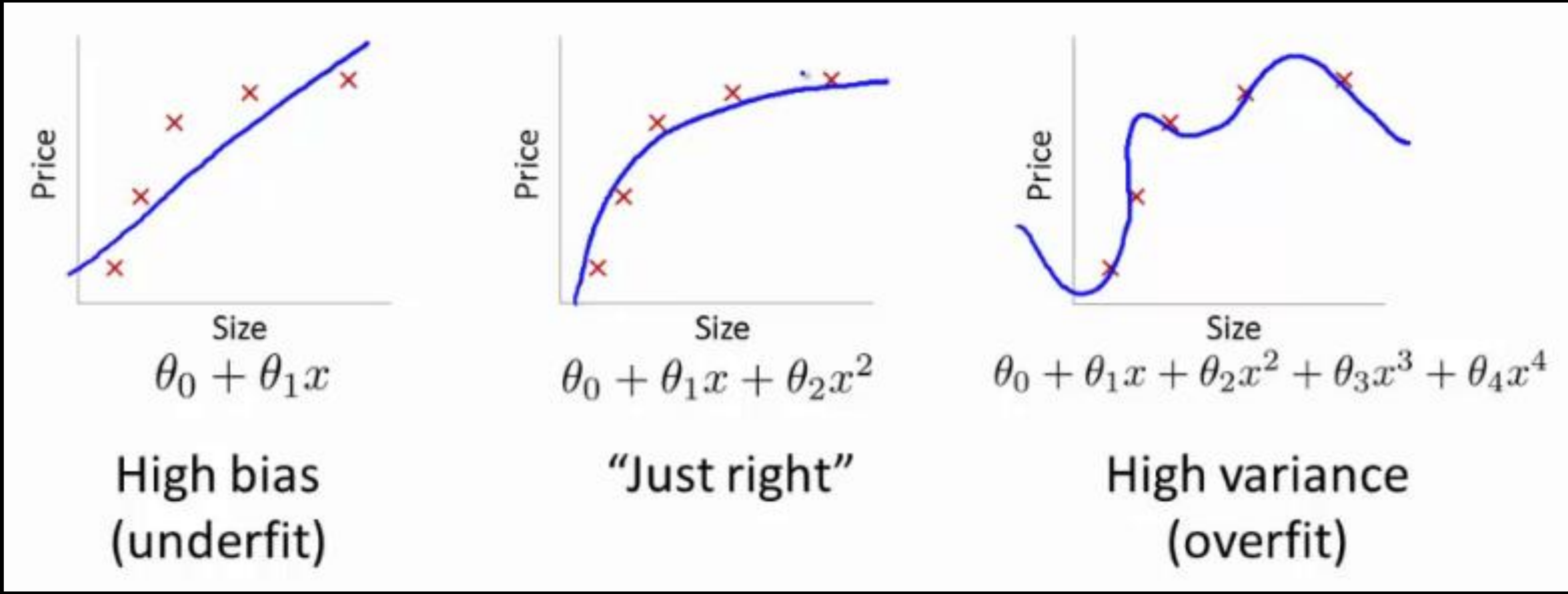
http://www.boozallen.com/datascience
http://www.jobs.net/jobs/booz-allen-hamilton/

Overfitting = bad data science!



- ***g(x)*** is a **poor fit** (a line through the plot) = **Underfitting**
- ***h(x)*** is a **good** fit (takes into account the variance in the data)
- ***d(x)*** is a **very poor** fit (fitting every point) = **Overfitting**

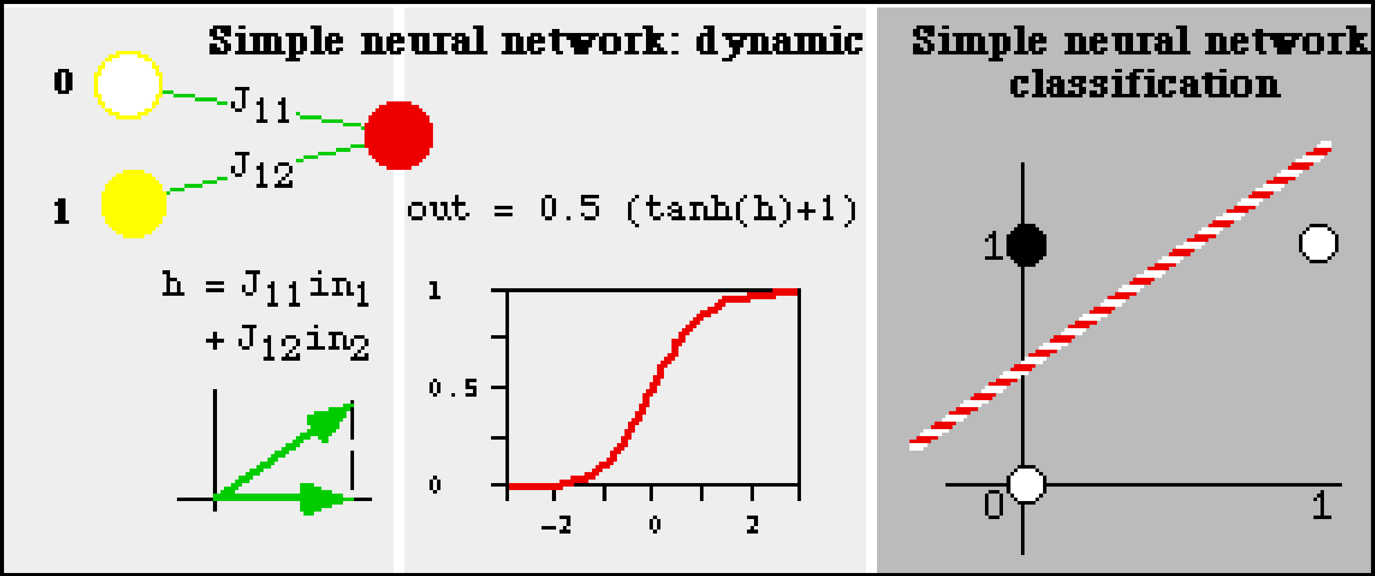
Overfitting = bad data science!



http://www.holehouse.org/mlclass/10_Advice_for_applying_machine_learning.html

Automated Wildfire Detection (and Prediction) **through Artificial Neural Networks *(ANN)***

- **Short Description of Wildfire Project:**
  - Identify all wildfires in Earth-observing satellite images
  - Train ANN to mimic human analysts' classifications
  - Apply ANN to new data (from 3 remote-sensing satellites: GOES, AVHRR, MODIS)
  - Extend NOAA fire product from USA to the whole Earth



Hazard Mapping System (HMS) ASCII Fire Product

OLD FORMAT

Lon,	Lat
-80.531,	25.351
-81.461,	29.072
-83.388,	30.360
-95.004,	30.949
-93.579,	30.459
-108.264,	27.116
-108.195,	28.151
-108.551,	28.413
-108.574,	28.441
-105.987,	26.549
-106.328,	26.291
-106.762,	26.152
-106.488,	26.006
-106.516,	25.828

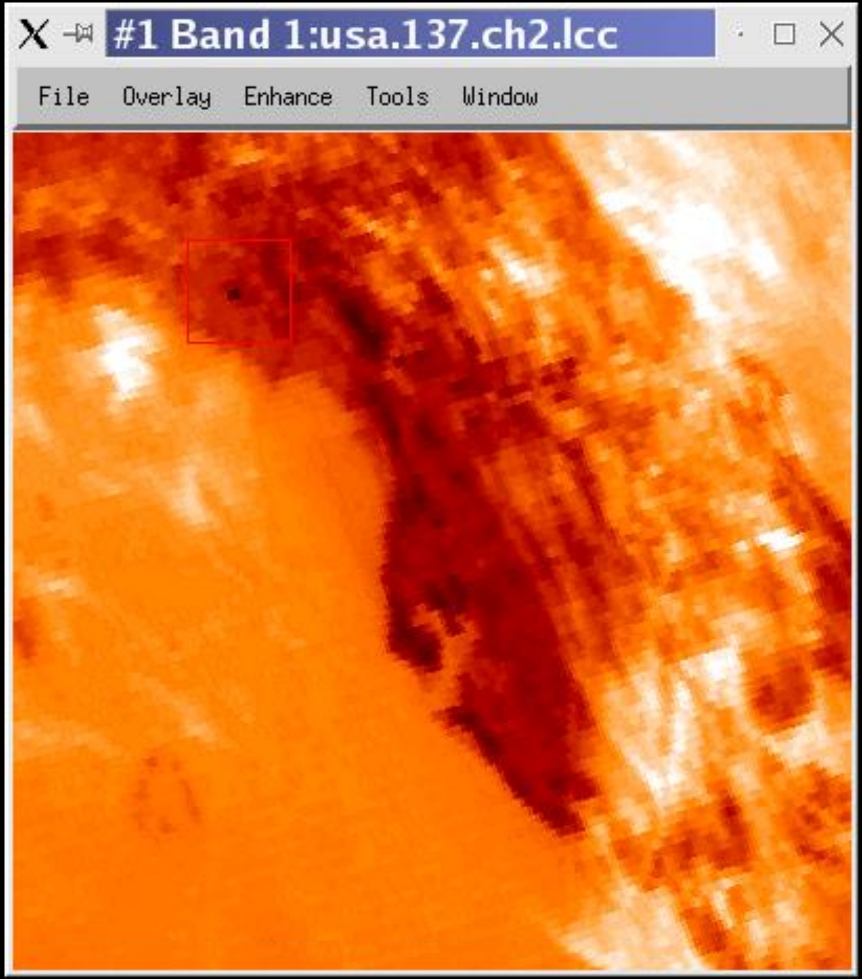
NEW FORMAT (as of May 16, 2003)

Lon,	Lat,	Time,	Satellite,	Method of Detection
-80.597,	22.932,	1830,	MODIS AQUA,	MODIS
-79.648,	34.913,	1829,	MODIS,	ANALYSIS
-81.048,	33.195,	1829,	MODIS,	ANALYSIS
-83.037,	36.219,	1829,	MODIS,	ANALYSIS
-83.037,	36.219,	1829,	MODIS,	ANALYSIS
-85.767,	49.517,	1805,	AVHRR NOAA-16,	FIMMA
-84.465,	48.926,	2130,	GOES-WEST,	ABBA
-84.481,	48.888,	2230,	GOES-WEST,	ABBA
-84.521,	48.864,	2030,	GOES-WEST,	ABBA
-84.557,	48.891,	1835,	MODIS AQUA,	MODIS
-84.561,	48.881,	1655,	MODIS TERRA,	MODIS
-84.561,	48.881,	1835,	MODIS AQUA,	MODIS
-89.433,	36.827,	1700,	MODIS TERRA,	MODIS
-89.750,	36.198,	1845,	GOES,	ANALYSIS

GOES CH2 (3.78 - 4.03 μm) – Northern Florida Fire

2003: Day 126 , –82.10 Deg West Longitude, 30.49 Deg North Latitude

File: florida_ch2.png

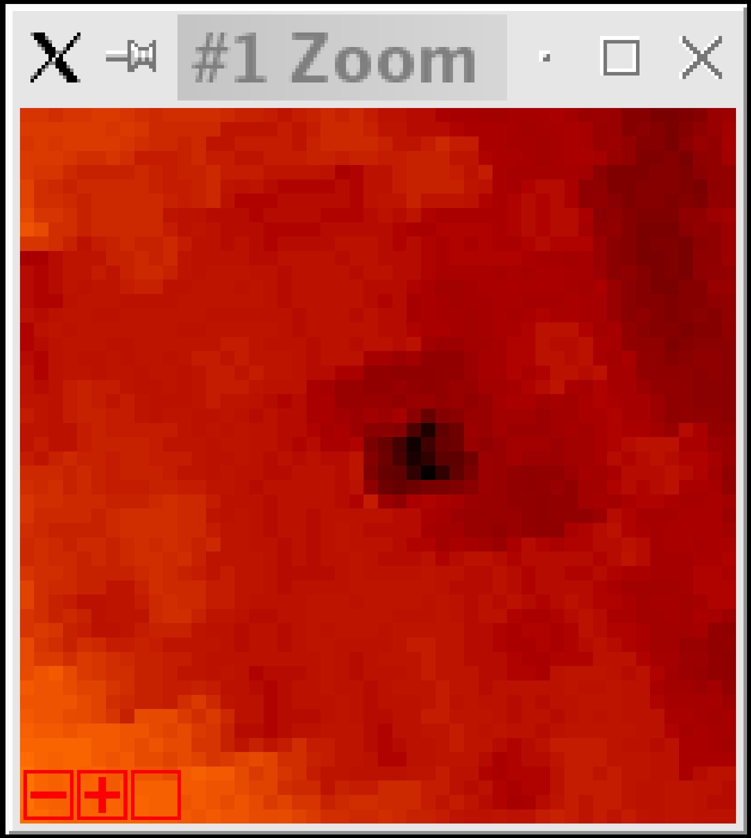


Zoom of GOES CH2 (3.78 - 4.03 µm) – Northern Florida Fire

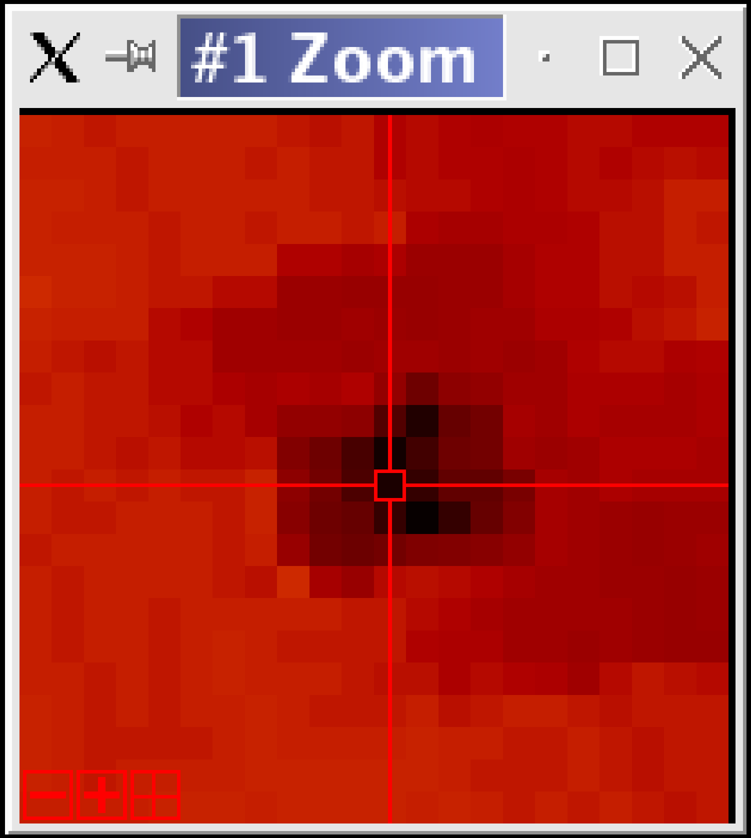
2003:Day 126, –82.10 Deg W Long, 30.49 Deg N Lat

Local minimum in vicinity of core pixel used as fire location.

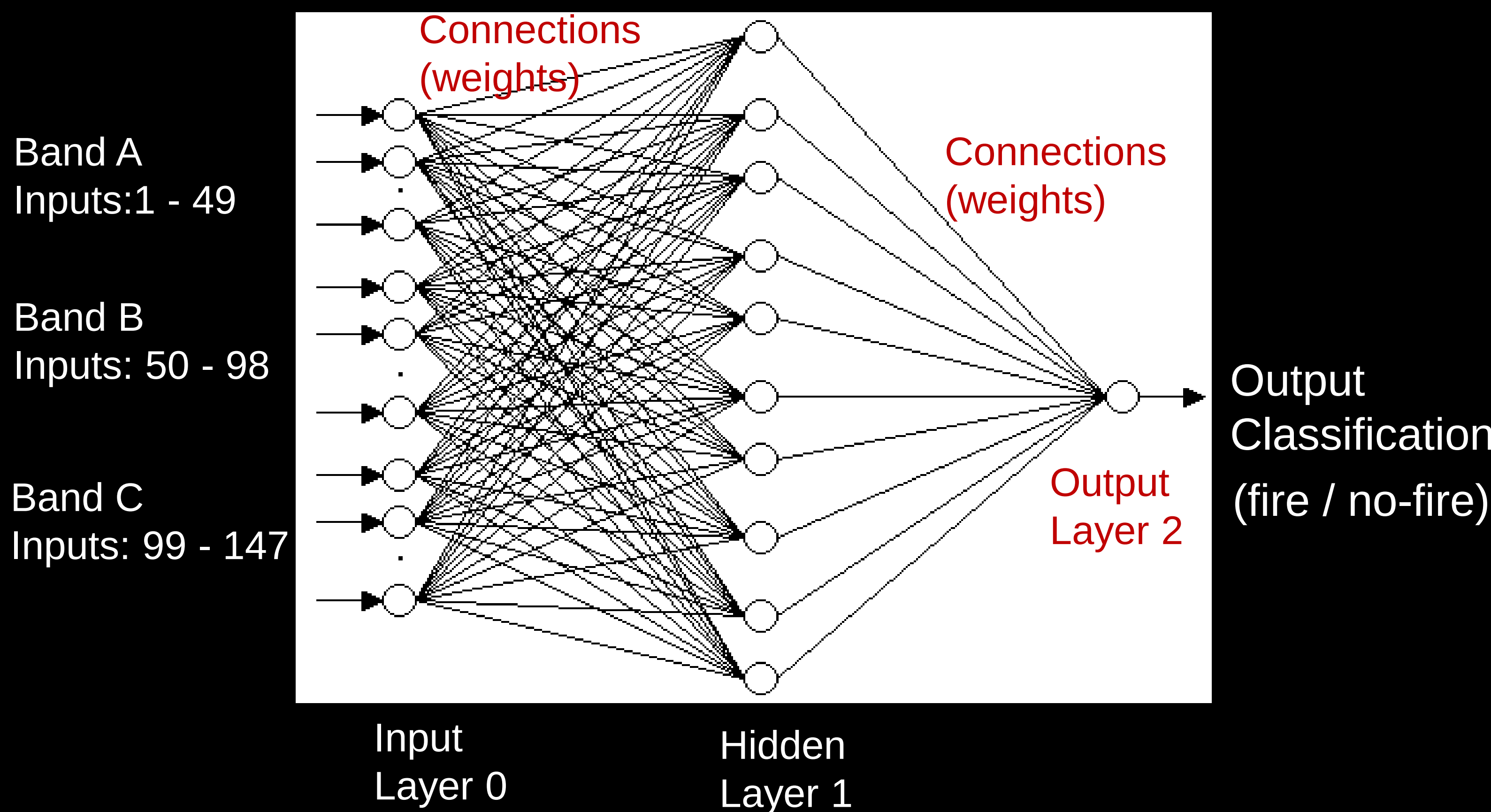
File: florida_fire_ch2_zoom.png



File: florida_ch2_zoom.png



Neural Network Configuration for Wildfire Detection Neural Network



Classification Accuracy

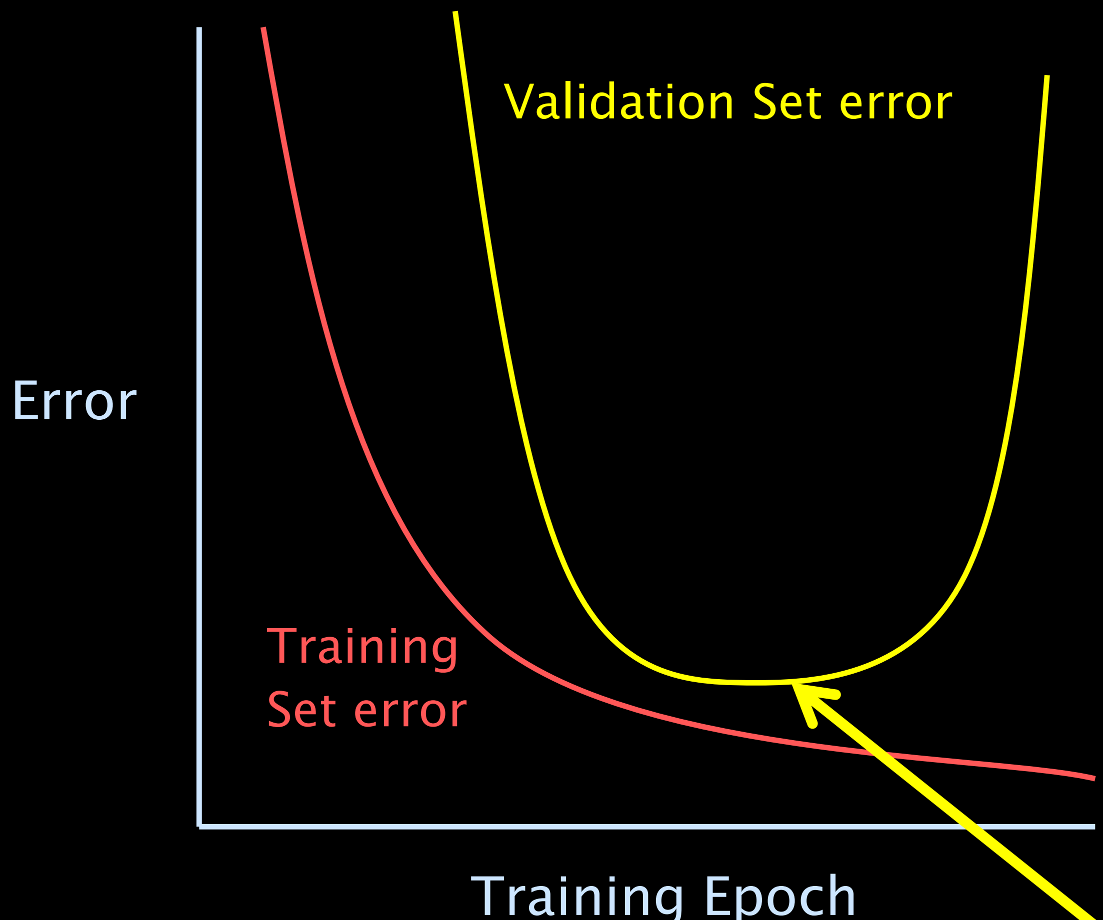
Error Matrix:

True Positive	False Positive
False Negative	True Negative

TRAINING DATA (actual classes)

NEURAL NETWORK CLASSIFICATION (output)	Class-A	Class-B	Totals
Class-A	2834 (TP)	173 (FP)	3007
Class-B	318 (FN)	3103 (TN)	3421
Totals	3152	3276	6428

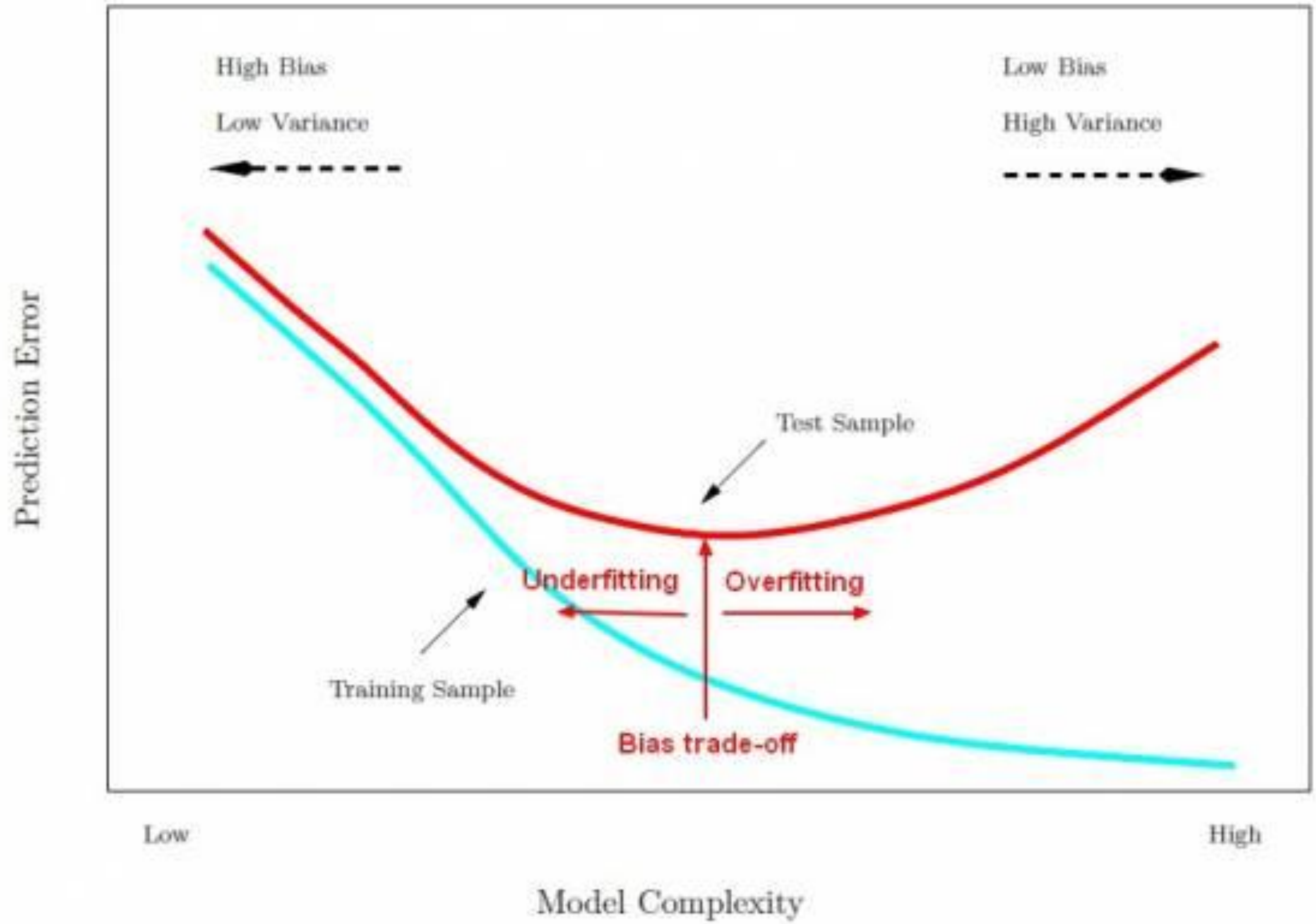
Schematic Approach to Avoiding Overfitting



To avoid overfitting, you need to know when to stop training the model.
Although the Training Set error may continue to decrease, you may simply be overfitting the Training Data.
Test this by applying the model to Validation Data Set (not part of Training Set).
If the Validation Data Set error starts to increase, then you know that you are overfitting the Training Set and it is time to stop!

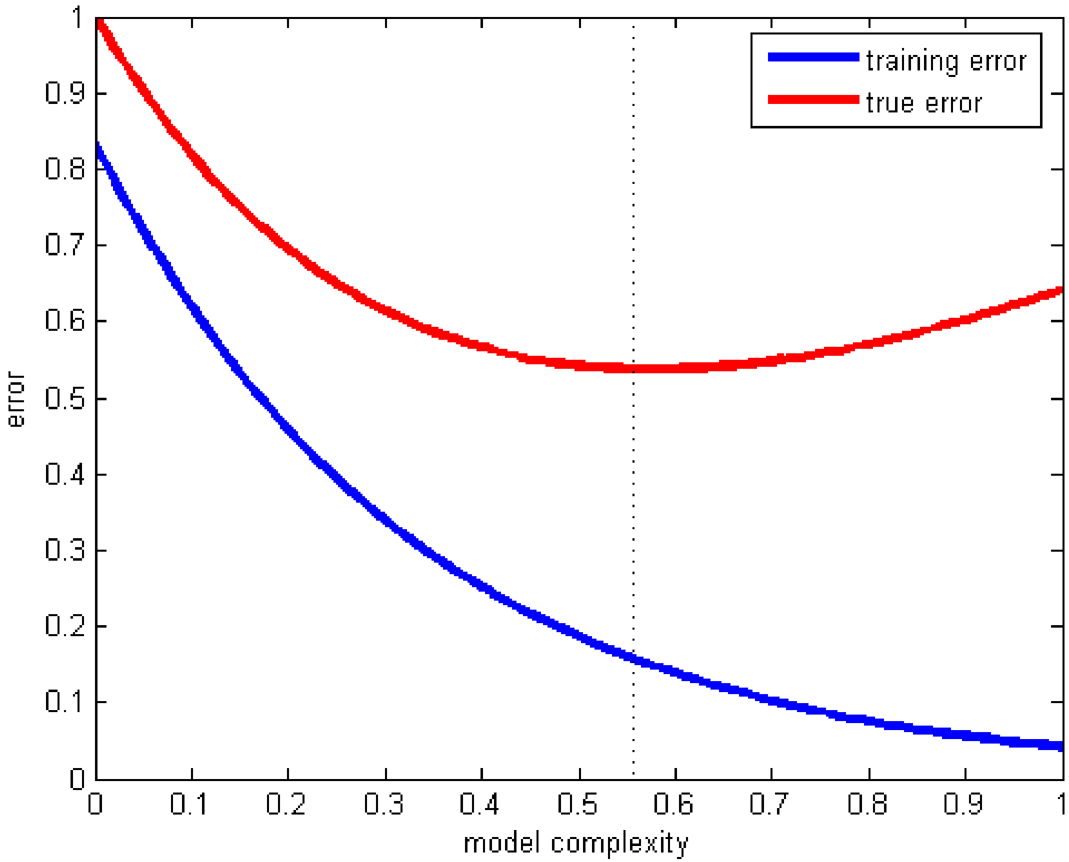
STOP Training HERE !

Overfitting = bad data science!



http://gerardnico.com/wiki/data_mining/bias_trade-off

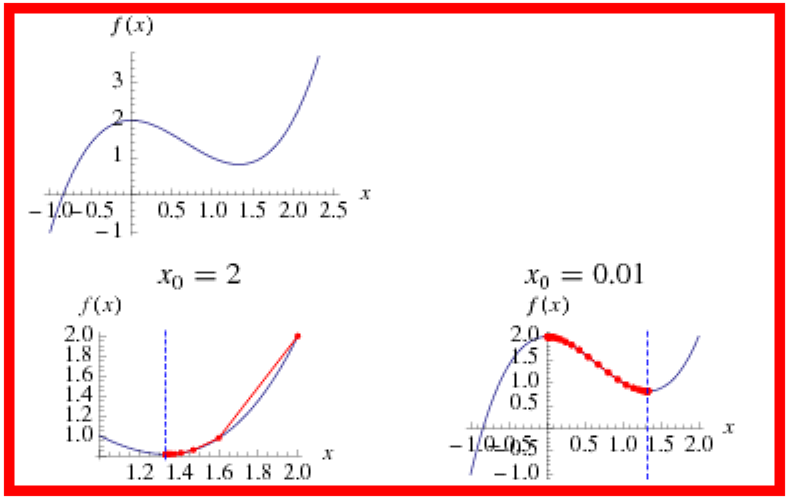
Minimize True Error = good data science!

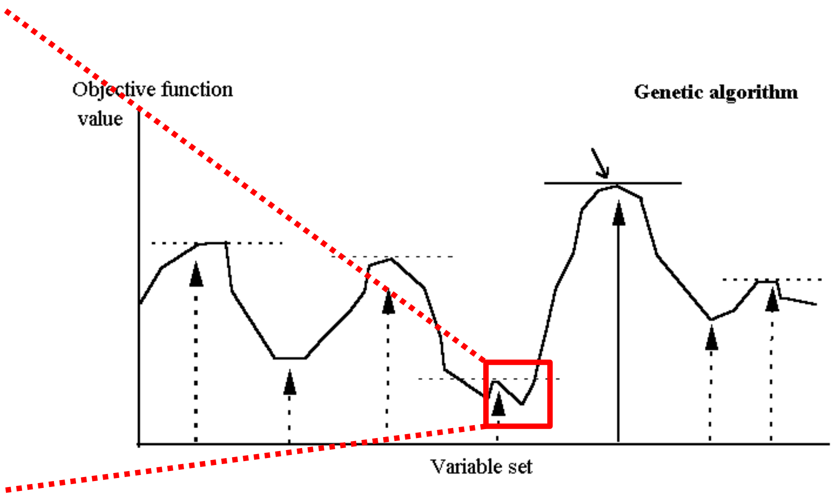


http://learning.cis.upenn.edu/cis520_fall2009/index.php?n=Lectures.Overfitting

Gradient Descent (and Hill Climbing)

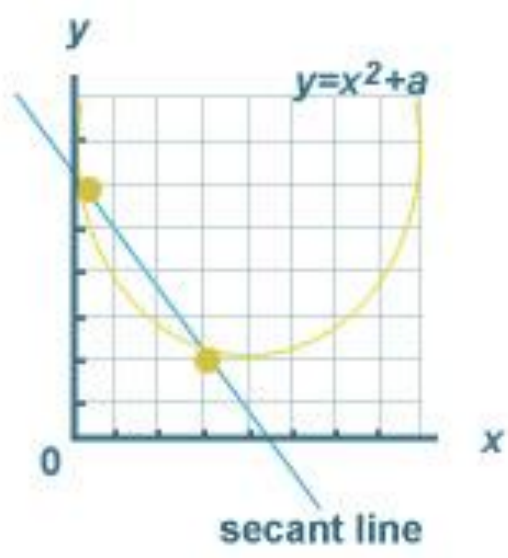
- Gradient Descent (and Hill Climbing) methods help us to find a local extremum (minimum or maximum value) of the objective function (*e.g.,* the True Error Curve in Neural Network models).

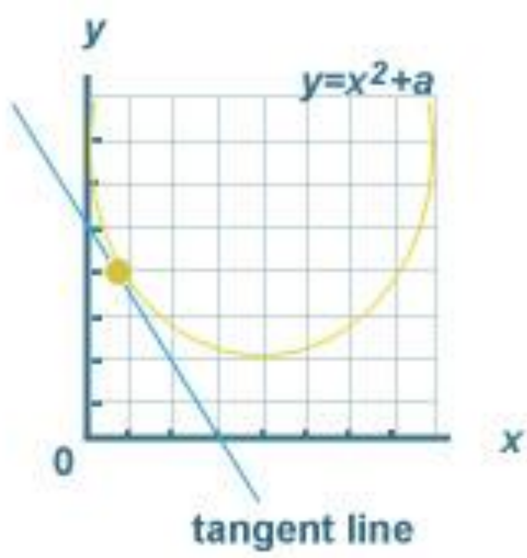


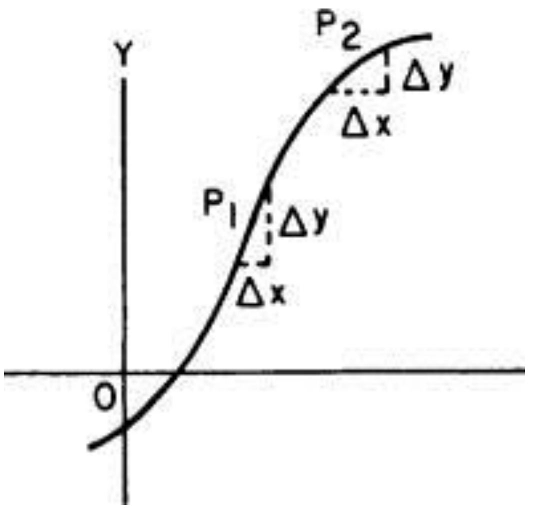


Learning Rate

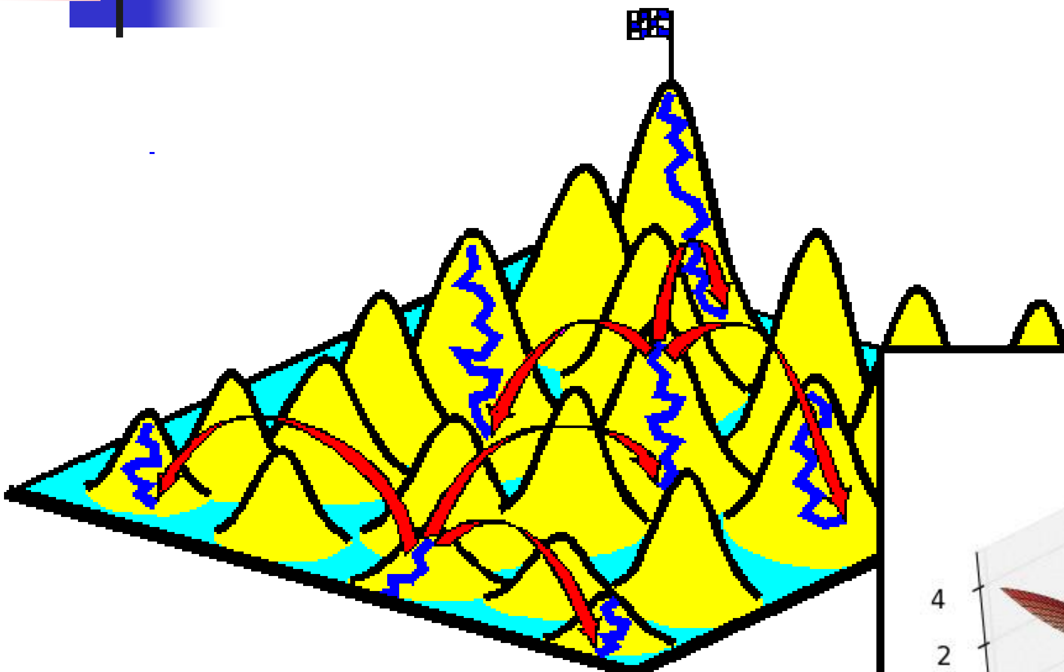
- Gradient Descent Learning Rate **Δy** must be small enough to avoid over-shooting the minimum, and large enough to avoid inefficient time-consuming search.



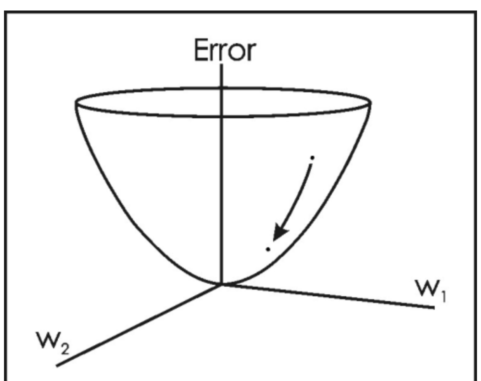


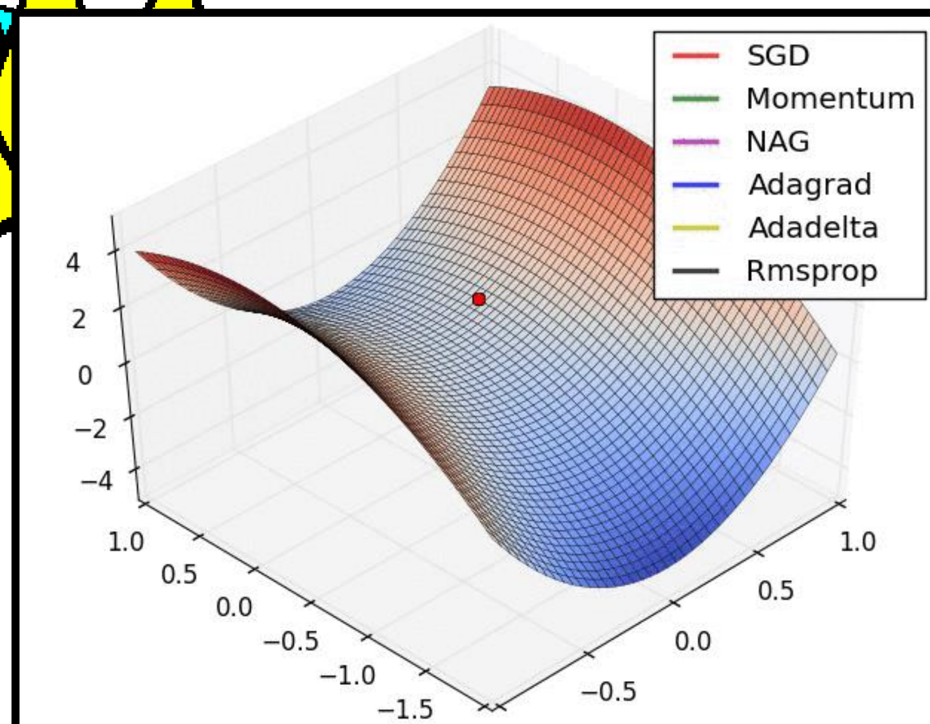


Most Real-world Objective Functions have complex topologies – need a better solution!



http://www.bionik.tu-berlin.de/institut/s2mulmo.html





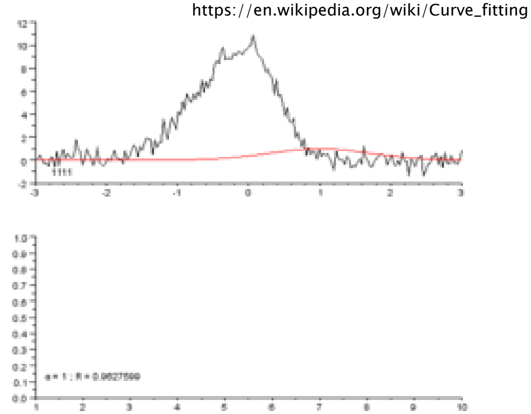
http://sebastianruder.com/optimizing-gradient-descent/

TensorFlow
https://www.tensorflow.org/

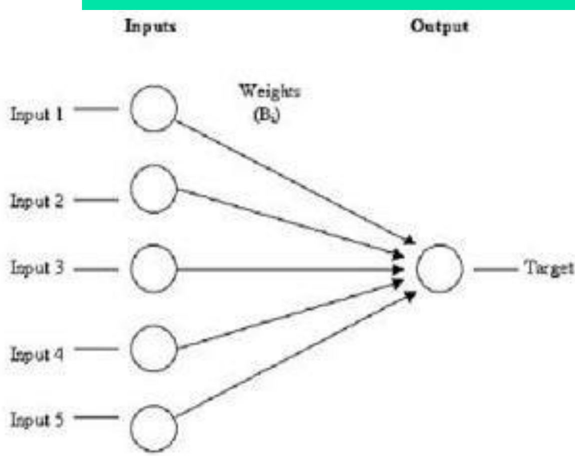
- TensorFlow applies gradient descent in complex topologies
- Finds variations of model accuracy as a function of variations in the weights of different input features (= a Calculus of Variations for Data Science!)
- Then adjusts the weights (through back-propagation of errors) in the direction of descent (thereby reducing the model error, without overfitting)
- The corrections to weights are nonlinear (since the features in our model are only representative of the complex real-world explanatory variables)
- Nonlinear combinations of the features make the best models!

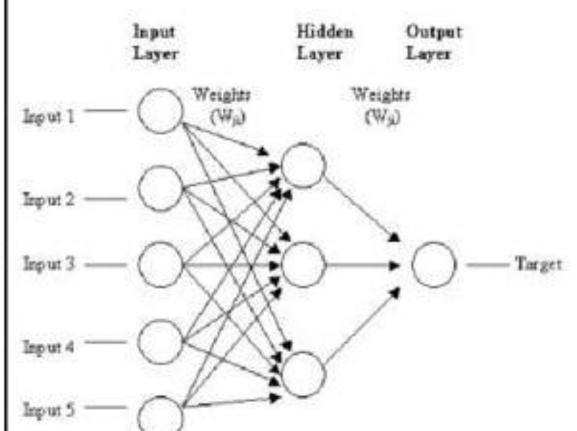
Regression fit flow

https://en.wikipedia.org/wiki/Curve_fitting



REGRESSION (linear) vs NEURAL NETWORK (nonlinear)





http://www.marcom-china.com/mva_neural.html

Summary 1 – some notes to myself (the novice)

(1) Gradient descent = typical ANN (NOAA wildfire example)

----

(2) Find point where slope of error curve d(ERROR)/d(model) is 0 in order to find model with minimum training error!

… this helps us to avoid Over-fitting (high variance) and Under-fitting (high bias).

----

(3) Carefully set the learning rate to avoid overshooting and to avoid inefficient learning rate.

----

(4) d(ERROR)/d(model) is a tensor, which is dependent on the coordinate system (feature space).

⇒ Modify weights on different terms/features in the data set to minimize model error E, using the tensor terms:

⇒ Partial derivatives: dE/dw_1, d^2E/dw_1dw_2, $d^3E/dw_1dw_2dw_3$

⇒ …that includes lots of cross-terms (= **the TENSOR**!)

... here, w_1 w_2 w_3 etc. are the weights on different features.

Summary 2 – some notes to myself (the novice)

(1) Math Examples: (a) The planar geometric shape that contains the most contained area for a fixed perimeter is a circle. (b) For a fixed volume, the object with the smallest surface area is a sphere. (c) The curve that contains the most area under the curve for a fixed length is a circular arc.

----

(2) Physics Example: Lagrangian **L=K–P** :=> variation(L) = δL = 0

… δ(Kinetic Energy) balances δ(Potential Energy) =>
yields the true trajectory

$$\delta \int_{t_1}^{t_2} L(q_i, \dot{q}_i)dt = 0$$

… Principle of Least Action minimizes a functional =
"sum of all Lagrangian variations δL along the true trajectory" = 0
… under a constant energy constraint: E = **K+P** = constant.

----

(3) HOT LANES Example (Variable Toll Lanes on Congested Highways):

… maximize revenue (slope=0 of a complex multi-variate revenue function),…

… under constant traffic volume constraint = moving the same number of cars from point A to point B.

All of these examples = A CALCULUS OF VARIATIONS !!

Summary 3 – final notes to myself (the novice)

1) **DEEP LEARNING** = builds machine learning models from data sets that have complex inter-dependent features …
   … Complex inputs: arrays / image segments / text patterns
2) **DEEP LEARNING** = minimizes error functional (maximizes accuracy) in a complex topology …
   … under the constraint that the inputs (the Training Data Labels) are fixed constants!
   … and that is Calculus of Variations!
3) **TensorFlow** = uses the TENSORS (partial derivatives) of the weight variations to back-propagate the corrections to the features' weights, to find the minimum-error model …
   … while avoiding overfitting!