Beginner’s Club: Machine Learning

Abe Hmiel

I used Andrew Ng’s (Stanford) Coursera course and Sebastian Raschka’s Machine Learning with Python textbook to assist with the preparation of this talk

https://www.coursera.org/course/ml
https://github.com/rasbt/python-machine-learning-book

What is machine learning?

What is machine learning?

Related to artificial intelligence?

What is machine learning?

Related to artificial intelligence?

Related to “big data”?

What is machine learning?

Related to artificial intelligence?

Related to “big data”?

Extremely buzzword-y?

What is machine learning?

Related to artificial intelligence?

Related to “big data”?

Extremely buzzword-y?

Will I get famous and/or people might pay me a lot of money if I can do this?

What is machine learning?

Related to artificial intelligence?

Related to “big data”?

Extremely buzzword-y?

Will I get famous and/or people might pay me a lot of money if I can do this?

Is it easy? Are there available resources to do this on my own?

What is machine learning?

Related to artificial intelligence? Yes- and also statistics, linear algebra, optimization

Related to “big data”? Yes, but moreover, field has grown due to recent computing power gains

Extremely buzzword-y? More like lotsa $5 words

Will I get famous and/or people might pay me a lot of money if I can do this? That’s what I hear

Is it easy? Are there available resources to do this on my own? The barrier to entry has never been lower, so it’s good to start now!

What is machine learning?

Let’s take focus on an overview of the machine learning (ML) field at large, and introduce basic concepts and terminology

We will draw attention to places where beginners can make inroads quickly

What is machine learning?

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

- Tom Mitchell

What is machine learning?

Example: playing checkers.

E = the experience of playing many games of checkers

T = the task of playing checkers.

P = the probability that the program will win the next game.

Where is machine learning used?

Database mining

- large datasets from growth and proliferation of web technologies, automation, ‘Internet of Things’ (IoT)

- e.g: web click data, advertising, medical records, engineering, finance

Where is machine learning used?

Autonomous applications:

-e.g., Autonomous helicopter, handwriting recognition, most of Natural Language Processing (NLP), Computer Vision, self-driving cars, etc.

Where is machine learning used?

Self-customizing programs

-e.g: recommendation systems: netflix, amazon

Where is machine learning used?

Understanding human learning

- Neuroscience, early childhood development, building real AI

Where is machine learning used?

Overall, a *general-purpose* technology that can affect an entire economy and possibly, dramatically alter society

Subtypes of machine learning

Supervised learning

- “right answers” have been given, algorithm is “trained” with respect to these features

Subtypes of machine learning

Supervised learning

- “right answers” have been given, algorithm is “trained” with respect to these features

Unsupervised learning

- algorithm must learn & differentiate structure existing between important and non-important features

Subtypes of machine learning

Supervised learning

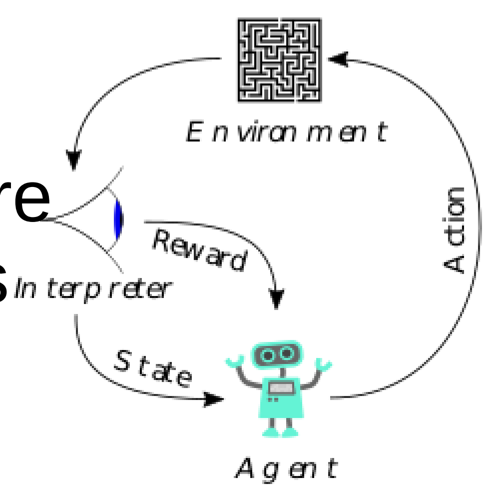
- “right answers” have been given, algorithm is “trained” with respect to these features

Unsupervised learning

- algorithm must learn & differentiate structure existing between important and non-important features; unlabeled data

Reinforcement learning

- inspired by behaviorist psychology, software agents take actions in an environment so as to maximize a notion of cumulative reward.



Types of problems: Supervised Learning

Classification

- predict membership among a small number of classes
- is a particular email spam or not?
- predict someone’s favorite MLB team

Types of problems: Supervised Learning

Classification

- predict membership among a small number of classes

- is a particular email spam or not?

- predict someone’s favorite MLB team

Regression

- prediction of continuous-valued outcome

- predict $ value of a house given a set of features

Types of problems: Unsupervised Learning

Clustering

- grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups

- community recognition in social networks

Types of problems: Unsupervised Learning

Clustering

- grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups

- community recognition in social networks

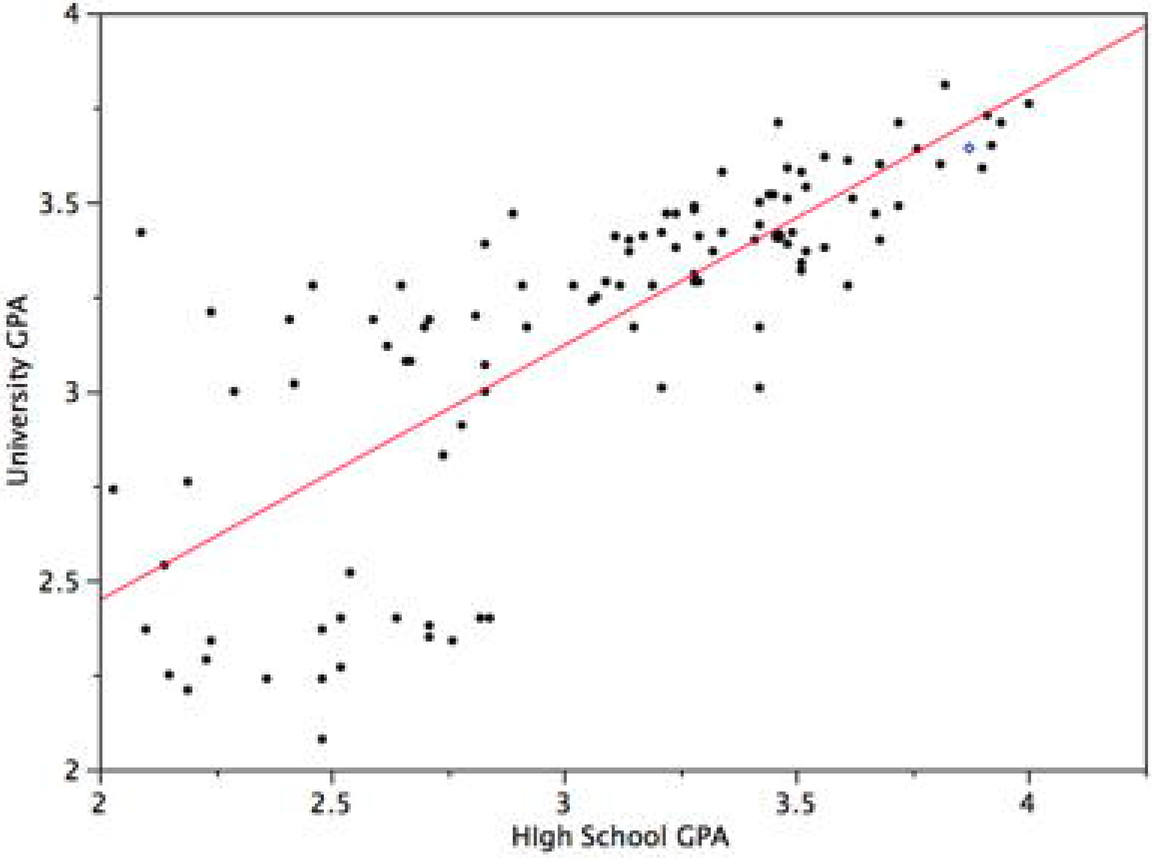
Dimensionality reduction

- given a large number of features n, cast data in terms of k new features in such a way that k < n

- exploratory analysis; aim is to use less computational resources/simplify & discard features

Defining an ML problem: linear regression in one variable

"Trivial example": not really a good example of an ML problem in the wild, but illustrates what we need to tackle more complicated problems



Defining an ML problem: linear regression in one variable

Hypothesis function: represents a mapping of features to a prediction

$$h_\theta(x_i) = \theta_1 x_i + \theta_0$$

Cost (loss) function: maps an event or values of one or more variables onto a real number intuitively representing some "cost" associated with the event.

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(\hat{y}_i - y_i\right)^2 = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x_i) - y_i\right)^2$$

Linear regression: hypothesis and cost function

$$h_\theta(x_i) = \theta_1 x_i + \theta_0$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(\hat{y}_i - y_i\right)^2 = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x_i) - y_i\right)^2$$

Have a notion of ‘cost’ for any choice of θ_0, θ_1

Objective is to find the minimum value of the cost function

Do this by calculating the gradient of the cost function and update θ_0, θ_1, and continue to do so until we arrive at the minimum

Gradient of cost function

Simple gradient descent algorithm (repeat until convergence):

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

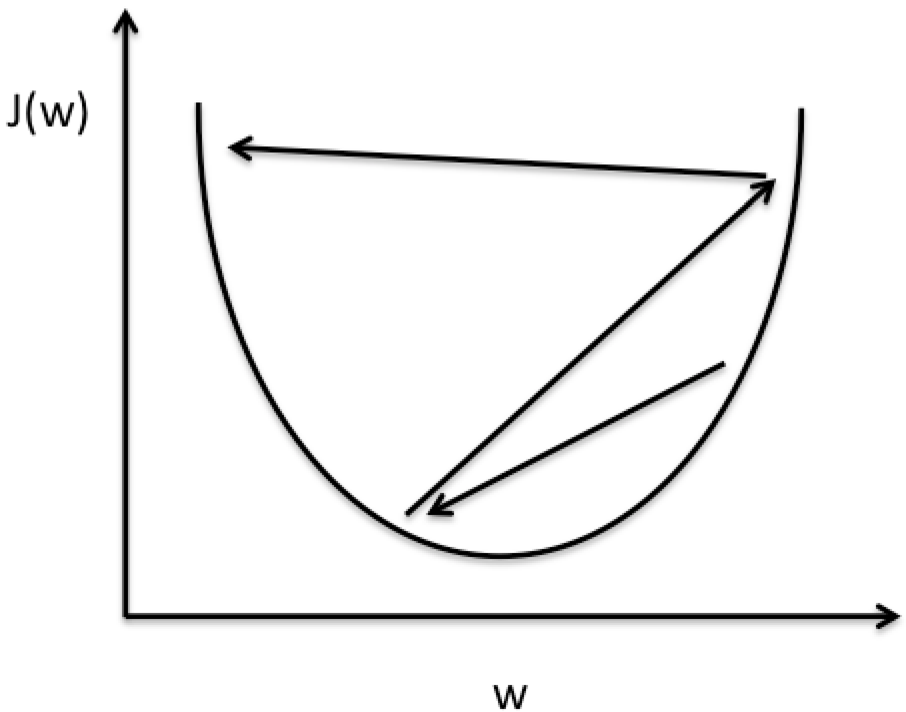
$j=0,1$ represents feature index number

α is a parameter called learning rate

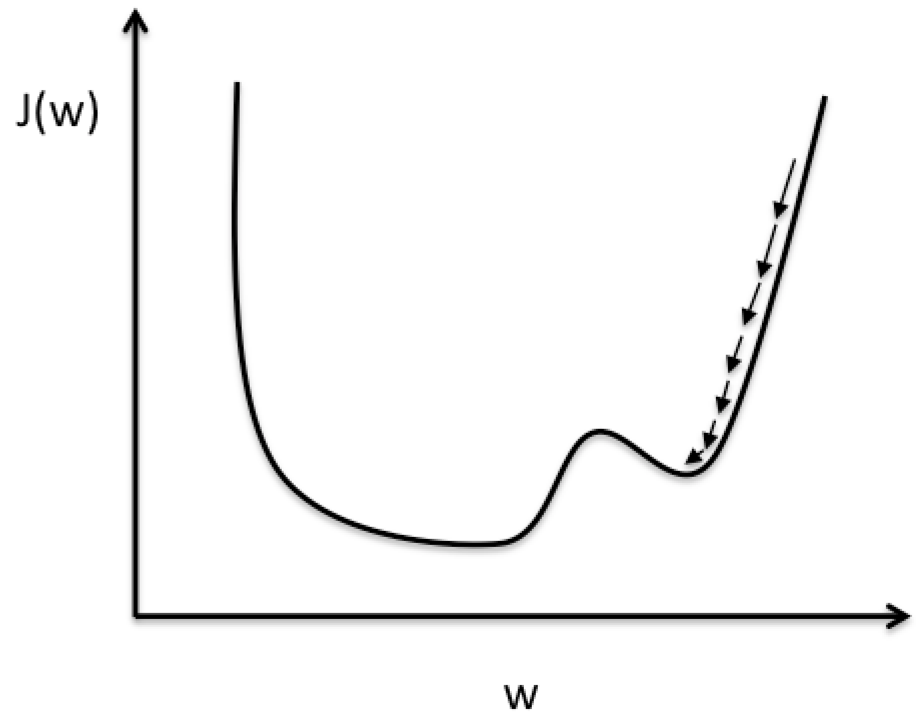
Simultaneously update θ_j's at each step

Learning rate

Should be chosen judiciously:



Large learning rate: Overshooting.



Small learning rate: Many iterations until convergence and trapping in local minima.

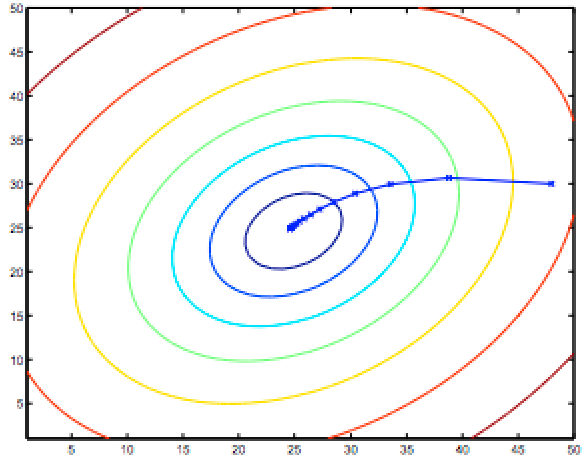
Full algorithm: linear regression in one variable

repeat until convergence: {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left((h_\theta(x_i) - y_i)x_i\right)$$

}



End while loop when differences between θ_j's from one step to the next falls below a certain tolerance

Logistic regression: hypothesis and cost function

For binary classification, only have two possibilities for each class: yes (1) or no (0)

Workshopping a cost function:

$Cost(h\theta(x),y)=0$ if $h\theta(x)=y$

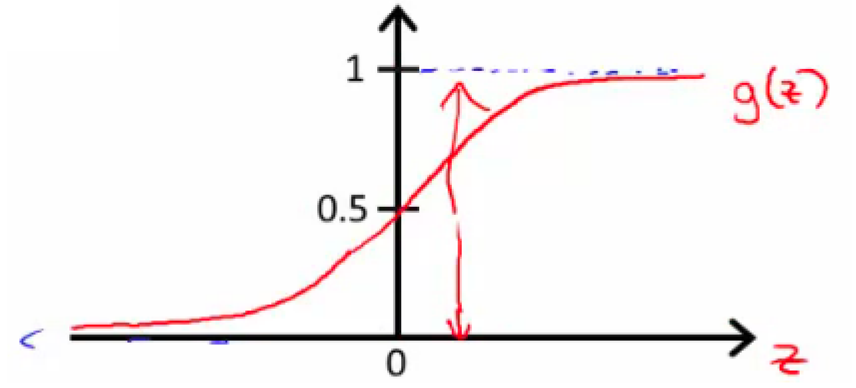
$Cost(h\theta(x),y) \rightarrow \infty$ if $y=0$ and $h\theta(x) \rightarrow 1$

$Cost(h\theta(x),y) \rightarrow \infty$ if $y=1$ and $h\theta(x) \rightarrow 0$

Logistic regression: hypothesis and cost function

Hypothesis (sigmoid function):

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$



Cost function:

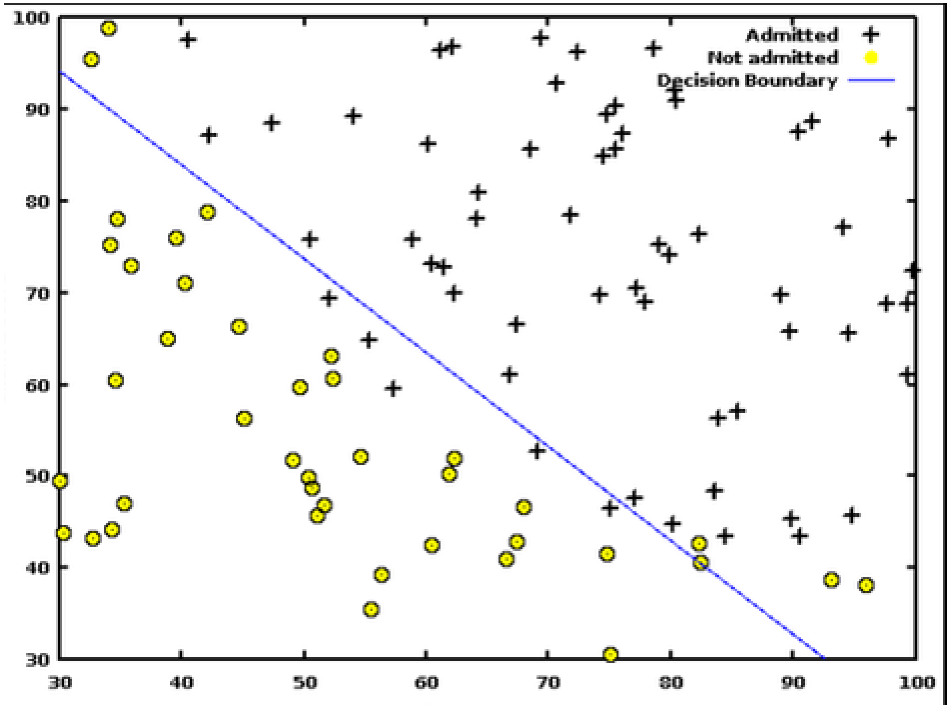
$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(y-1)\log(1-h_\theta(x)) - y\log(h_\theta(x))$$

Decision boundary

Decision boundary refers to the boundary between classifying a data point as a “yes” or “no,” for example:

$h_\theta(x) \geq 0.5 \rightarrow y=1$

$h_\theta(x) < 0.5 \rightarrow y=0$:



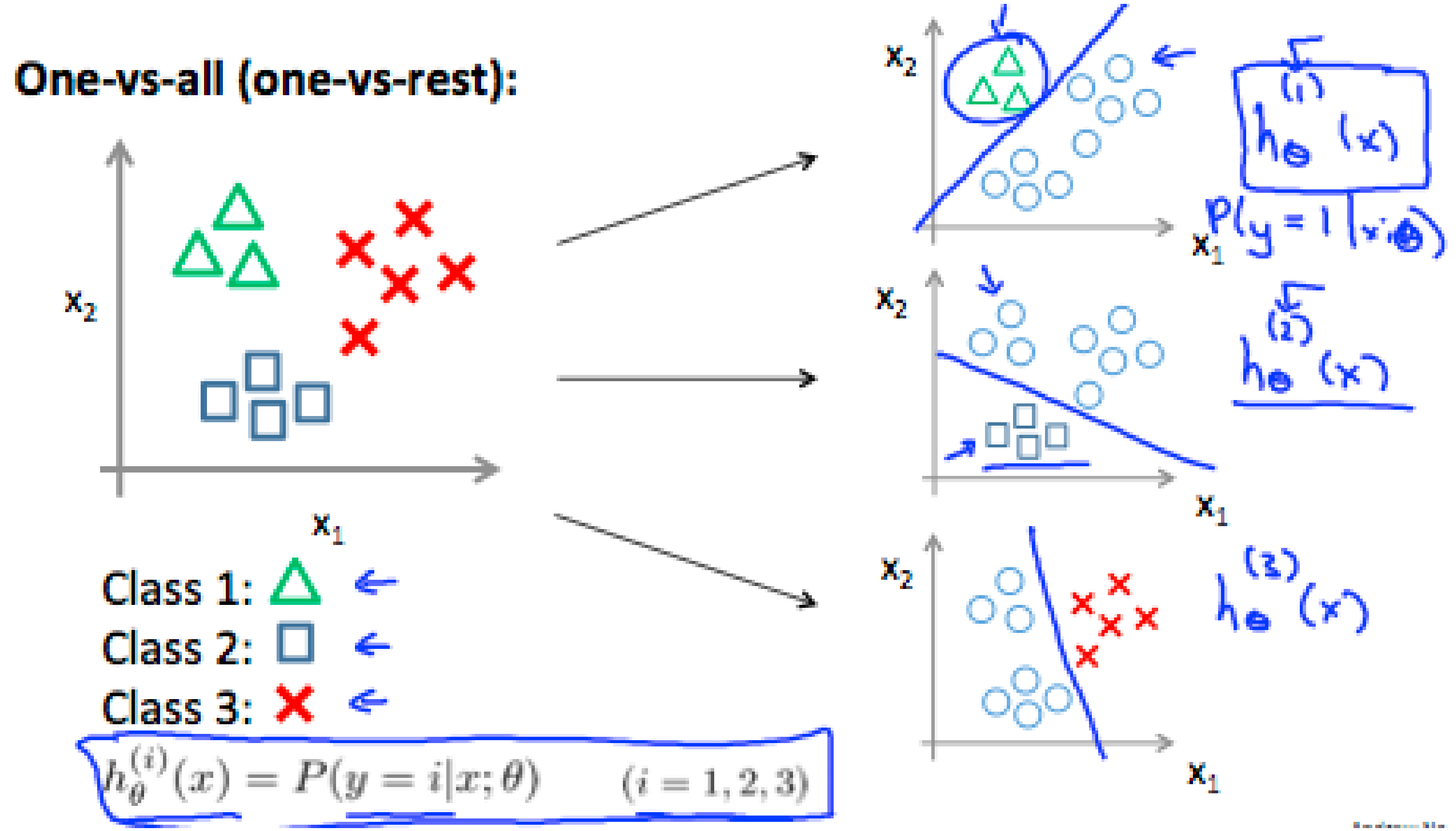
Did the classifier perform well?

Precision: How many items we identified were relevant?

Recall: How many relevant items did we identify?

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Multiclass classification



Train a logistic regression classifier for each class to predict the probability that y = i

To make a prediction on a new x, pick the class that maximizes $h_\theta(x)$

Scaling up simple problems

Easily generalized to problems with several (or hundreds!) of features

Can be used with various formulations of cost functions and methods to perform gradient descent

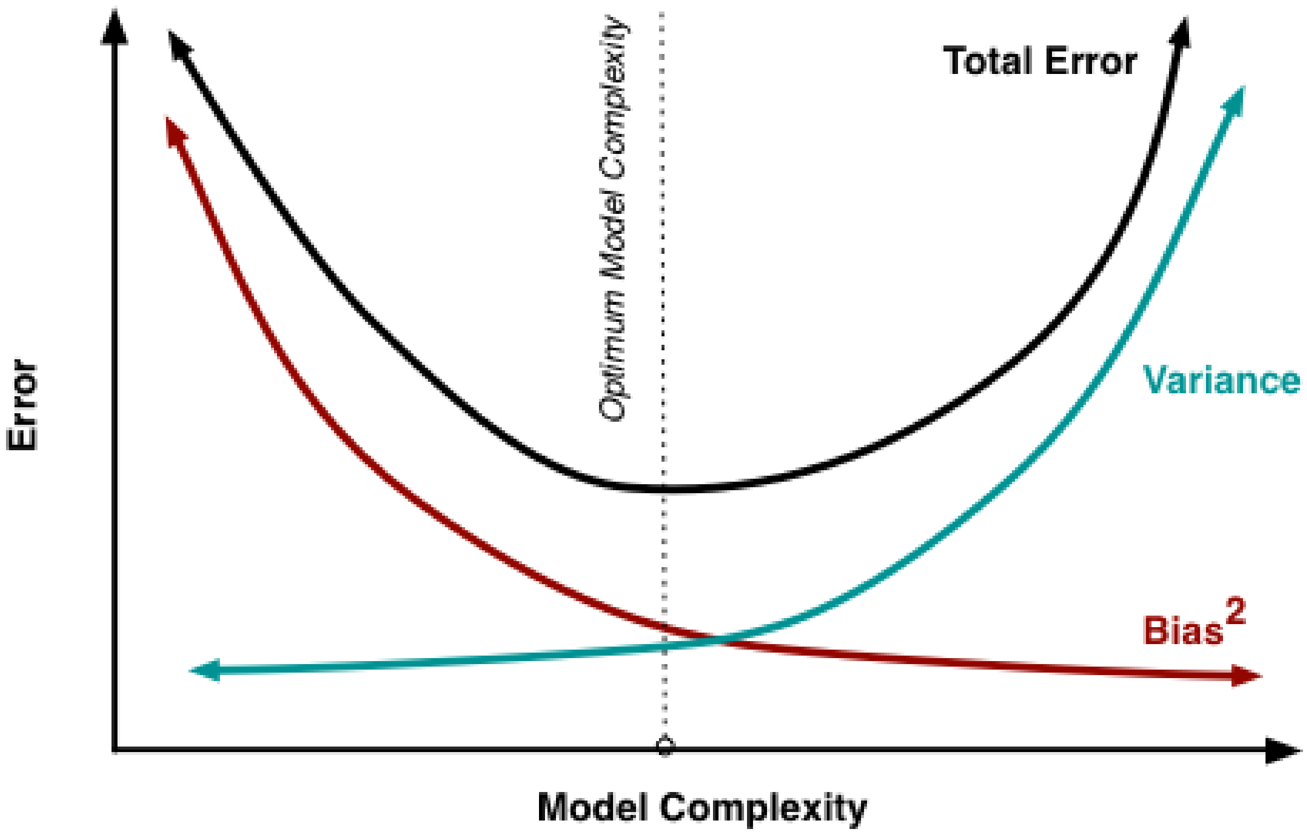
Feature scaling

Good performance and fast convergence is achieved when features are about the same size:

$$x_i := \frac{x_i - \bar{x}_i}{x_i^{max} - x_i^{min}}$$

Bias vs. Variance

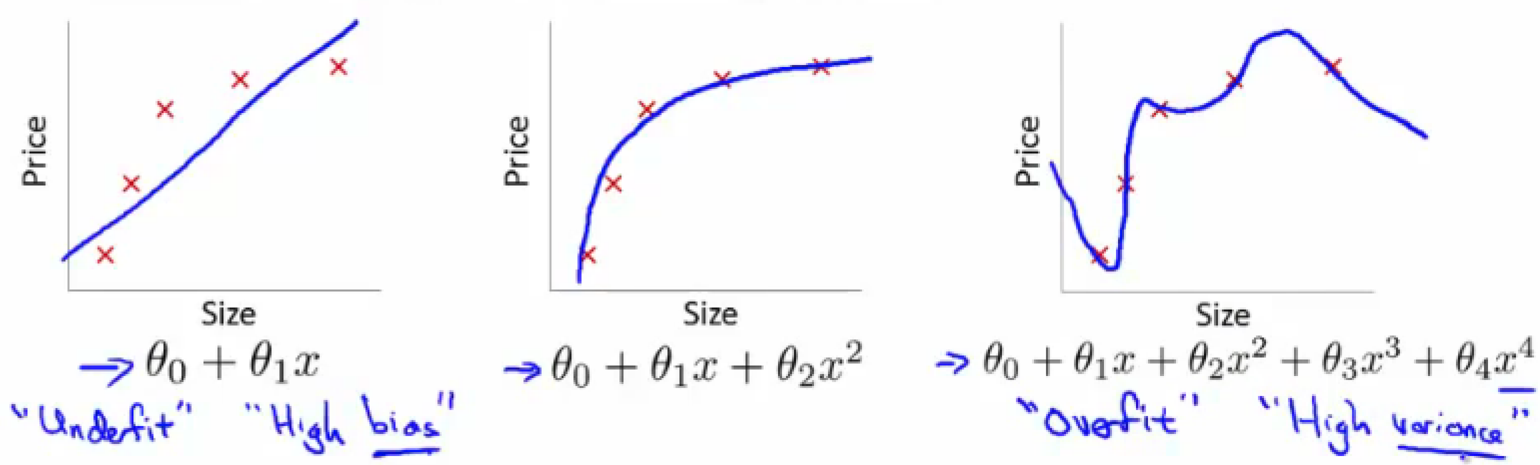
Tradeoff between minimizing individual sources of error in a model:



Regularization

Introduce additional penalty to features (e.g. high polynomial degree) to avoid overfitting the data:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta x^{(i)} - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{i=1}^{m}\theta_j^2$$



Hyperparameter tuning

For a particular model, there may be a large number of tunable parameters, e.g. # of trees in a decision forest, #

Run several trials of an algorithm with different parameters, testing one or two parameter choices at a time, focus on best-performers and then don’t think about it too much

Grid search vs. Bayesian approach

Little universal advice- go with what works on a case-by-case basis

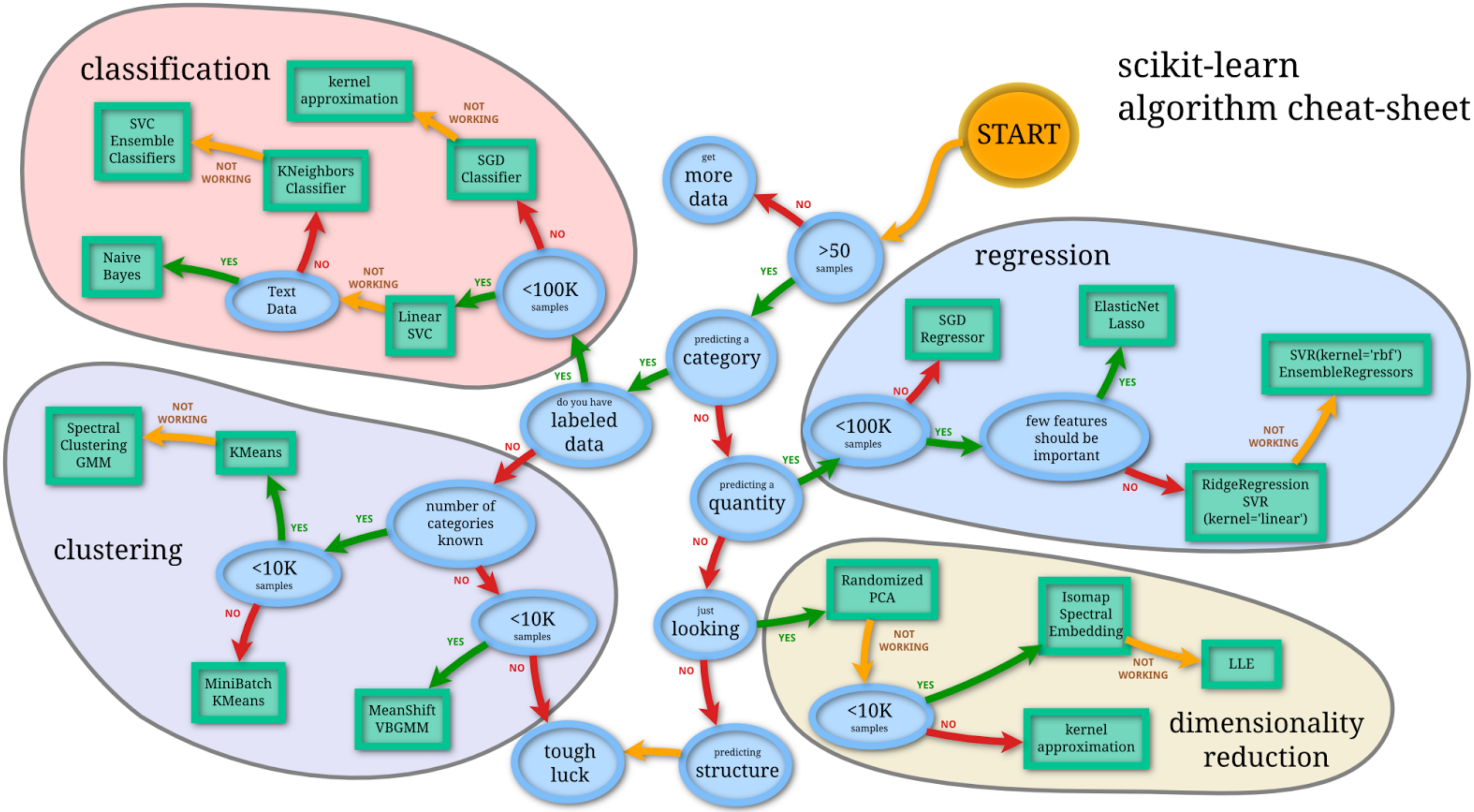
Data cleaning/pipelining

Before you can really do anything, need to get data into the right “shape”

Preprocessing steps can include removing/filling in missing values, converting strings to numerics, various operations with dates, etc.

Sometimes, this is a considerable amount of work itself, and you haven’t even defined a hypothesis or built a model yet!

Need knowledge of when to use each algorithm; lots to choose from!



Evaluating a hypothesis

Split data (randomly) into training (~70%) and test (~30%) sets:

- learn θ and minimize training cost function J using only training data

- compute cost function J for test data

- compare error in test results (misclassification, results of regression) – can help you answer the question, “is this a good model, after all?”

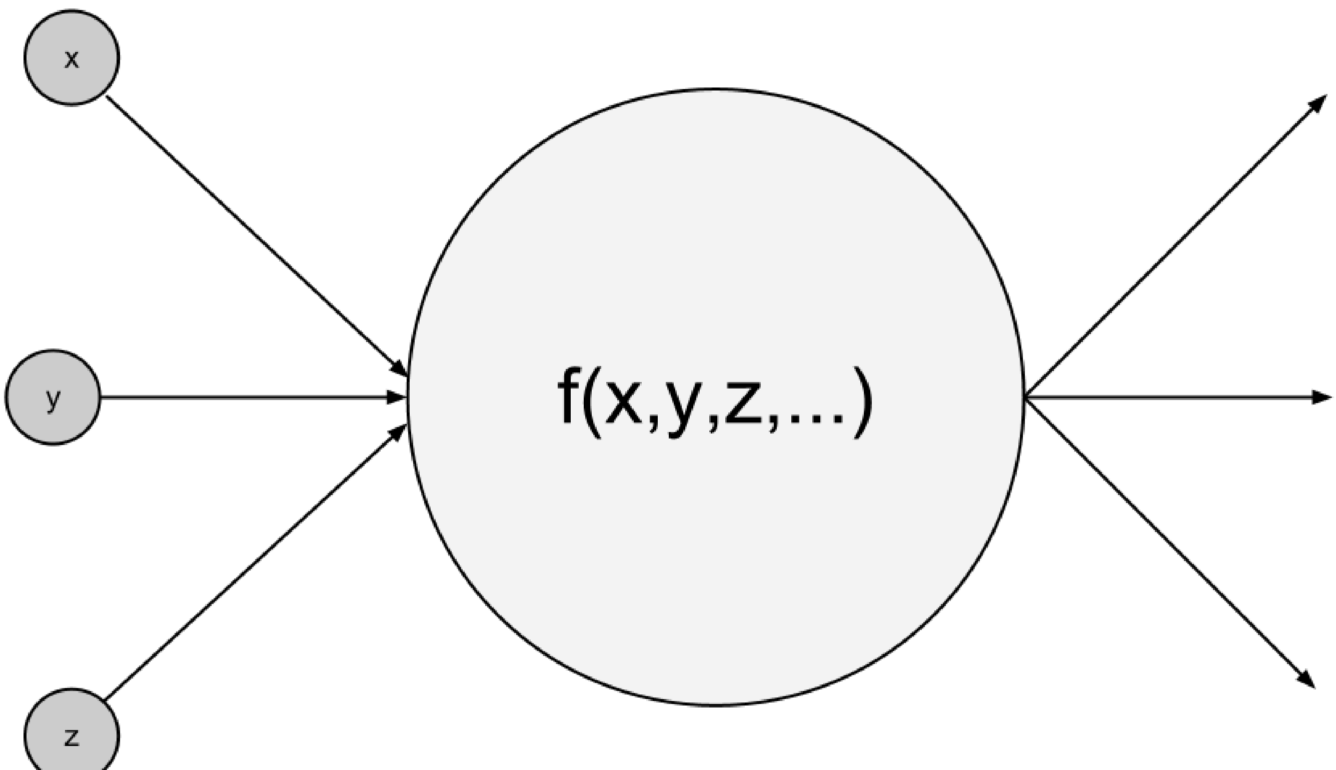
Evaluating a hypothesis

Troubleshoot for errors in predictions by:

- getting more training examples fix high variance
- trying smaller sets of features fix high variance
- trying to add additional features fix high bias
- trying polynomial features fix high bias
- increasing λ fix high variance
- decreasing λ fix high bias

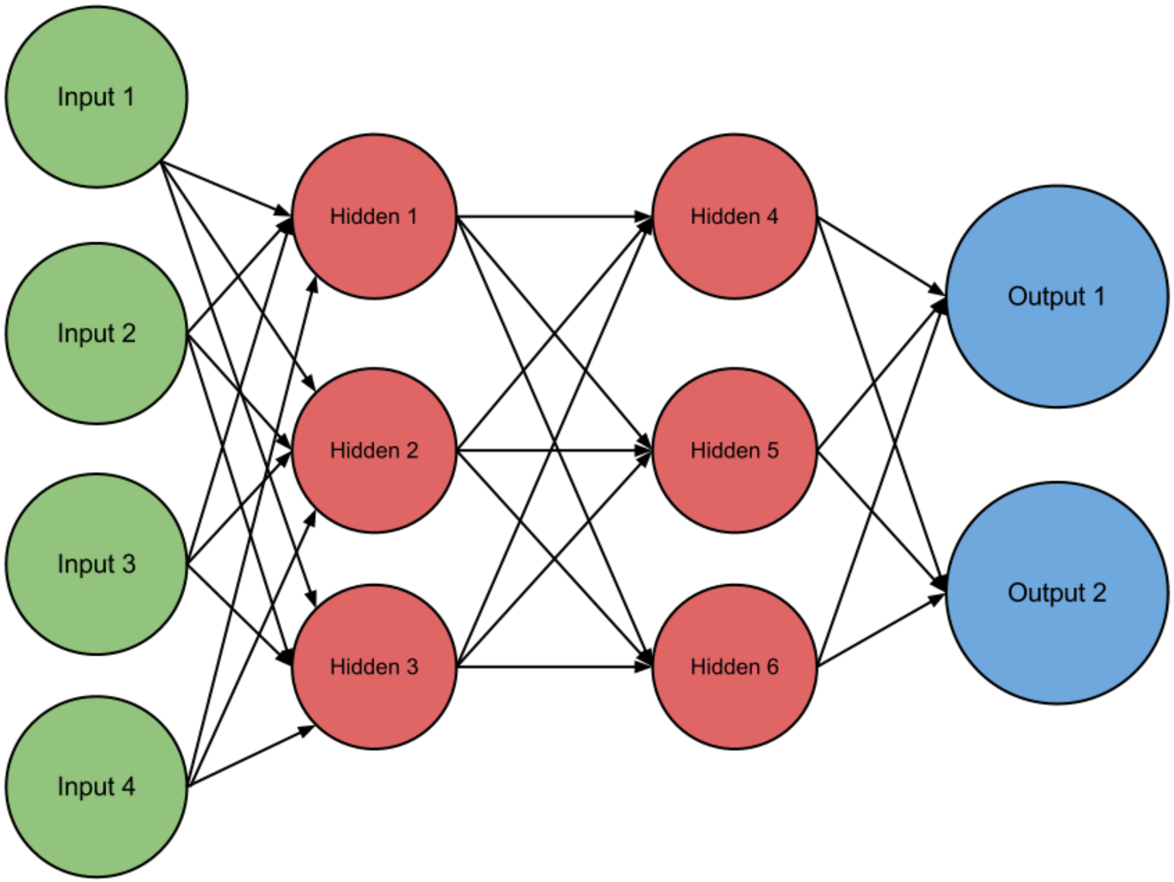
Neural Networks

Allows representation of highly complex, nonlinear relationships



Neural Networks

Allows representation of highly complex, nonlinear relationships

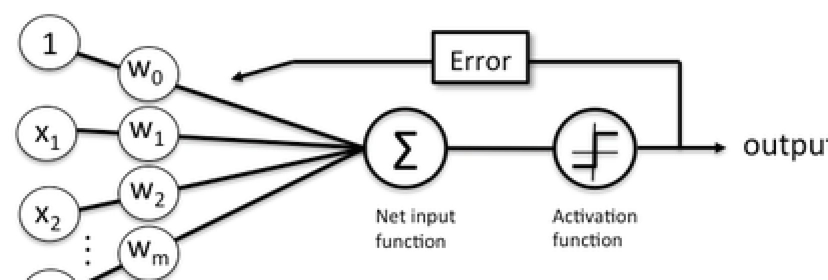


Activation function and weights

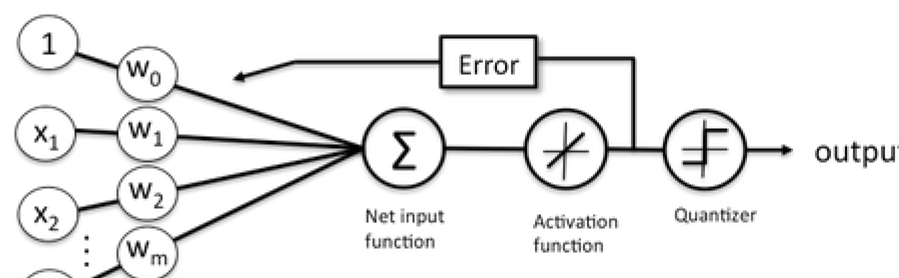
Each connection between neurons has a weight and bias that is tuned through the learning algorithm

Each neuron has an activation function, a function that maps a continuously-valued function to either 0 (off) or 1 (on)

Weights, biases computed by backpropagation algorithm

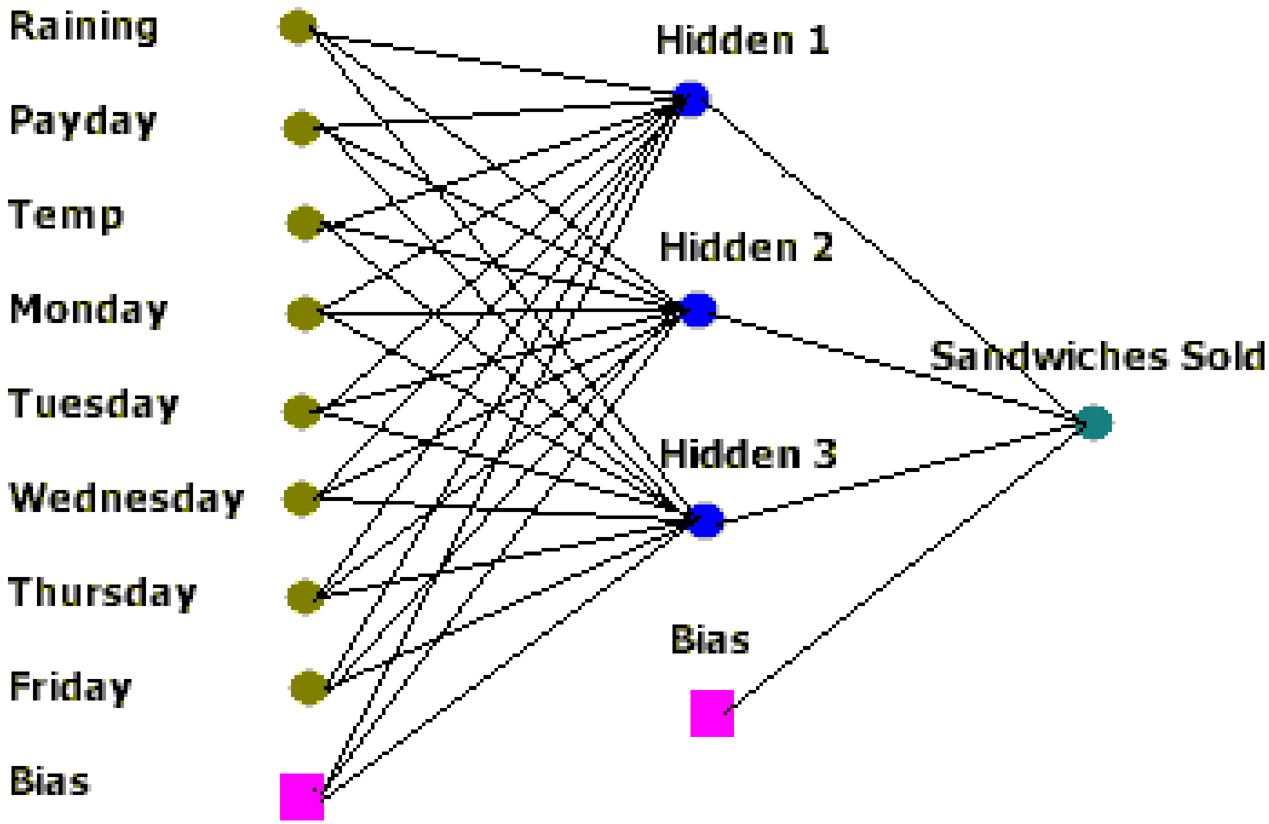


Perceptron rule.



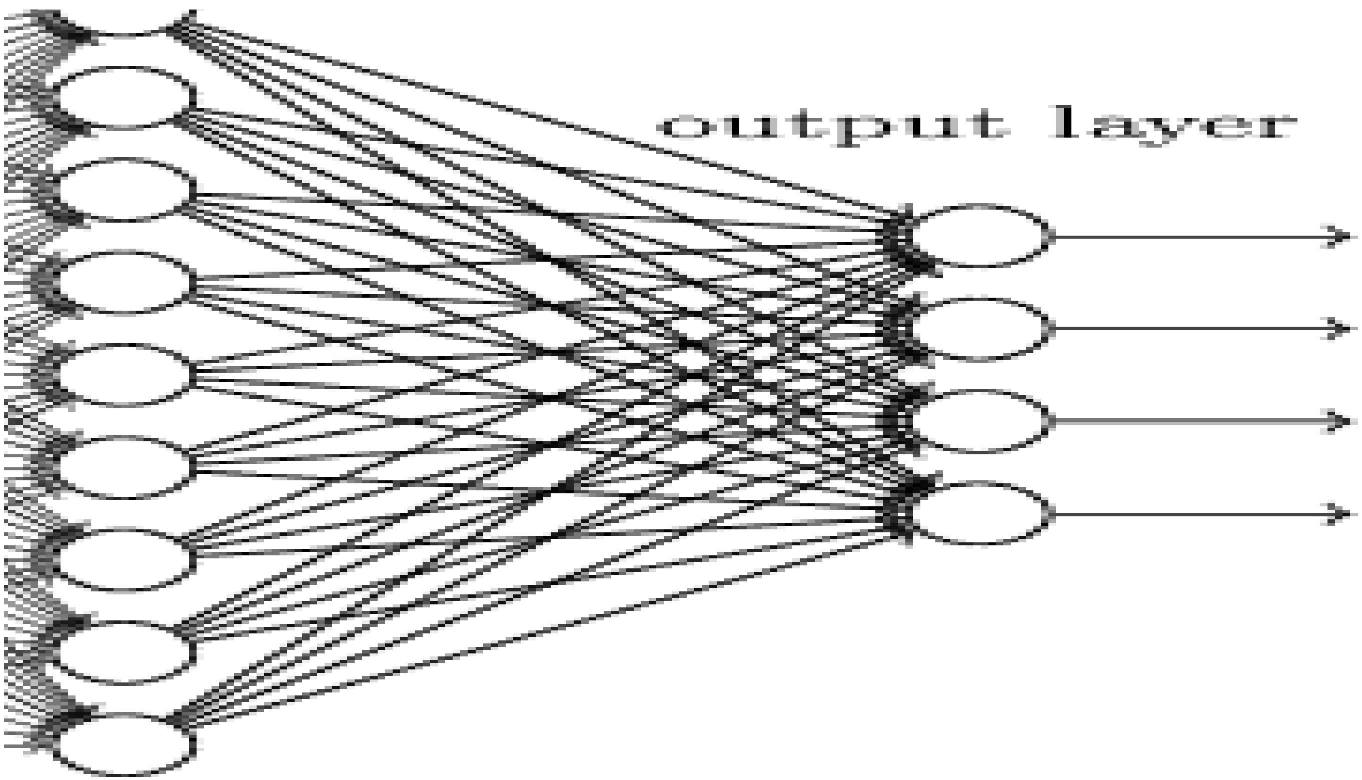
Adaline.

NN with single hidden layer

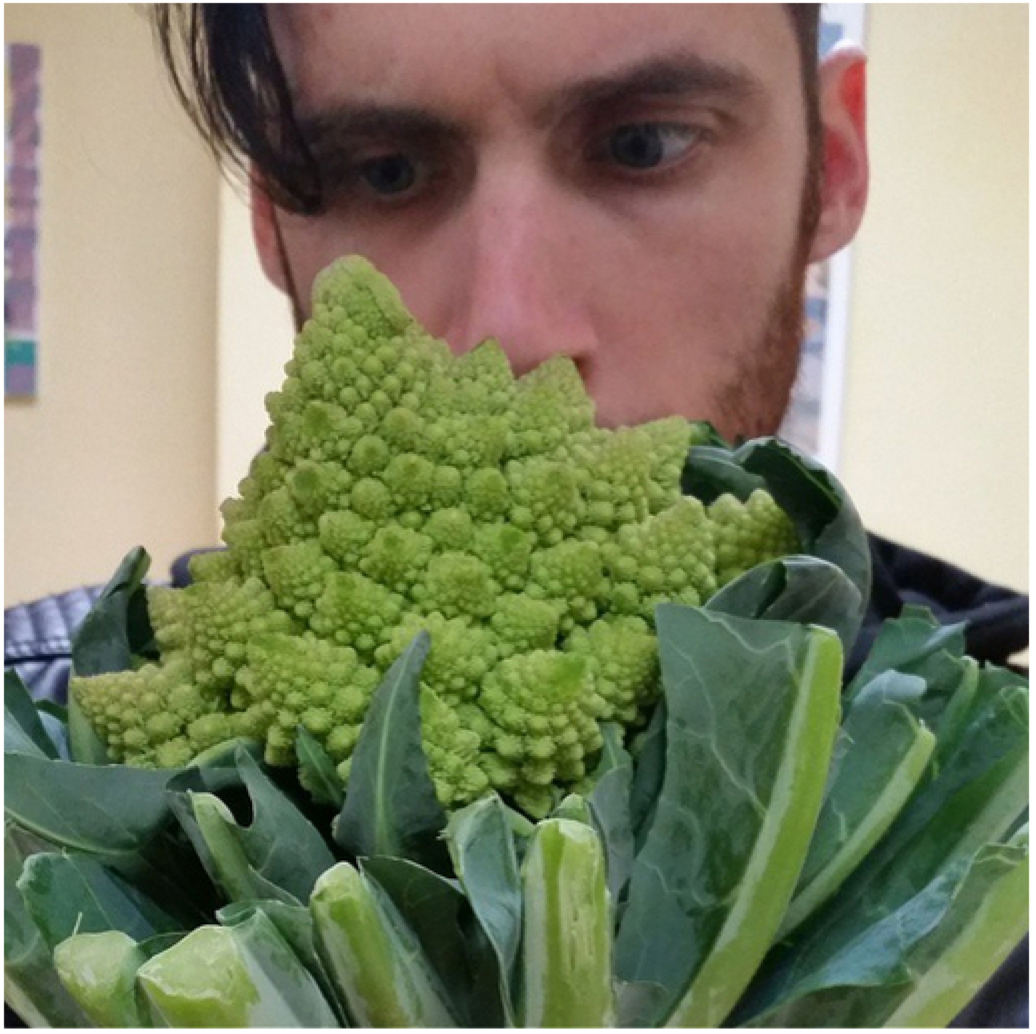


Deep NN

Multiple hidden layers



tfw u get 2 deep



tfw u get 2 deep


tfw u get 2 deep

Non-NN ML frameworks/libraries

Apache Spark MLlib: {Python, Java, Scala, R}

Scikit-learn: Python

klaR, kernlab, e1071, rpart, caret, igraph: R

Shogun: {C++, Java, Python, C#, Ruby, R, Lua, Octave, and Matlab}

NN ML frameworks/libraries

Caffe

Tensorflow

Torch

Microsoft CNTK

MXNet

Deeplearning4j

Keras, Theano (truly, a mathematical expression compiler)

+Lots more

(Many are multi-platform and almost all can make use of GPU)