Introduction to Machine Learning (ML)

邓仰东
清华大学软件学院

提纲

1. Basic Concepts
2. Types of ML Problems
3. Optimization
4. Taxonomy of Algorithms

What Is Machine Learning

- “field of study that gives computers the ability to learn without being explicitly programmed”

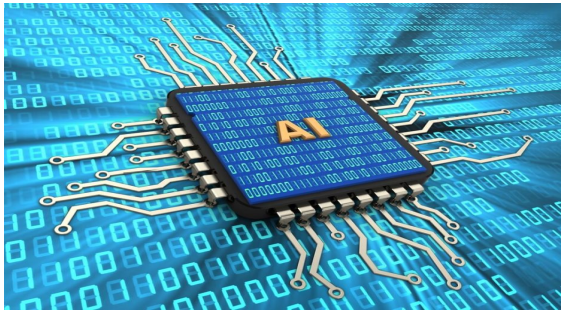
Arthur Samuel (1959)

- The capacity of a computer to learn from experience, i.e. to modify its processing on the basis of newly acquired information

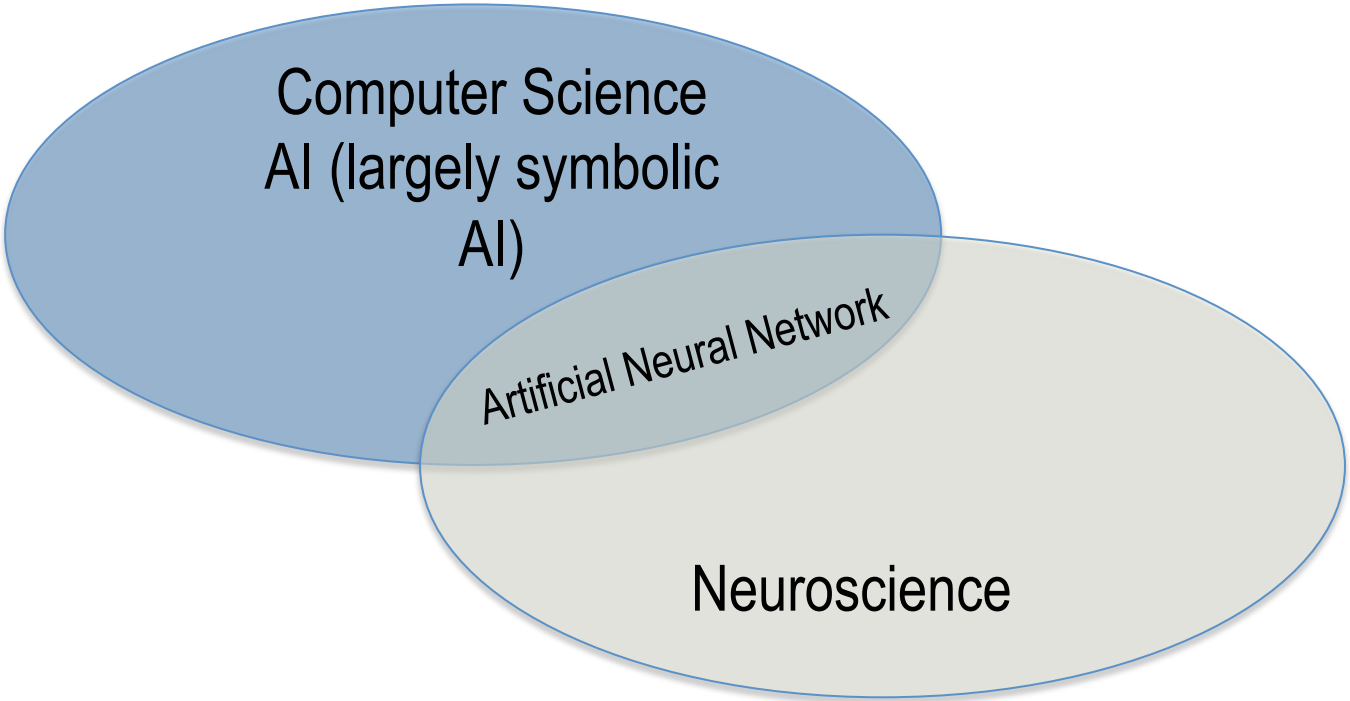
Oxford dictionaries

What Is Machine Learning

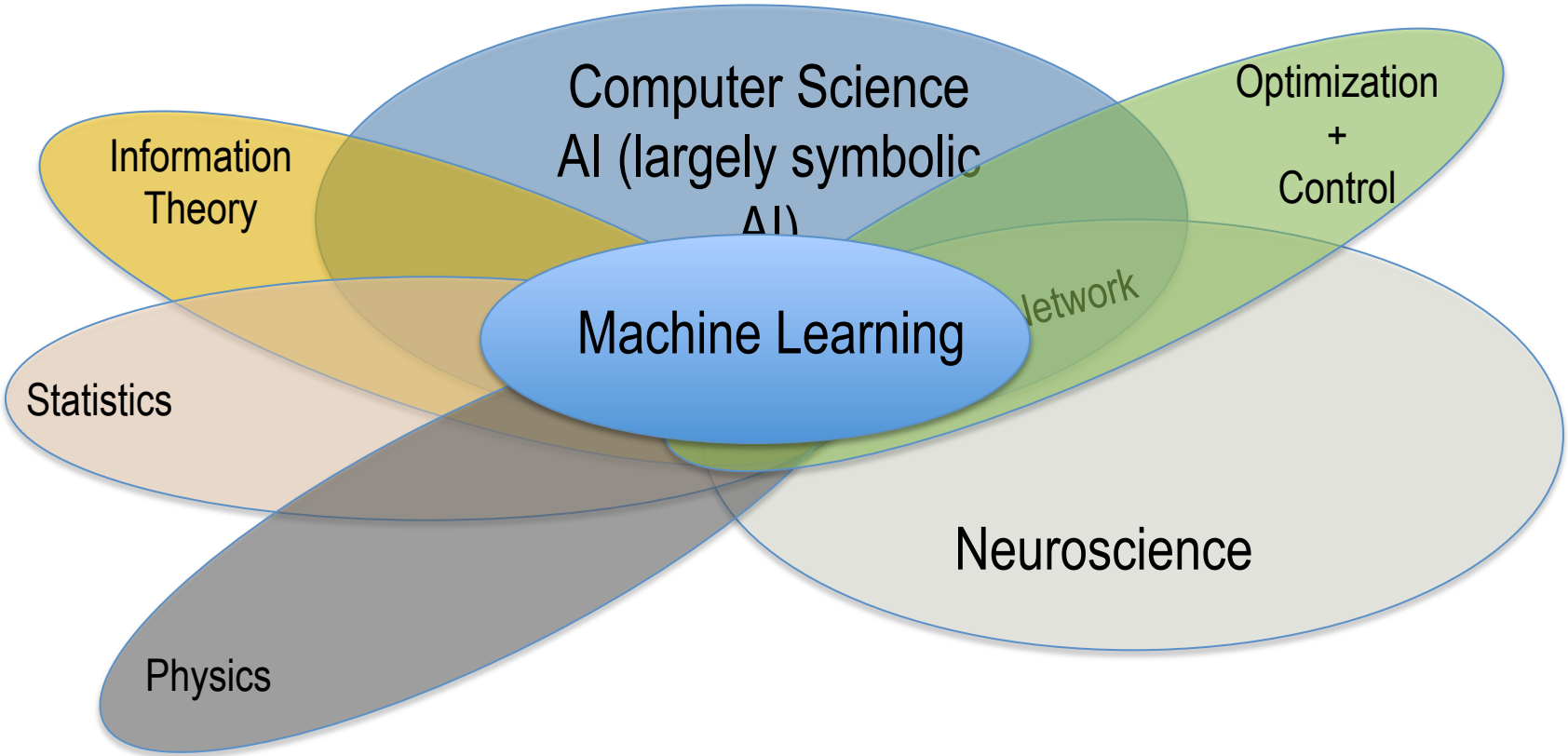
- Explores the study and construction of algorithms that can learn from and make predictions on data
- Evolved from the ambitious goal of Artificial Intelligence
  - Two historically opposed approaches to AI
    - Neuroscience inspired: Neural nets learning from examples
    - Classical symbolic AI: Logic reasoning, symbolic computations (human specified rules), Bayesian, …
  - Founding project: The Perceptron (Frank Rosenblatt 1957)
    - First artificial neuron learning form examples



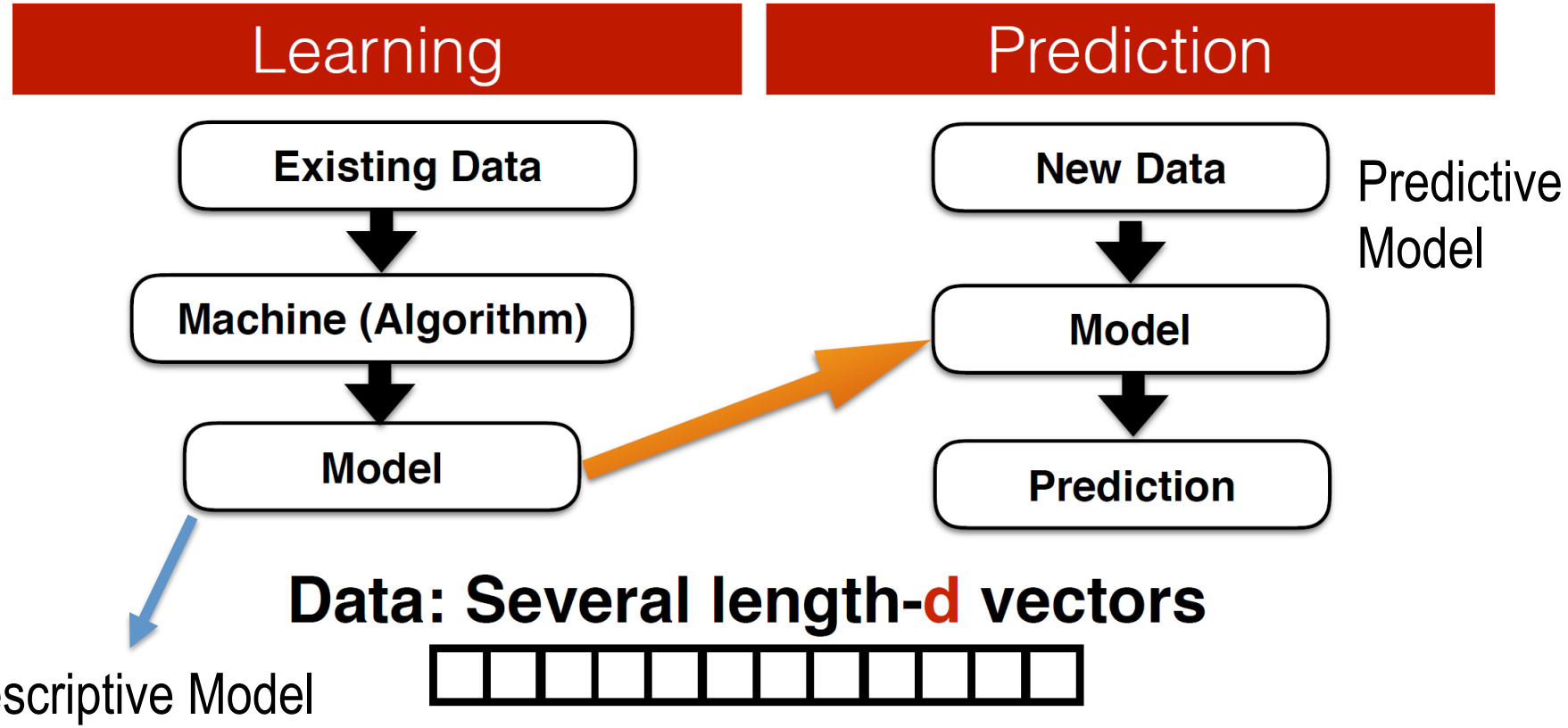
Early AI


Computer Science
AI (largely symbolic
AI)
Artificial Neural Network
Neuroscience

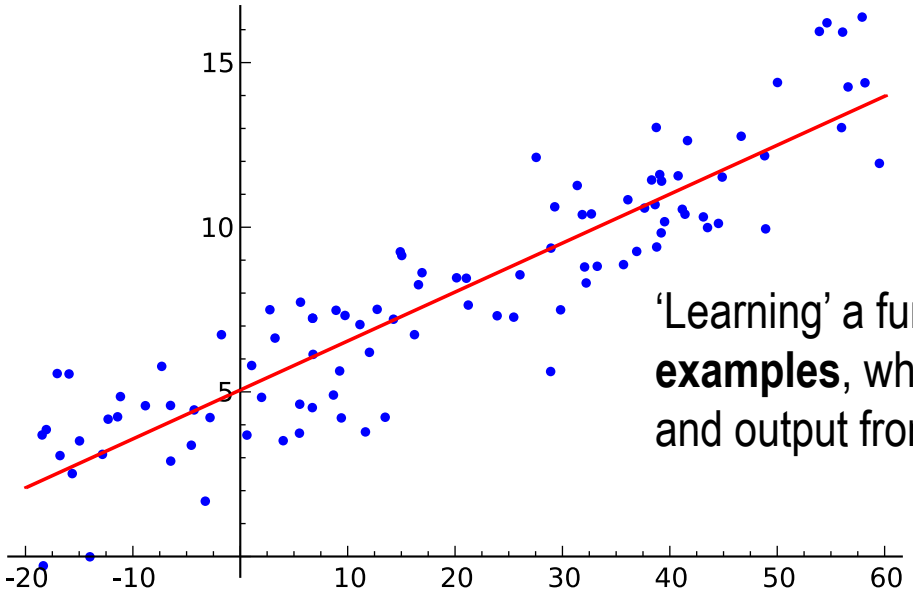
Modern AI


Computer Science
AI (largely symbolic
AI)
Information
Theory
Optimization
+
Control
Machine Learning
letwork
Statistics
Neuroscience
Physics

What Is Machine Learning?



A Simple Machine Learning Problem

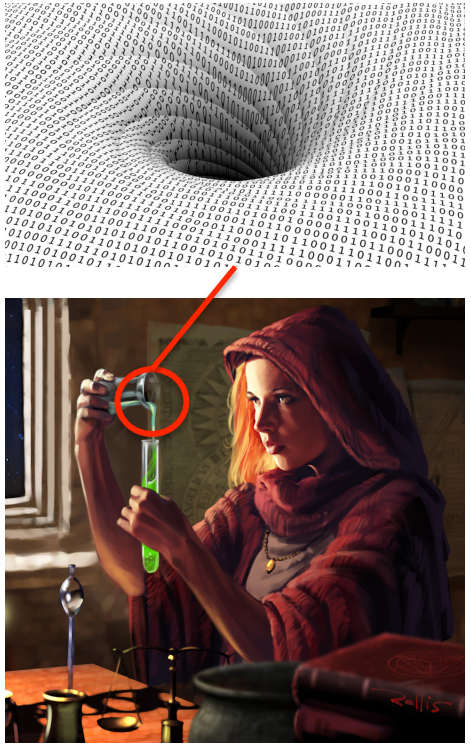


Linear regression (Source)

'Learning' a function given a **training set** of **examples**, where each example is a pair of an input and output from the function

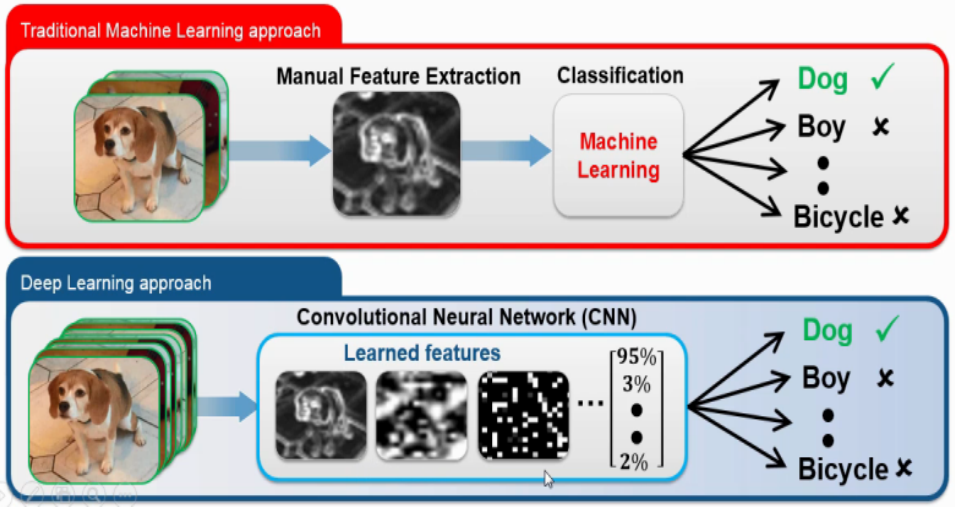
Data: The Key Ingredient of Machine Learning

- Collected from nature or industrial processes
  - Stored in many forms (and formats...)
    - Structured, unstructured
    - Occasionally clean, usually messy, ...
- In ML we like to view data as a **list of examples**
  - Ideally **many** examples of the same nature.
  - Preferably with each example a vector of numbers



What Is Deep Learning

- The modern reincarnation of Artificial Neural Networks from the 1980s and 90s.
- A collection of simple trainable mathematical units, which collaborate to compute a complicated function
- Compatible with supervised, unsupervised, and reinforcement learning



Deep Neural Network Can Be Impressive

Both recognized as a
“meal”

Deep Neural Network Can Be Impressive

Human: Three different types of pizza on top of a stove.

Model sample 1: Two pizzas sitting on top of a stove top oven.

Model sample 2: A pizza sitting on top of a pan on top of a stove.

提纲

1. Basic Concepts

2. Types of ML Problems

3. Optimization

4. Taxonomy of Algorithms

General Machine Learning Approaches

- Supervised learning: Leaning by labeled examples
  - e.g. Image classification with convolutional neural networks
  - Amazingly effective if you have lots of labeled examples
- Reinforcement Learning: Feedback right/wrong
  - e.g. Learning to play chess by winning or loss
  - Works well in some domains, becoming more important
- Unsupervised learning: Discovering patters
  - e.g. Data clustering
  - Difficult in practice, but useful if you lack labeled examples

Supervised Learning

- Training experience: a set of labeled examples of the form

  $$\langle x_1\ x_2\ \ldots x_n,\ y_i \rangle$$

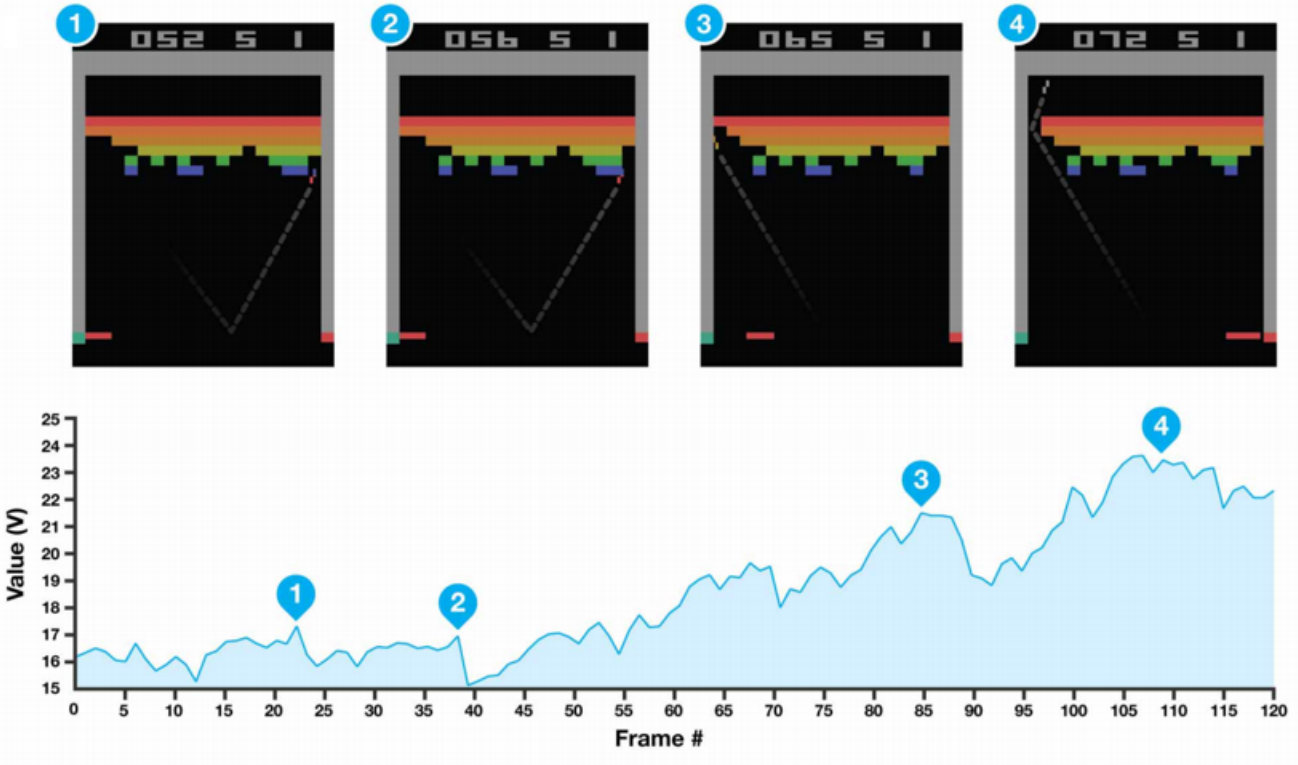
  where x_j are values for input variables and y is the output
- This implies the existence of a “teacher" who knows the right answers
- What to learn: A function $f : X_1 \times X_2 \times \ldots \times X_n \rightarrow Y$, which maps the input variables into the output domain
- Goal: *minimize the error (loss) function*
  - Ideally, we would like to minimize error on all possible instances
  - But we only have access to a limited set of data...

Example: ImageNet Image Classification

Reinforcement Learning

- Training experience: interaction with an environment; the agent receives a numerical reward signal
  - E.g., a trading agent in a market; the reward signal is the profit
- What to learn: a way of behaving that is very rewarding in the long run
- Goal: estimate and maximize the long-term cumulative reward

Example: Playing Atari

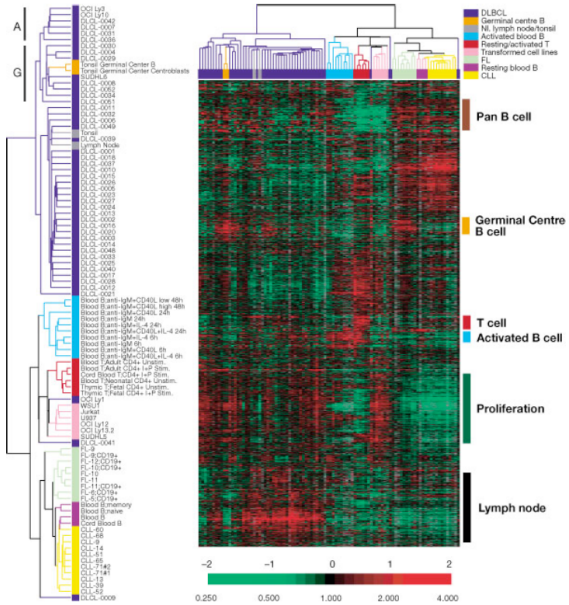


Unsupervised Learning

- Training experience: unlabelled data
- What to learn: interesting associations in the data
  - E.g., clustering, dimensionality reduction
- Often there is no single correct answer

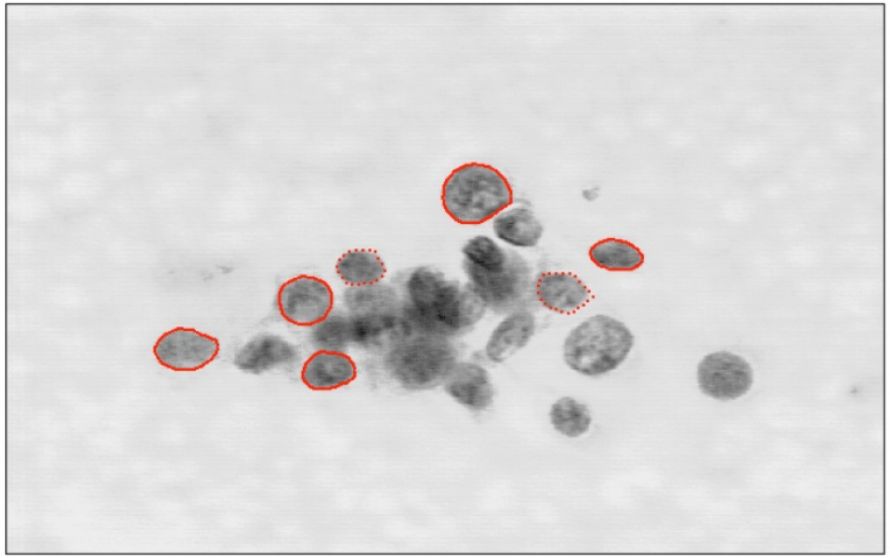
Example: Gene Analysis for Lymphoma

- Activity levels of all (25,000) genes were measured in lymphoma patients
- Cluster analysis determined three dierent subtypes (where only two were known before), having different clinical outcomes



A Working Example

- Cell samples were taken from tumors in breast cancer patients before surgery, and imaged
- Tumors were excised
- Patients were followed to determine whether or not the cancer recurred, and how long until recurrence or disease free



Data

- Thirty real-valued variables per tumor
- Two variables that can be predicted:
  - Outcome (R=recurrence, N=non-recurrence)
  - Time (until recurrence, for R, time healthy, for N)

tumor size	texture	perimeter	. . .	outcome	time
18.02	27.6	117.5		N	31
17.99	10.38	122.8		N	61
20.29	14.34	135.1		R	27
. . .					

Terminology

- Columns are called **input variables** or **features** or **attributes**
- The outcome and time (which we are trying to predict) are called **output variables** or **targets**
- A row in the table is called **training example** or **instance**
- The whole table is called (training) **data set**
- The problem of predicting the recurrence is called (binary) **classification**
- The problem of predicting the time is called **regression**

tumor size	texture	perimeter	. . .	outcome	time
18.02	27.6	117.5		N	31
17.99	10.38	122.8		N	61
20.29	14.34	135.1		R	27
. . .					

Terminology

- A training example *i* has the form: $\langle x_{i,1}, x_{i,2}, \ldots, x_{i,n}, y_i \rangle$ where *n* is the number of attributes (30 in our case).
- We will use the notation x_i to denote the column vector with elements $x_{i,1}, x_{i,2}, \ldots, x_{i,n}$
- The training set *D* consists of *m* training examples
- We denote the $m \times n$ matrix of attributes by *X* and the size-*m* column vector of outputs from the data set by *y*.

tumor size	texture	perimeter	. . .	outcome	time
18.02	27.6	117.5		N	31
17.99	10.38	122.8		N	61
20.29	14.34	135.1		R	27
. . .					

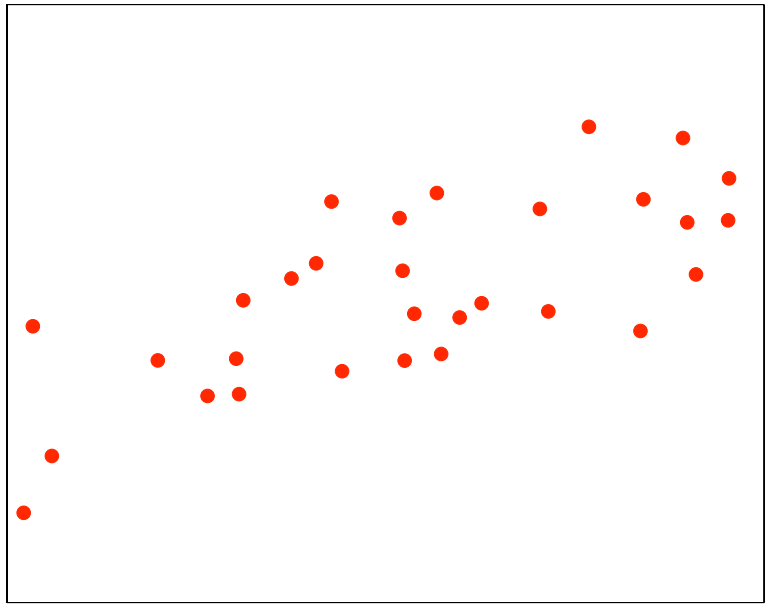
Supervised Learning Problem

- Let X denote the space of input values
- Let Y denote the space of output values
- Given a data set $D \subset X \rightarrow Y$, find a function $h : X \rightarrow Y$ such that $h(x)$ is a “good predictor" for the value of y
- h is called a hypothesis
- Problems are categorized by the type of output domain
  - If $Y = R$, this problem is called **regression**
  - If Y is a categorical variable (i.e., part of a finite discrete set), the problem is called **classification**
  - In general, Y could be a lot more complex (graph, tree, etc), which is called **structured prediction**

Steps to Solving A Supervised Learning Problem

1. Decide what the input-output pairs are
2. Decide how to encode inputs and outputs
   - This defines the input space *X*, and the output space *Y*
3. Choose a class of hypotheses/representations *H*
4. Choose an error function (cost function) to define the best hypothesis
5. Choose an algorithm for searching efficiently through the space of hypotheses

Example: What Hypothesis Class Should We Pick?



x	y
0.86	2.49
0.09	0.83
-0.85	-0.25
0.87	3.10
-0.44	0.87
-0.43	0.02
-1.10	-0.12
0.40	1.81
-0.96	-0.83
0.17	0.43

Linear Hypothesis

- Suppose y was a linear function of x:

$$h_w(x) = w_0 + w_1 x_1 + \cdots + w_n x_n$$

- w_i are called **parameters** or **weights**
- To simplify notation, we can add an attribute $x_0 = 1$ to the other n attributes (also called bias term or intercept term):

$$h_w(x) = \sum_{i=0}^{n} w_i x_i = w^T x$$

  where w and x are vectors of size $n + 1$.
- How should we pick w?

Error Minimization

- Intuitively, *w* should make the predictions of h_w close to the true values y on the data we have
- Hence, we will define an **error function** or **cost function** to measure how much our prediction differs from the ”true” answer
- We will pick *w* such that the error function is minimized
  - How should we choose the error function?

Least Mean Squares (LMS)

- Main idea: try to make $h_w(x)$ close to y on the examples in the training set
- We define a **sum-of-squares** error function

$$J(w) = \frac{1}{2}\sum_{i=0}^{n}(h_w x_i - y_i)^2$$

(the 1/2 is just for convenience)

- We will choose w such as to minimize $J(w)$

Notation Reminder

- Consider a function $f(u_1,u_2,...,u_n) : R^n \rightarrow R$ (for us, this will usually be an error function)
- The partial derivative w.r.t. u_i is denoted:

$$\frac{\partial}{\partial u_i} f(u_1, u_2, \ldots, u_n) : \mathbb{R}^n \mapsto \mathbb{R}$$

- The partial derivative is the derivative along the u_i axis, keeping all other variables fixed
- The gradient $\nabla f(u_1,u_2,...,u_n) : R^n \rightarrow R$ is a function which outputs a vector containing the partial derivatives.

$$\nabla f = \left\langle \frac{\partial}{\partial u_1} f, \frac{\partial}{\partial u_2} f, \ldots, \frac{\partial}{\partial u_n} f \right\rangle$$

Partial Difference

$$
\begin{aligned}
\frac{\partial}{\partial w_j} J(\mathbf{w}) &= \frac{\partial}{\partial w_j} \frac{1}{2} \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i)^2 \\
&= \frac{1}{2} \cdot 2 \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) \frac{\partial}{\partial w_j} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) \\
&= \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) \frac{\partial}{\partial w_j} \left(\sum_{l=0}^{n} w_l x_{i,l} - y_i \right) \\
&= \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) x_{i,j}
\end{aligned}
$$

- Setting all these partial derivatives to 0, we get a linear system with *(n + 1)* equations and *(n + 1)* unknowns.

Solution

- Recalling some multivariate calculus:

$$\begin{aligned}\nabla_{\mathbf{w}} J &= \nabla_{\mathbf{w}} (\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y}) \\ &= \nabla_{\mathbf{w}} (\mathbf{w}^T \mathbf{X}^T \mathbf{X}\mathbf{w} - \mathbf{y}^T \mathbf{X}\mathbf{w} - \mathbf{w}^T \mathbf{X}^T \mathbf{y} + \mathbf{y}^T \mathbf{y}) \\ &= 2\mathbf{X}^T \mathbf{X}\mathbf{w} - 2\mathbf{X}^T \mathbf{y}\end{aligned}$$

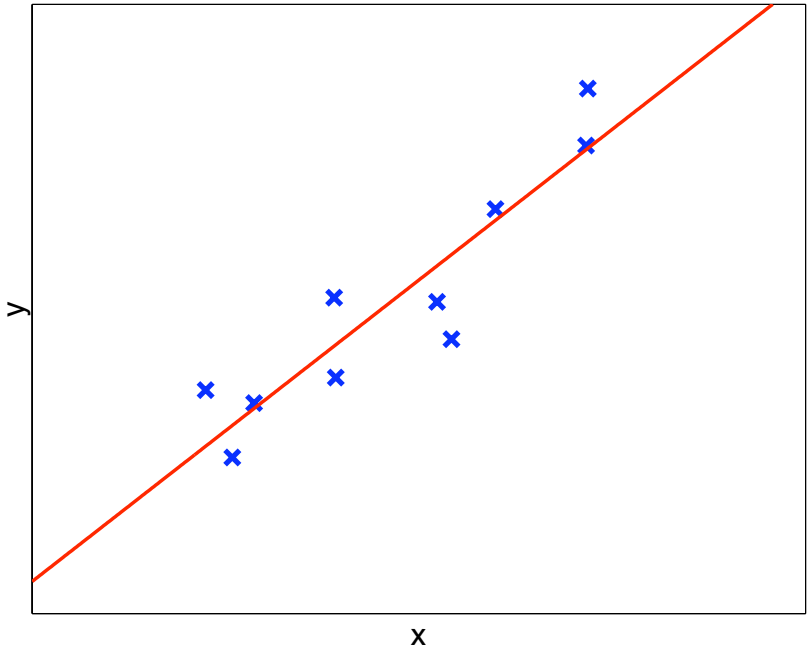
- Setting gradient equal to zero:

$$\begin{aligned}2\mathbf{X}^T \mathbf{X}\mathbf{w} - 2\mathbf{X}^T \mathbf{y} &= 0 \\ \Rightarrow \mathbf{X}^T \mathbf{X}\mathbf{w} &= \mathbf{X}^T \mathbf{y} \\ \Rightarrow \mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}\end{aligned}$$

- The inverse exists if the columns of X are linearly independent

Example: Data and Best Linear Hypothesis

$$y = 1.60x + 1.05$$



Remarks

- Linear models are an example of parametric models, because we choose a priori a number of parameters that does not depend on the size of the data
- Non-parametric models grow with the size of the data
  - Eg. Nearest neighbour, locally weighted linear regression
- Deep nets are very large parametric models.

提纲

1. Basic Concepts

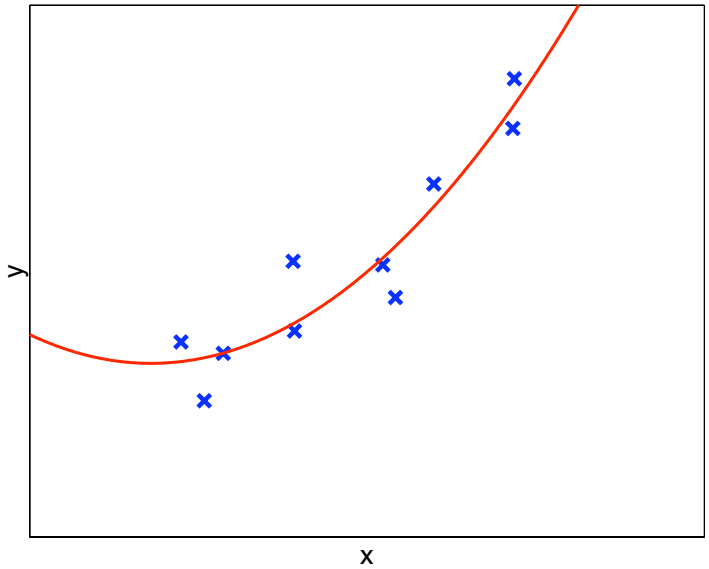
2. Types of ML Problems

3. Optimization

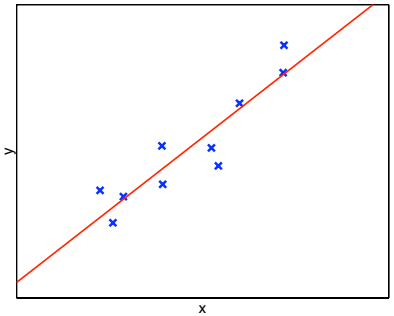
4. Taxonomy of Algorithms

Order-2 Fit

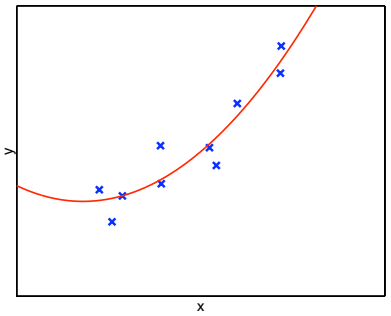
- Linear fit is actually order-1 fit and we can use a quadratic function to fit



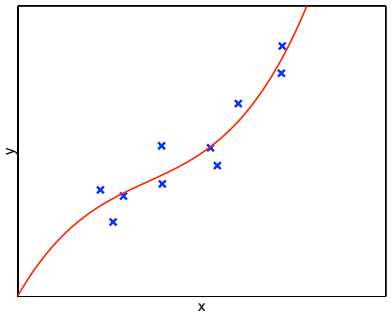
Higher Order Fit



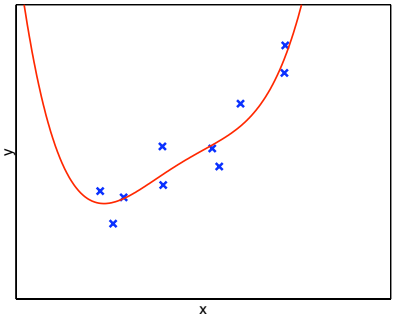
Linear fit



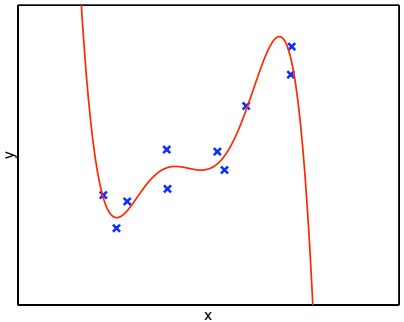
Order-2 fit



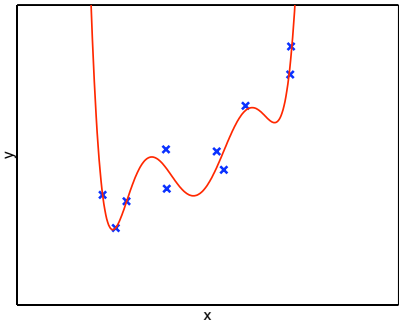
Order-3 fit



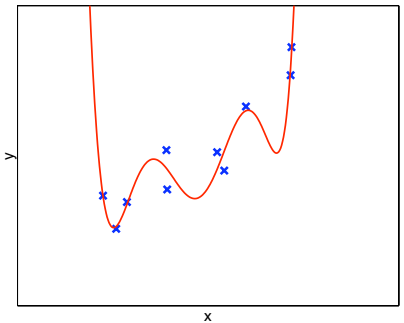
Order-4 fit



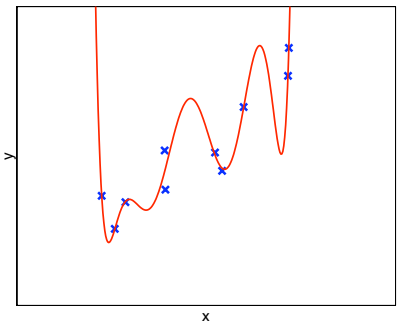
Order-5 fit



Order-6 fit



Order-7 fit



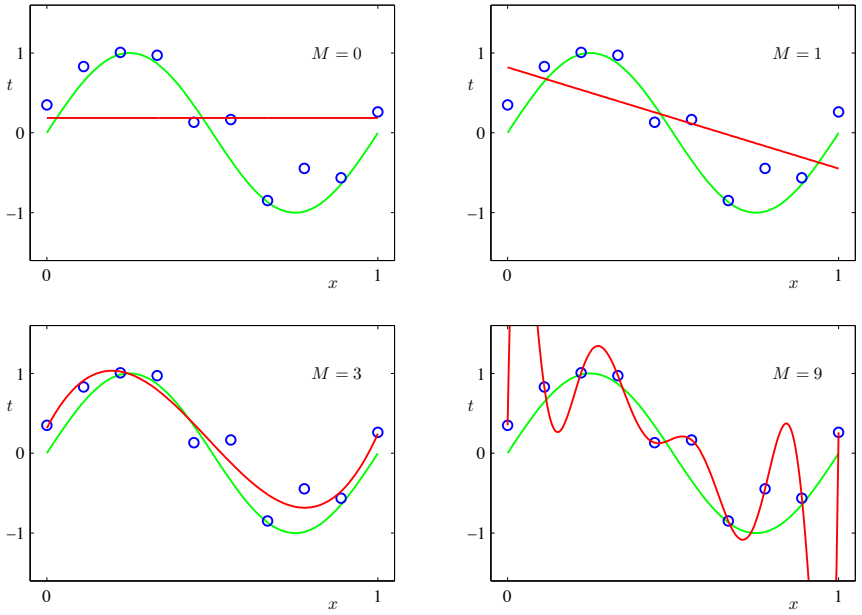
Order-8 fit

Overfitting

- A general, HUGELY IMPORTANT problem for all machine learning algorithms
- We can find a hypothesis that predicts perfectly the training data but does not generalize well to new data
  - E.g., a lookup table!
- We are seeing an instance here: if we have a lot of parameters, the hypothesis ”memorizes” the data points, but is wild everywhere else.

Overfitting and Underfitting

- The higher the degree of the polynomial M, the more degrees of freedom
- Typical overfitting means that error on the training data is very low, but error on new instances is high
- Typical underfitting means that error on the training data is very high (few dof)



Cross-Validation

- A general procedure for estimating the true error of a predictor
- The data is split into two subsets:
  - A training and validation set used only to find the right predictor
  - A test set used to report the prediction error of the algorithm
- These sets must be disjoint!
- The process is repeated several times, and the results are averaged to provide error estimates.

Cross-Validation

Regulation

- Remember the intuition: complicated hypotheses lead to overfitting
- Idea: change the error function to penalize hypothesis complexity:

$$J(\mathbf{w}) = J_D(\mathbf{w}) + \lambda J_{pen}(\mathbf{w})$$

- This is called **regularization** in machine learning and shrinkage in statistics
- λ is called **regularization coefficient** and controls how much we value fitting the data well, vs. a simple hypothesis

Regularization for Linear Models

- A squared penalty on the weights would make the math work nicely in our case:

$$\frac{1}{2}(\mathbf{\Phi w} - \mathbf{y})^T(\mathbf{\Phi w} - \mathbf{y}) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

- This is also known as L_2 regularization, or weight decay in neural networks
- By re-grouping terms, we get:

$$J_D(\mathbf{w}) = \frac{1}{2}(\mathbf{w}^T(\mathbf{\Phi}^T\mathbf{\Phi} + \lambda\mathbf{I})\mathbf{w} - \mathbf{w}^T\mathbf{\Phi}^T\mathbf{y} - \mathbf{y}^T\mathbf{\Phi w} + \mathbf{y}^T\mathbf{y})$$

- Optimal solution (obtained by solving $\nabla_{\mathbf{w}} J_D(\mathbf{w}) = 0$

$$\mathbf{w} = (\mathbf{\Phi}^T\mathbf{\Phi} + \lambda I)^{-1}\mathbf{\Phi}^T\mathbf{y}$$

提纲

1. Basic Concepts

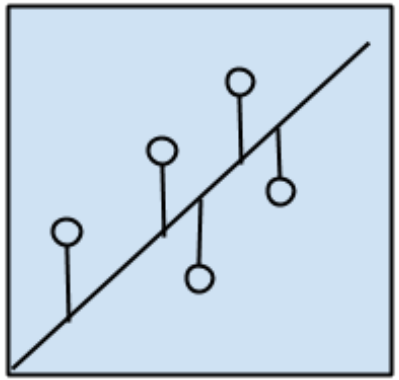
2. Types of ML Problems

3. Optimization

4. Taxonomy of Algorithms

Regression Algorithms

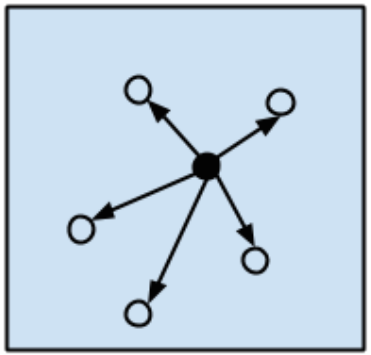
- Regression is concerned with modeling the relationship between variables that is iteratively refined using a measure of error in the predictions made by the model
- Regression methods are a workhorse of statistics and have been co-opted into statistical machine learning. This may be confusing because we can use regression to refer to the class of problem and the class of algorithm
- The most popular regression algorithms are:
  - Ordinary Least Squares Regression (OLSR)
  - Linear Regression
  - Logistic Regression
  - Stepwise Regression
  - Multivariate Adaptive Regression Splines (MARS)
  - Locally Estimated Scatterplot Smoothing (LOESS)



Regression Algorithms

Instance-Based Algorithms

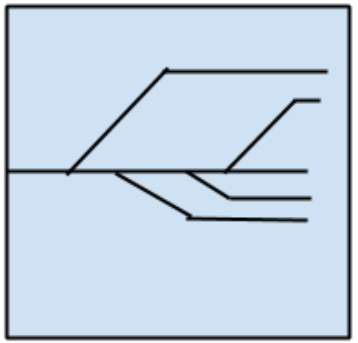
- Instance-based learning model is a decision problem with instance s or examples of training data that are deemed important or required to the model
- Such methods typically build up a database of example data and compare new data to the database using a similarity measure in order to find the best match and make a prediction
  - Also called winner-take-all methods and memory-based learning
  - Focus is put on the representation of the stored instances and similarity measures used between instances
- The most popular instance-based algorithms are:
  - k-Nearest Neighbor (kNN)
  - Learning Vector Quantization (LVQ)
  - Self-Organizing Map (SOM)
  - Locally Weighted Learning (LWL)



Instance-based Algorithms

Regularization Algorithms

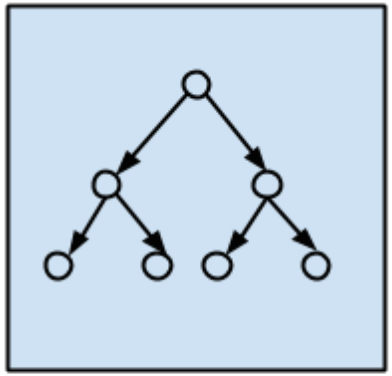
- An extension made to another method (typically regression methods) that penalizes models based on their complexity, favoring simpler models that are also better at generalizing
  - Listed regularization algorithms separately here because they are popular, powerful and generally simple modifications made to other
- The most popular regularization algorithms are:
  - Ridge Regression
  - Least Absolute Shrinkage and Selection Operator (LASSO)
  - Elastic Net
  - Least-Angle Regression (LARS)



Regularization Algorithms

Decision Tree Algorithms

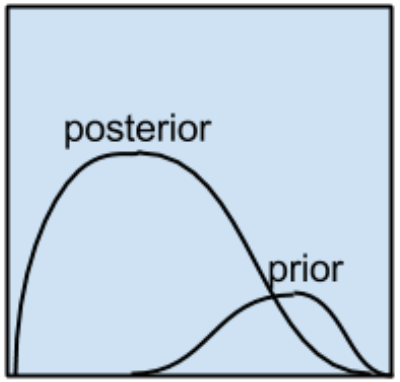
- Decision tree methods construct a model of decisions made based on actual values of attributes in the data
- Decisions fork in tree structures until a prediction decision is made for a given record. Decision trees are trained on data for classification and regression problems. Decision trees are often fast and accurate and a big favorite in machine learning.
- The most popular decision tree algorithms are:
  - Classification and Regression Tree (CART)
  - Iterative Dichotomiser 3 (ID3)
  - C4.5 and C5.0 (different versions of a powerful approach)
  - Chi-squared Automatic Interaction Detection (CHAID)
  - Decision Stump
  - M5
  - Conditional Decision Trees



Decision Tree Algorithms

Bayesian Algorithms

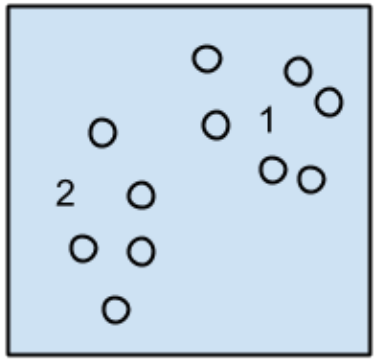
- Bayesian methods are those that explicitly apply Bayes' Theorem for problems such as classification and regression.
- The most popular Bayesian algorithms are:
  - Naive Bayes
  - Gaussian Naive Bayes
  - Multinomial Naive Bayes
  - Averaged One-Dependence Estimators (AODE)
  - Bayesian Belief Network (BBN)
  - Bayesian Network (BN)



Bayesian Algorithms

Clustering Algorithms

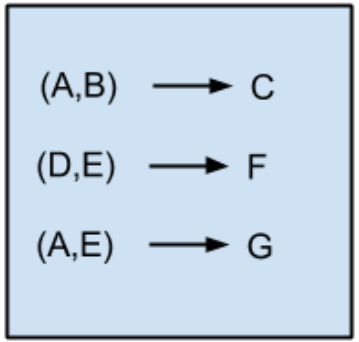
- Clustering, like regression, describes the class of problem and the class of methods.
- Clustering methods are typically organized by the modeling approaches such as centroid-based and hierarchal. All methods are concerned with using the inherent structures in the data to best organize the data into groups of maximum commonality
- The most popular clustering algorithms are:
  - k-Means
  - k-Medians
  - Expectation Maximisation (EM)
  - Hierarchical Clustering



Clustering Algorithms

Association Rule Learning Algorithms

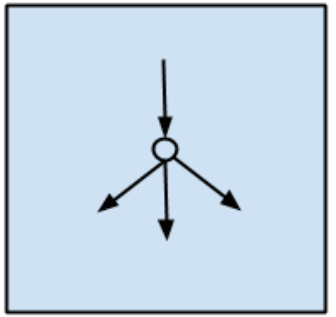
- Association rule learning methods extract rules that best explain observed relationships between variables in data
- These rules can discover important and commercially useful associations in large multidimensional datasets that can be exploited by an organization
- The most popular association rule learning algorithms are:
  - Apriori algorithm
  - Eclat algorithm



Association Rule Learning Algorithms

Association Rule Learning Algorithms

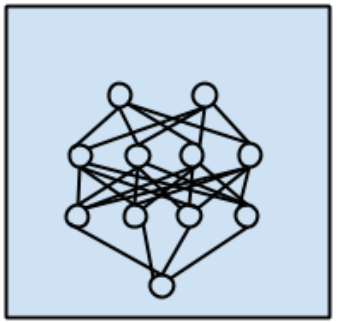
- Artificial Neural Networks are models that are inspired by the structure and/or function of biological neural networks
- They are a class of pattern matching that are commonly used for regression and classification problems but are really an enormous subfield comprised of hundreds of algorithms and variations for all manner of problem types.
- Note that I have separated out Deep Learning from neural networks because of the massive growth and popularity in the field. Here we are concerned with the more classical methods.
- The most popular artificial neural network algorithms are:
  - Perceptron
  - Back-Propagation
  - Hopfield Network
  - Radial Basis Function Network (RBFN)



Artificial Neural Network Algorithms

Deep Learning Algorithms

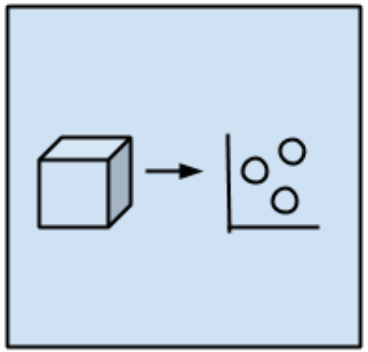
- Deep Learning methods are a modern update to Artificial Neural Networks that exploit abundant cheap computation
- They are concerned with building much larger and more complex neural networks and, as commented on above, many methods are concerned with semi-supervised learning problems where large datasets contain very little labeled data
- The most popular deep learning algorithms are:
  - Deep Boltzmann Machine (DBM)
  - Deep Belief Networks (DBN)
  - Convolutional Neural Network (CNN)
  - Stacked Auto-Encoders



Deep Learning Algorithms

Dimensionality Reduction Algorithms

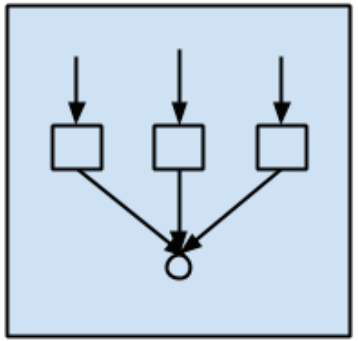
- Like clustering methods, dimensionality reduction seek and exploit the inherent structure in the data, but in this case in an unsupervised manner or order to summarize or describe data using less information
- This can be useful to visualize dimensional data or to simplify data which can then be used in a supervised learning method. Many of these methods can be adapted for use in classification and regression
- The most popular dimensionality reduction algorithms are:
- Principal Component Analysis (PCA)
- Principal Component Regression (PCR)
- Partial Least Squares Regression (PLSR)
- Sammon Mapping
- Multidimensional Scaling (MDS) Projection Pursuit
- Linear Discriminant Analysis (LDA)
- Mixture Discriminant Analysis (MDA)
- Quadratic Discriminant Analysis (QDA)
- Flexible Discriminant Analysis (FDA)



Dimensional Reduction Algorithms

Ensemble Algorithms

- Ensemble methods are models composed of multiple weaker models that are independently trained and whose predictions are combined in some way to make the overall prediction
- Much effort is put into what types of weak learners to combine and the ways in which to combine them
- This is a very powerful class of techniques and as such is very popular
  - Boosting
  - Bootstrapped Aggregation (Bagging)
  - AdaBoost
  - Stacked Generalization (blending)
  - Gradient Boosting Machines (GBM)
  - Gradient Boosted Regression Trees (GBRT)
  - Random Forest



Ensemble Algorithms

Microsoft Azure Machine Learning: Algorithm Cheat Sheet

This cheat sheet helps you choose the best Azure Machine Learning Studio algorithm for your predictive analytics solution. Your decision is driven by both the nature of your data and the question you're trying to answer.

START

Finding unusual data points

Discovering structure

Three or more

Predicting categories

Predicting values

Two

ANOMALY DETECTION

- **One-class SVM** — >100 features, aggressive boundary
- **PCA-based anomaly detection** — Fast training

CLUSTERING

- **K-means**

MULTI-CLASS CLASSIFICATION

- Fast training, linear model — **Multiclass logistic regression**
- Accuracy, long training times — **Multiclass neural network**
- Accuracy, fast training — **Multiclass decision forest**
- Accuracy, small memory footprint — **Multiclass decision jungle**
- Depends on the two-class classifier, see notes below — **One-v-all multiclass**

REGRESSION

- **Ordinal regression** — Data in rank ordered categories
- **Poisson regression** — Predicting event counts
- **Fast forest quantile regression** — Predicting a distribution
- **Linear regression** — Fast training, linear model
- **Bayesian linear regression** — Linear model, small data sets
- **Neural network regression** — Accuracy, long training time
- **Decision forest regression** — Accuracy, fast training
- **Boosted decision tree regression** — Accuracy, fast training, large memory footprint

TWO-CLASS CLASSIFICATION

- **Two-class SVM** — >100 features, linear model
- **Two-class averaged perceptron** — Fast training, linear model
- **Two-class logistic regression** — Fast training, linear model
- **Two-class Bayes point machine** — Fast training, linear model

- Accuracy, fast training — **Two-class decision forest**
- Accuracy, fast training, large memory footprint — **Two-class boosted decision tree**
- Accuracy, small memory footprint — **Two-class decision jungle**
- >100 features — **Two-class locally deep SVM**
- Accuracy, long training times — **Two-class neural network**

Recap

- Machine learning algorithms make choices of hypothesis space, error function and optimization procedure
- In some cases, optimization is easy
- Gradient descent is a general procedure (lots more on this to come)
- All algorithms are affected by bias-variance trade-off (too much variance=overfitting)
- Bayesian interpretation gives us a handle on what the algorithms really do

Resources: Datasets

- UCI Repository: http://www.ics.uci.edu/~mlearn/MLRepository.html
- UCI KDD Archive: http://kdd.ics.uci.edu/summary.data.application.html
- Statlib: http://lib.stat.cmu.edu/
- Delve: http://www.cs.utoronto.ca/~delve/

Resources: Journals

- Journal of Machine Learning Research www.jmlr.org
- Machine Learning
- IEEE Transactions on Neural Networks
- IEEE Transactions on Pattern Analysis and Machine Intelligence
- Annals of Statistics
- Journal of the American Statistical Association
- ...

Resources: Conferences

- International Conference on Machine Learning (ICML)
- European Conference on Machine Learning (ECML)
- Neural Information Processing Systems (NIPS)
- International Joint Conference on Artificial Intelligence (IJCAI)
- ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)
- IEEE Int. Conf. on Data Mining (ICDM)
- International Conference on Learning Representations (ICLR)