Machine Learning: Summary

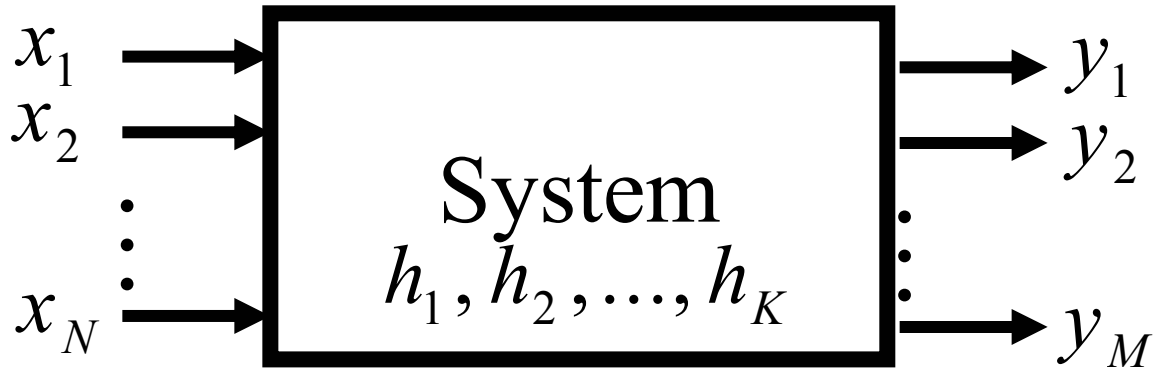
Greg Grudic

CSCI-4830

What is Machine Learning?

- "The goal of machine learning is to build computer systems that can adapt and learn from their experience."
  - Tom Dietterich

A Generic System



Input Variables: $\mathbf{x} = (x_1, x_2, \ldots, x_N)$

Hidden Variables: $\mathbf{h} = (h_1, h_2, \ldots, h_K)$

Output Variables: $\mathbf{y} = (y_1, y_2, \ldots, y_K)$

Another Definition of Machine Learning

- Machine Learning algorithms discover the relationships between the variables of a system (input, output and hidden) from direct samples of the system

- These algorithms originate form many fields:
  - Statistics, mathematics, theoretical computer science, physics, neuroscience, etc

When are ML algorithms NOT needed?

- When the relationships between all system variables (input, output, and hidden) is completely understood!

- This is NOT the case for almost any real system!

The Sub-Fields of ML

- Supervised Learning
- Reinforcement Learning
- Unsupervised Learning

Supervised Learning

- Given: Training examples

$$\left\{\left(\mathbf{x}_1, \mathbf{f}\left(\mathbf{x}_1\right)\right), \left(\mathbf{x}_2, \mathbf{f}\left(\mathbf{x}_2\right)\right), \ldots, \left(\mathbf{x}_P, \mathbf{f}\left(\mathbf{x}_P\right)\right)\right\}$$

for some unknown function (system) $\mathbf{y} = \mathbf{f}\left(\mathbf{x}\right)$

- Find $\mathbf{f}\left(\mathbf{x}\right)$
  - Predict $\mathbf{y}' = \mathbf{f}\left(\mathbf{x}'\right)$, where $\mathbf{x}'$ is not in the training set

Supervised Learning Algorithms

- Classification

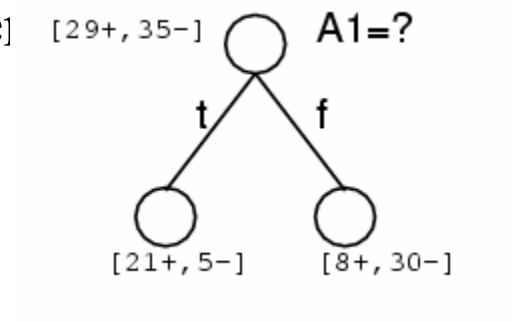
$$y \in \{1, ..., C\}$$

- Regression

$$y \in \Re$$

1-R (A Decision Tree Stump)

- Main Assumptions
  - Only one attribute is necessary.
  - Finite number of splits on the attribute.
- Hypothesis Space
  - Fixed size (parametric): Limited modeling pote



Naïve Bayes

- Main Assumptions:
  - All attributes are equally important.
  - All attributes are statistically independent (given the class value)
- Hypothesis Space
  - Fixed size (parametric): Limited modeling potential

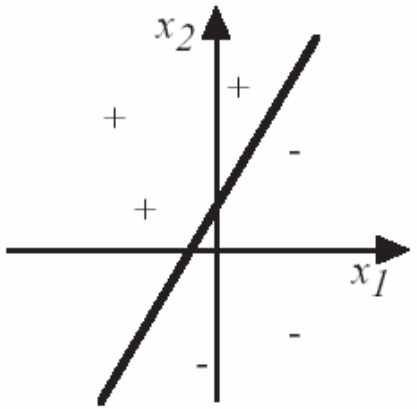
$$\Pr\left[y\middle|\mathbf{x}\right]=\frac{\Pr\left[x_1\middle|y\right]\Pr\left[x_2\middle|y\right]\cdots\Pr\left[x_d\middle|y\right]}{\Pr\left[\mathbf{x}\right]}$$

Linear Regression

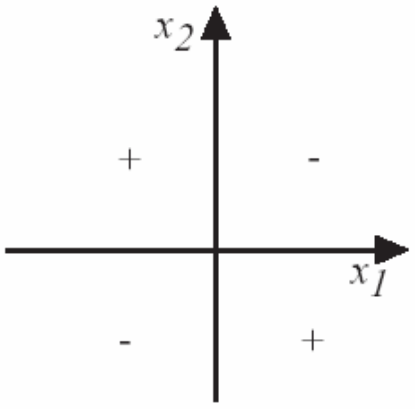
- Main Assumptions:
  - Linear weighted sum of attribute values.
  - Data is linearly separable.
  - Attributes and target values are real valued.
- Hypothesis Space
  - Fixed size (parametric) : Limited modeling potential

$$y = \sum_{i=1}^{d} a_i x_i + b$$

Linear Regression (Continued)



(a)



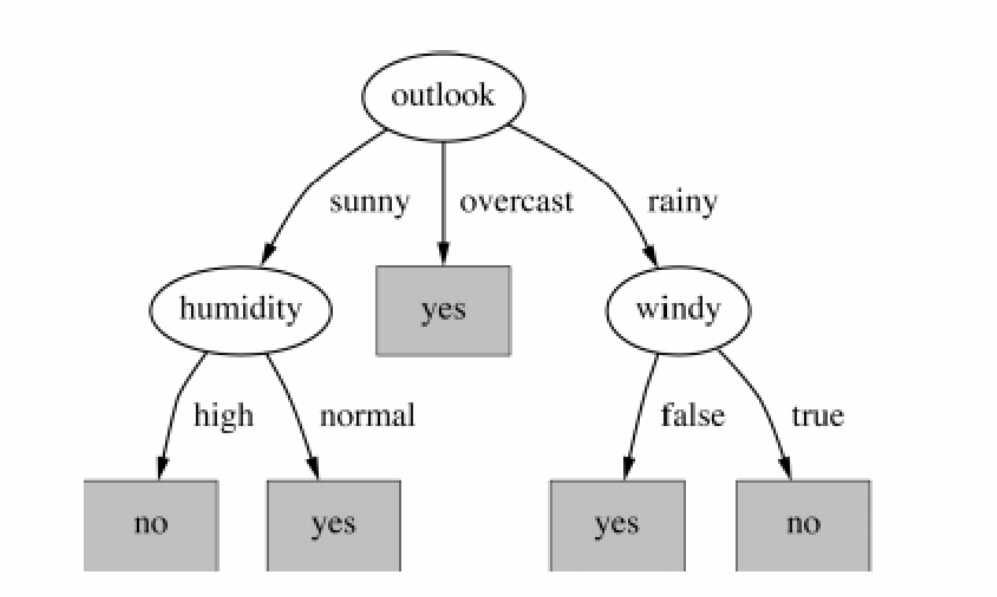
(b)

Linearly Separable

Not Linearly Separable

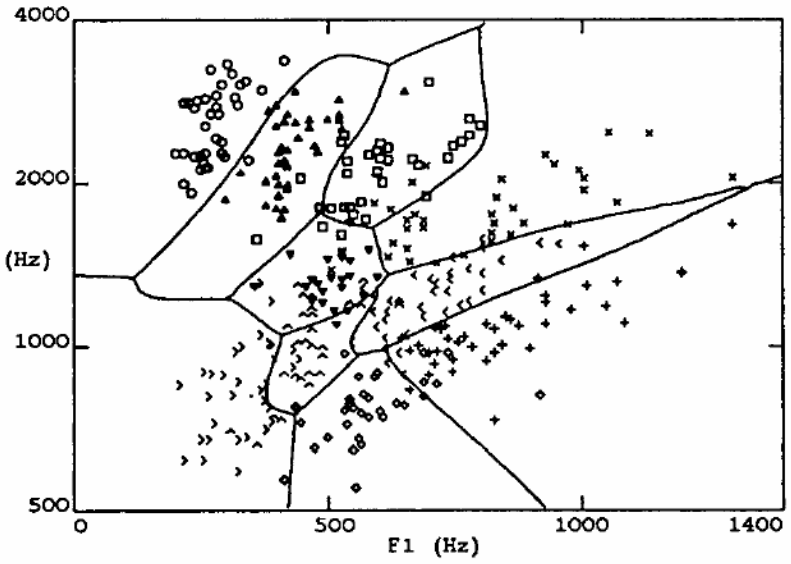
Decision Trees

- Main Assumption:
  - Data effectively modeled via decision splits on attributes.
- Hypothesis Space
  - Variable size (nonparametric): Can model any function

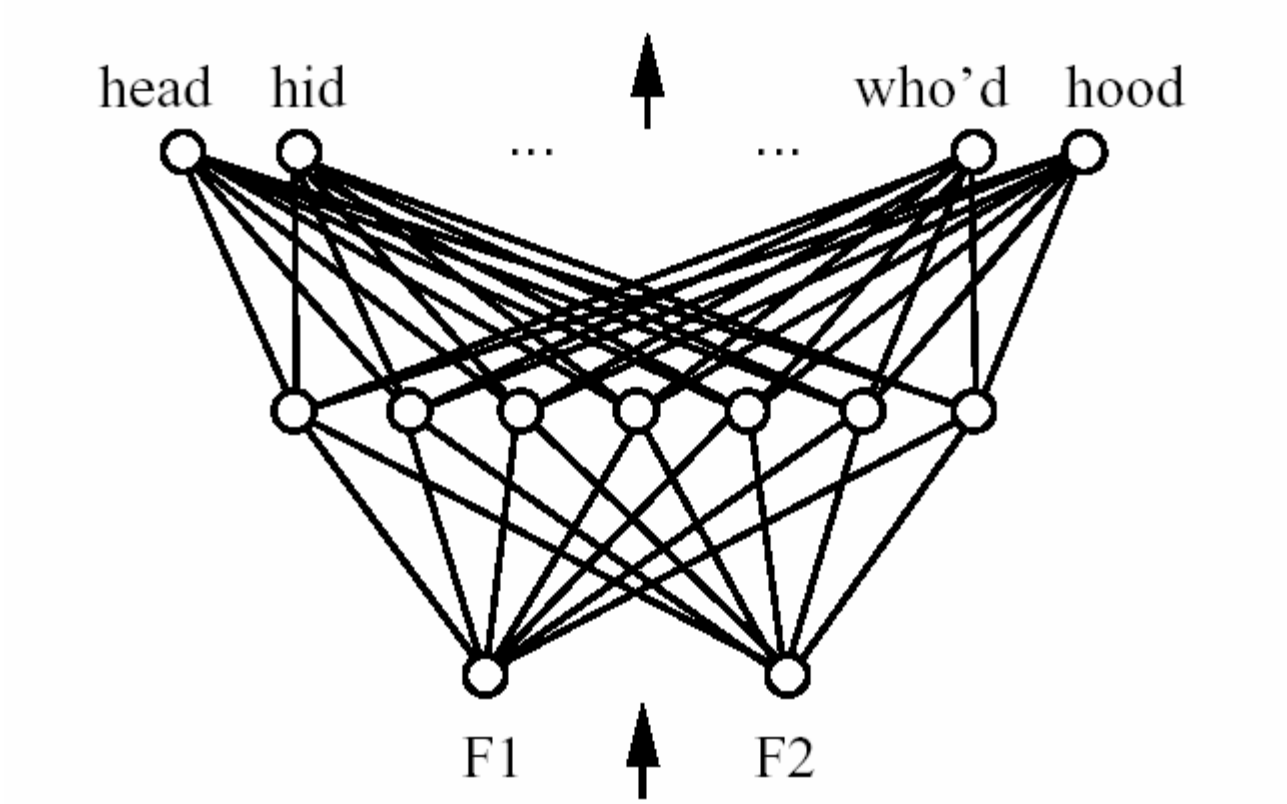


Neural Networks

- Main Assumption:
  - Many simple functional units, combined in parallel, produce effective models.
- Hypothesis Space
  - Variable size (nonparametric): Can model any function

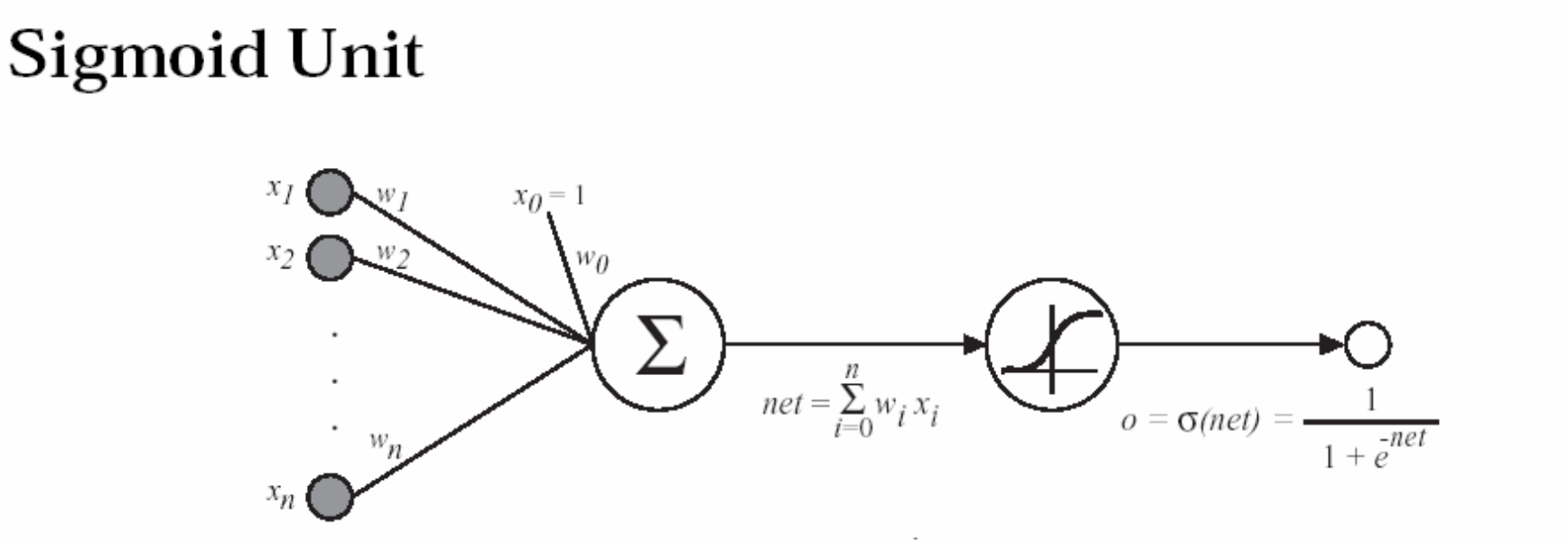


Neural Networks (Continued)



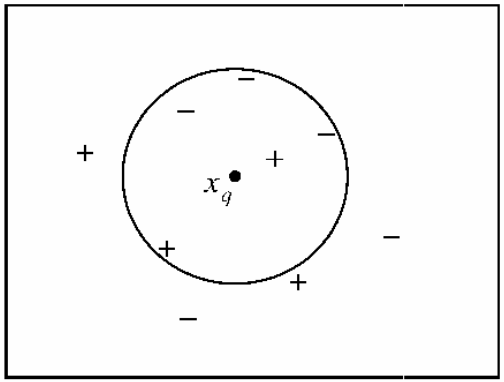
Neural Networks (Continued)

- Learn by modifying weights in Sigmoid Unit

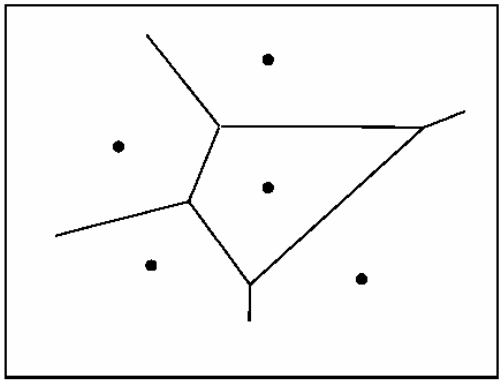


K Nearest Neighbor

- Main Assumption:
  - An effective distance metric exists.
- Hypothesis Space
  - Variable size (nonparametric): Can model any function



Classify according to Nearest Neighbor



Separates the input space

Bagging

- Main Assumption:
  - Combining many unstable predictors to produce a ensemble (stable) predictor.
  - Unstable Predictor: small changes in training data produce large changes in the model.
    - e.g. Neural Nets, trees
    - Stable: SVM, nearest Neighbor.
- Hypothesis Space
  - Variable size (nonparametric): Can model any function

Bagging (continued)

- Each predictor in ensemble is created by taking a bootstrap sample of the data.
- Bootstrap sample of N instances is obtained by drawing N example at random, with replacement.
- On average each bootstrap sample has 63% of instances
  - Encourages predictors to have uncorrelated errors.

Boosting

- Main Assumption:
  - Combining many weak predictors (e.g. tree stumps or 1-R predictors) to produce an ensemble predictor.
- Hypothesis Space
  - Variable size (nonparametric): Can model any function

Boosting (Continued)

- Each predictor is created by using a biased sample of the training data
  - Instances (training examples) with high error are weighted higher than those with lower error
- Difficult instances get more attention

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train base learner using distribution D_t.
- Get base classifier $h_t : X \to \mathbb{R}$.
- Choose $\alpha_t \in \mathbb{R}$.
- Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final classifier:

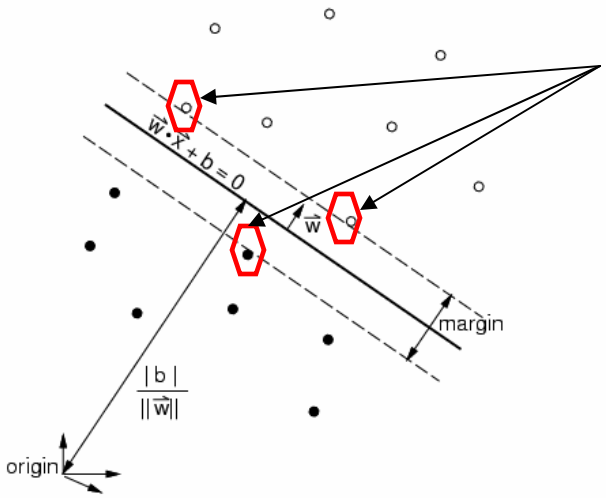
$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Figure 1: The boosting algorithm AdaBoost.

Support Vector Machines

- Main Assumption:
  - Build a model using minimal number of training instances (Support Vectors).
- Hypothesis Space
  - Variable size (nonparametric): Can model any function
- Based on PAC (probably almost correct) learning theory:
  - Minimize the probability that model error is greater than ε (small number)

Linear Support Vector Machines



We'd like the hyperplane with maximum margin

- size of margin is $\frac{2}{||\vec{w}||}$

So view our problem as a constrained optimization problem:

Minimize $||\vec{w}||^2$, subject to

$$y_i(\vec{x_i} \cdot \vec{w} + b) - 1 \geq 0, (\forall i)$$

Nonlinear Support Vector Machines

- Project into Kernel Space (Kernels constitute a distance metric in inputs space)
  - Polynomial classifier of degree p

$$K(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j} + 1)^p$$

  - Gaussian radial basis function classifier

$$K(\vec{x_i}, \vec{x_j}) = e^{-||\vec{x_i} \cdot \vec{x_j}||^2/2\sigma^2}$$

  - This one doesn't quite have a Φ

$$K(\vec{x_i}, \vec{x_j}) = tanh(\kappa \vec{x_i} \cdot \vec{x_j} - \delta)$$

Competing Philosophies in Supervised Learning

Goal is always to minimize the probability of model errors on future data!

- **A single Model:** Motivation - build a single good model.
  - Models that don't adhere to Occam's razor:
    - Minimax Probability Machine (MPM)
    - Trees
    - Neural Networks
    - Nearest Neighbor
    - Radial Basis Functions
  - Occam's razor models: The best model is the simplest one!
    - Support Vector Machines
    - Bayesian Methods
    - Other kernel based methods:
      - Kernel Matching Pursuit

Competing Philosophies in Supervised Learning

- **An Ensemble of Models:** Motivation – a good single model is difficult to compute (impossible?), so build many and combine them. Combining many uncorrelated models produces better predictors...
  - Models that don't use randomness or use directed randomness:
    - Boosting
      - Specific cost function
    - Gradient Boosting
      - Derive a boosting algorithm for any cost function
  - Models that incorporate randomness:
    - Bagging
      - Bootstrap Sample: Uniform random sampling (with replacement)
    - Stochastic Gradient Boosting
      - Bootstrap Sample: Uniform random sampling (with replacement)
    - Random Forests
      - Uniform random sampling (with replacement)
      - Randomize inputs for splitting at tree nodes

Evaluating Models

- Infinite data is best, but…
- N (N=10) Fold cross validation
  - Create N folds or subsets from the training data (approximately equally distributed with approximately the same number of instances).
  - Build N models, each with a different set of N-1 folds, and evaluate each model on the remaining fold
  - Error estimate is average error over all N models

Boostrap Estimate

The *bootstrap* is an estimation method that uses sampling with replacement to form the training set

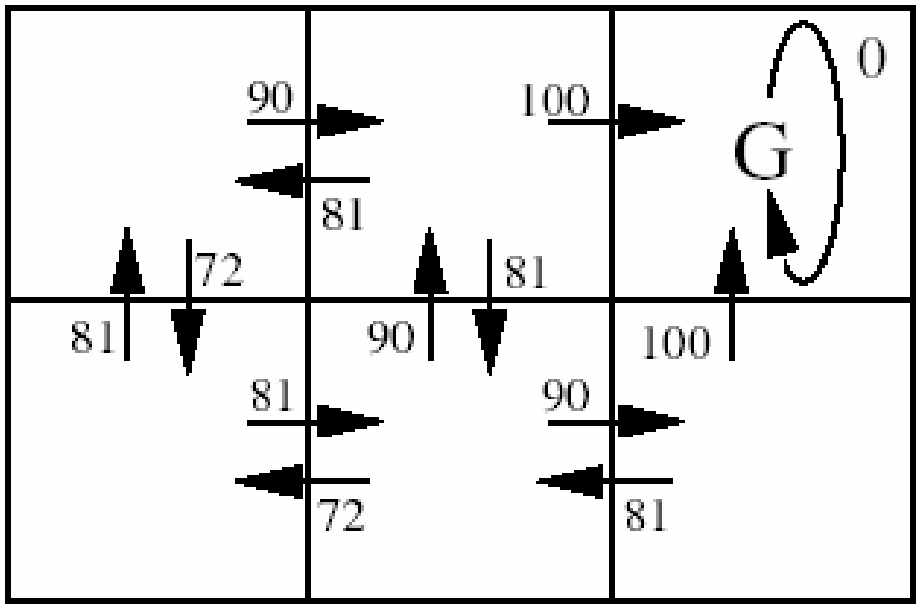
- A dataset of *n* instances is sampled *n* times with replacement to form a new dataset of *n* instances
- This data is used as the training set
- The instances from the original dataset that don't occur in the new training set are used for testing

Reinforcement Learning (RL)

Autonomous agent learns to act "optimally" without human intervention

- *Agent learns by stochastically interacting with its environment, getting infrequent rewards*
- *Goal: maximize infrequent reward*

Q Learning



$Q(s, a)$ values ($\gamma = 0.9$)

Agent's Learning Task

Execute actions in environment, observe results, and

- learn action policy $\pi : S \rightarrow A$ that maximizes

$$E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots]$$

  from any starting state in S

- here $0 \leq \gamma < 1$ is the discount factor for future rewards

Different from supervised learning:

- Target function is $\pi : S \rightarrow A$
- but we have no training examples of form $\langle s, a \rangle$
- training examples are of form $\langle \langle s, a \rangle, r \rangle$

Unsupervised Learning

- Studies how input patterns can be represented to reflect the statistical structure of the overall collection of input patterns
- No outputs are used (unlike supervised learning and reinforcement learning)
- unsupervised learner brings to bear prior biases as to what aspects of the structure of the input should be captured in the output.

Expectation Maximization (EM) Algorithm

- Clustering of data
  - K-Means
- Estimating unobserved or hidden variables