Lect-2: MLinNLP

Institute of Computational Linguistics
Peking University
王厚峰

Outline

- **Why is ML important**
- General Frame
- Generalization
- Experimental Evaluation

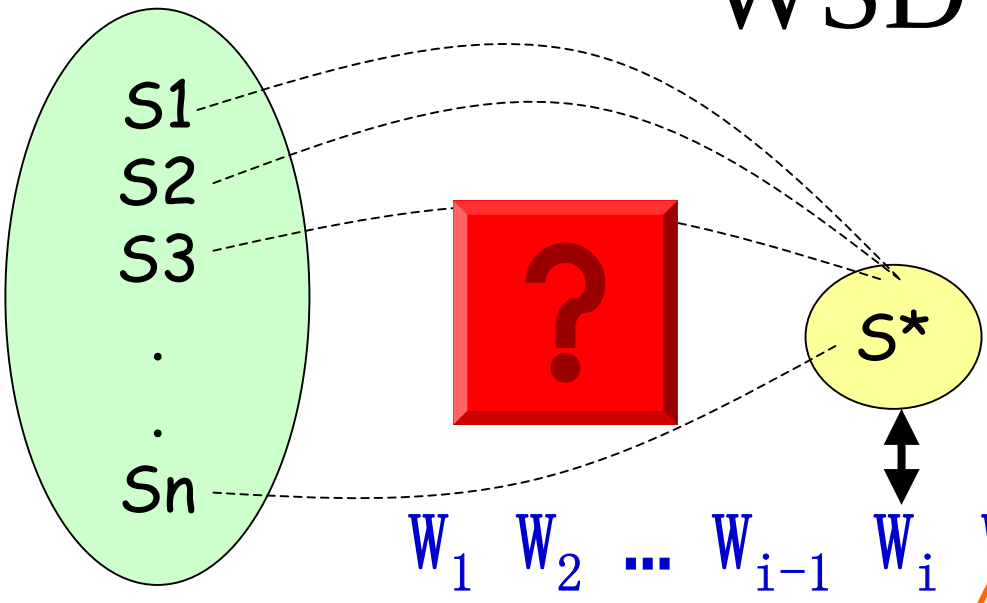
About some applications of NLP

- Many tasks in NLP can be reduced to **disambiguation**:
  - Word Segmentation
  - POS tagging
  - Parsing
  - WSD(polysemy)
  - Coreference resolution
  - …
- **Disambiguation => Classification**
- Which class(category) is the best?

Disambiguation=>classification

- **Classicification:** given a input x, find a best class y, i.e. MAX Pr(y|x) or y=h(x)
- **Ambiguity Resolution:** is a crucial problem for natural language understanding/processing.
- Ambiguity Resolution => Classification

WSD



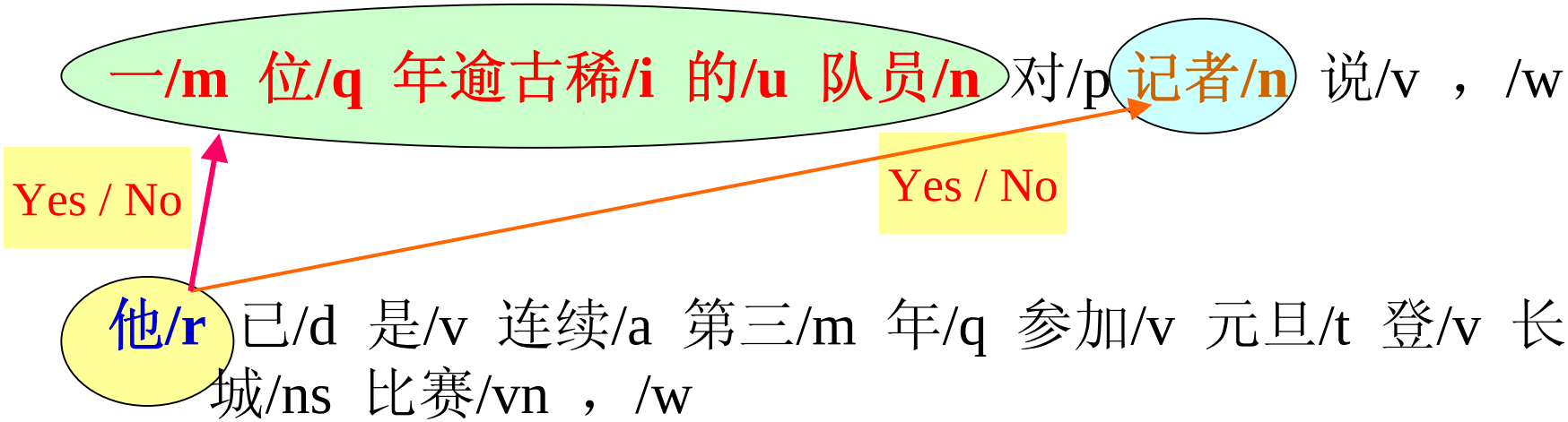
抄/v	照抄稿件
抄/v	走近道

- 此类/r 编著/v 内容/n 抄/v 自/p 别人/r 的/u 多/a ，/w
- 周边/n 群众/n 进城/v ，/w 习惯/v 抄/v 近道/n 。/w

Another Famous Example

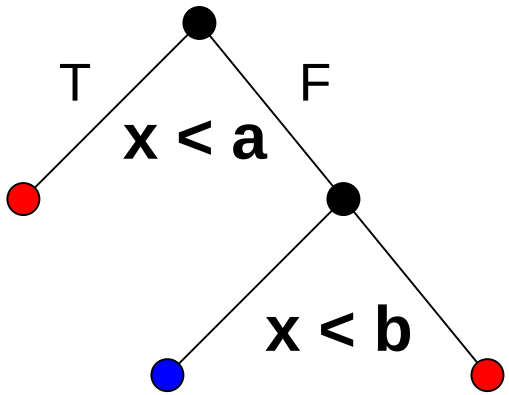
- Pen:He looked for his box everywhere. Finally, he found it. The box is in the pen!
- Bank

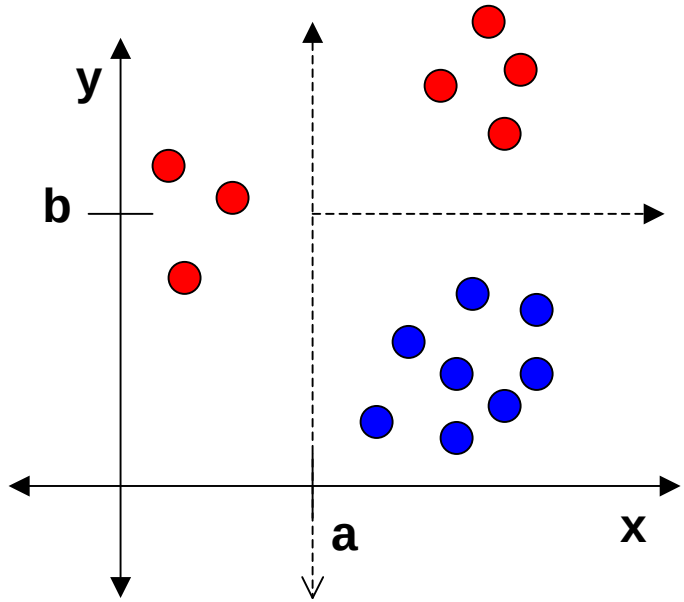
Coreference Resolution



Classification

- Decision Trees
  - Function space is Boolean formulae in Disjunctive Normal Form (DNF)



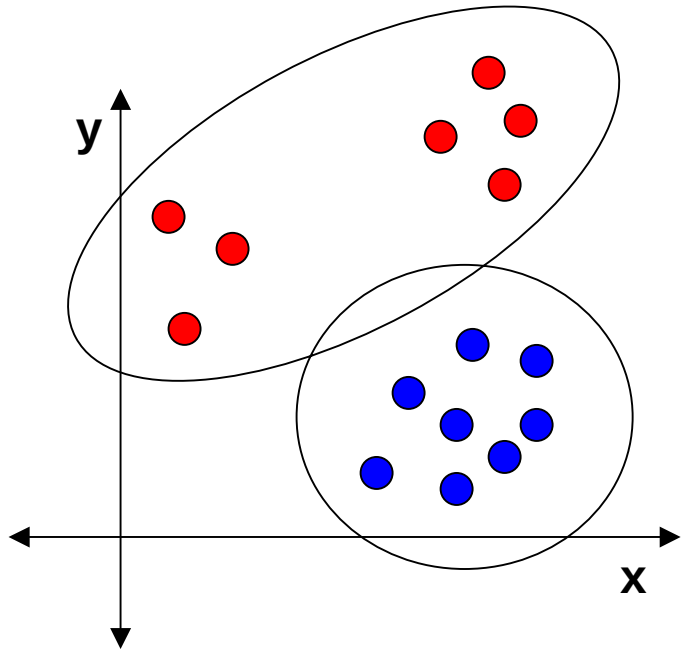


$$(x < a) \vee (\neg(x < a) \wedge \neg(x < b)) \rightarrow$$

else

Classification

- Probability Model
  - Function space is dependent on the distribution assumptions of the model



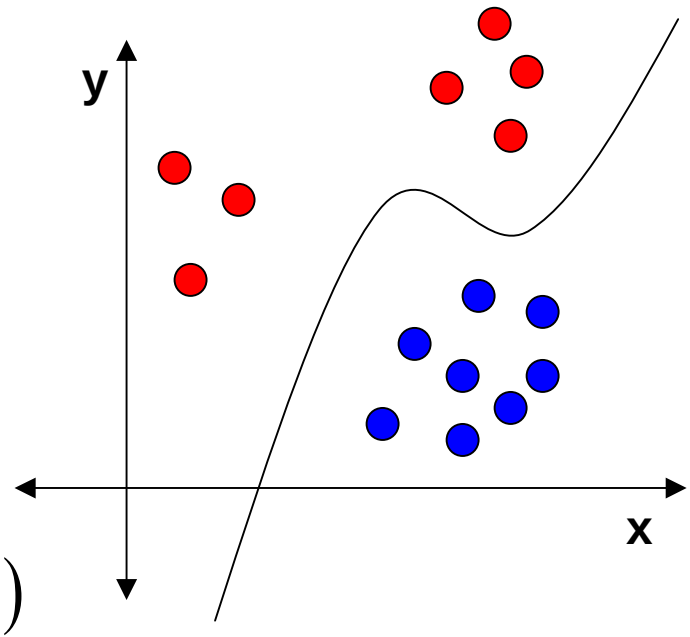
$$P(y \mid \bar{x}) = \frac{P(\bar{x} \mid y)P(y)}{P(\bar{x})}$$

$$y = \arg\max_{y \in Y} P(y \mid \bar{x})$$

Classification

- Discriminant Functions
  - Partition the d dimensional space with a (d-1) dimensional function
  - Function space is dependent on the function used to discriminate

$$\left[f(\bar{x}) > \theta \right] \Rightarrow h(\bar{x}) = \mathrm{sgn}\left(f(\bar{x}) - \theta \right)$$



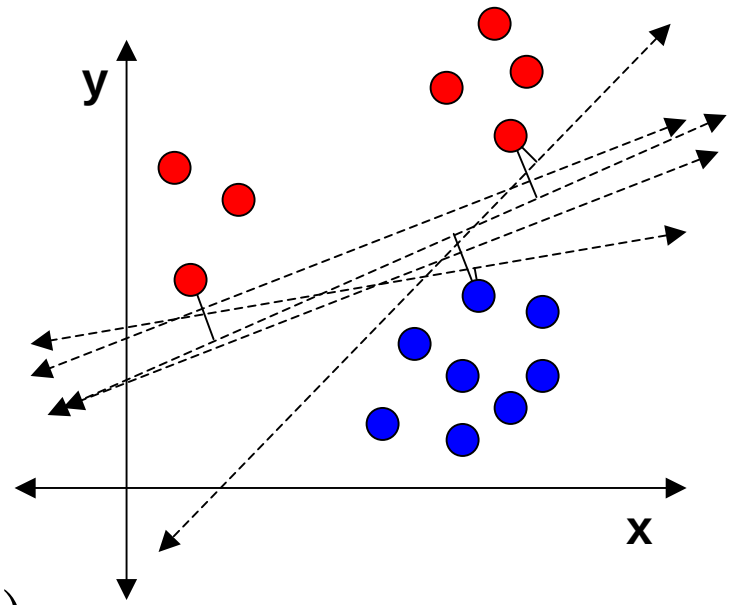
Note: A more complex function requires more data to generate an accurate model (sample complexity)

Classification

- Linear Discriminants
  - Partition the d dimensional space with a (d-1) dimensional linear function

$$f(\bar{x}) = \bar{w}^T \bar{x}$$

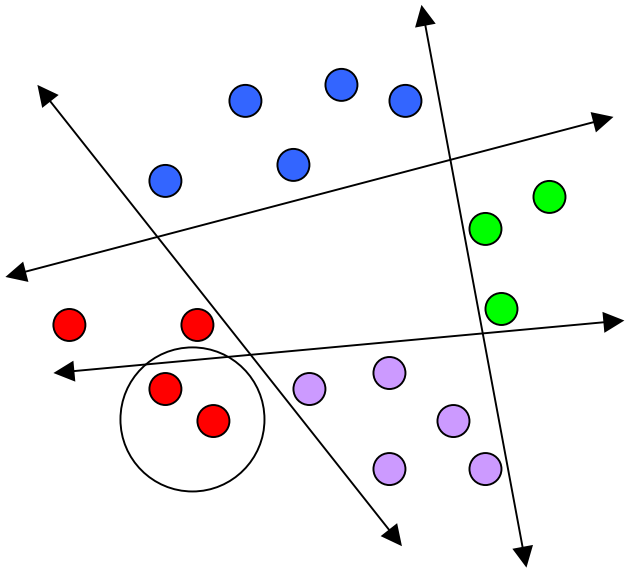
$$[f(\bar{x}) > \theta] \Rightarrow h(\bar{x}) = \mathrm{sgn}(f(\bar{x}) - \theta)$$



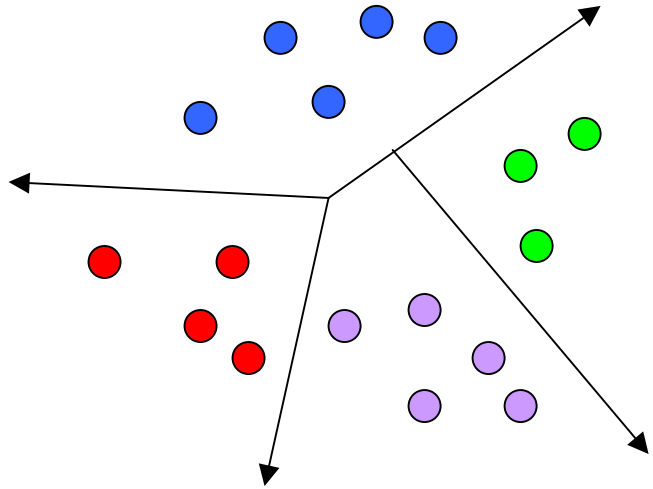
Define *margin* as the distance from the hyperplane to the point closest to it.

Multiclass Classification

$$y = \arg\max_{y \in Y} f_y(\bar{x})$$



One Versus All (**OvA**)



Constraint Classification

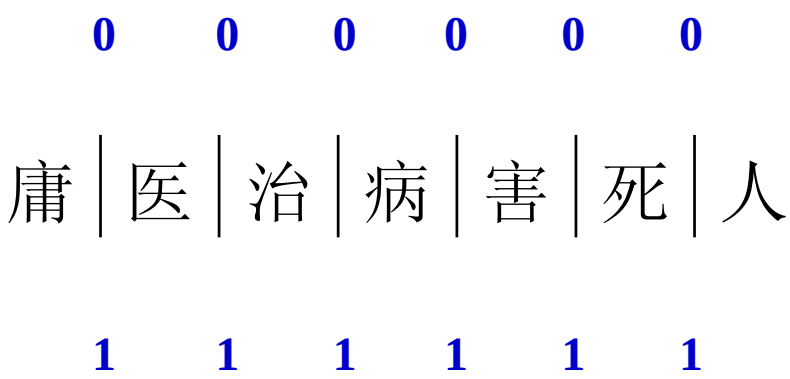
Beyond Classification Learning

- Standard classification problem assumes individual cases are disconnected and independent (i.i.d.: independently and identically distributed).
- Many NLP problems do not satisfy this assumption and involve making many connected decisions, each resolving a different ambiguity, but which are mutually dependent.
- More sophisticated learning and inference techniques are needed to handle such situations in general.

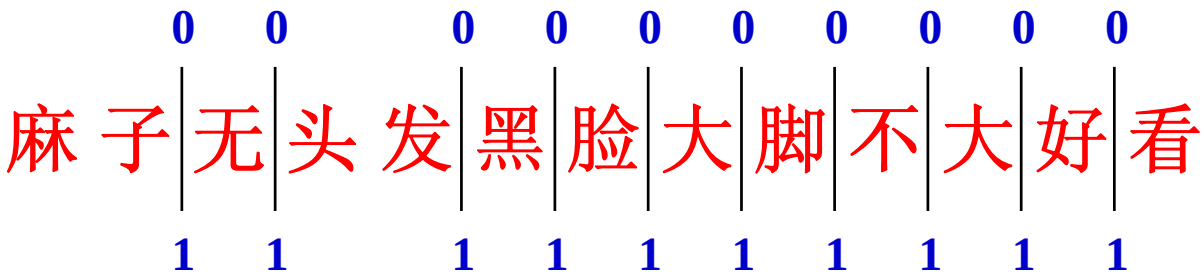
Sequence Label

- Many NLP problems can viewed as sequence labeling.
- Each token in a sequence is assigned a label.
- Labels of tokens are dependent on the labels of other tokens in the sequence, particularly their neighbors (not independent).

Word Segmentation



Sentences Segmentation



Information Extraction

- Identify phrases in language that refer to specific types of entities and relations in text.
- Named entity recognition is task of identifying names of people, places, organizations, etc. in text.

people **organizations** **places**

– 比尔.盖茨 创建了微软公司，近年多次到中国访问.

Semantic Role Labeling

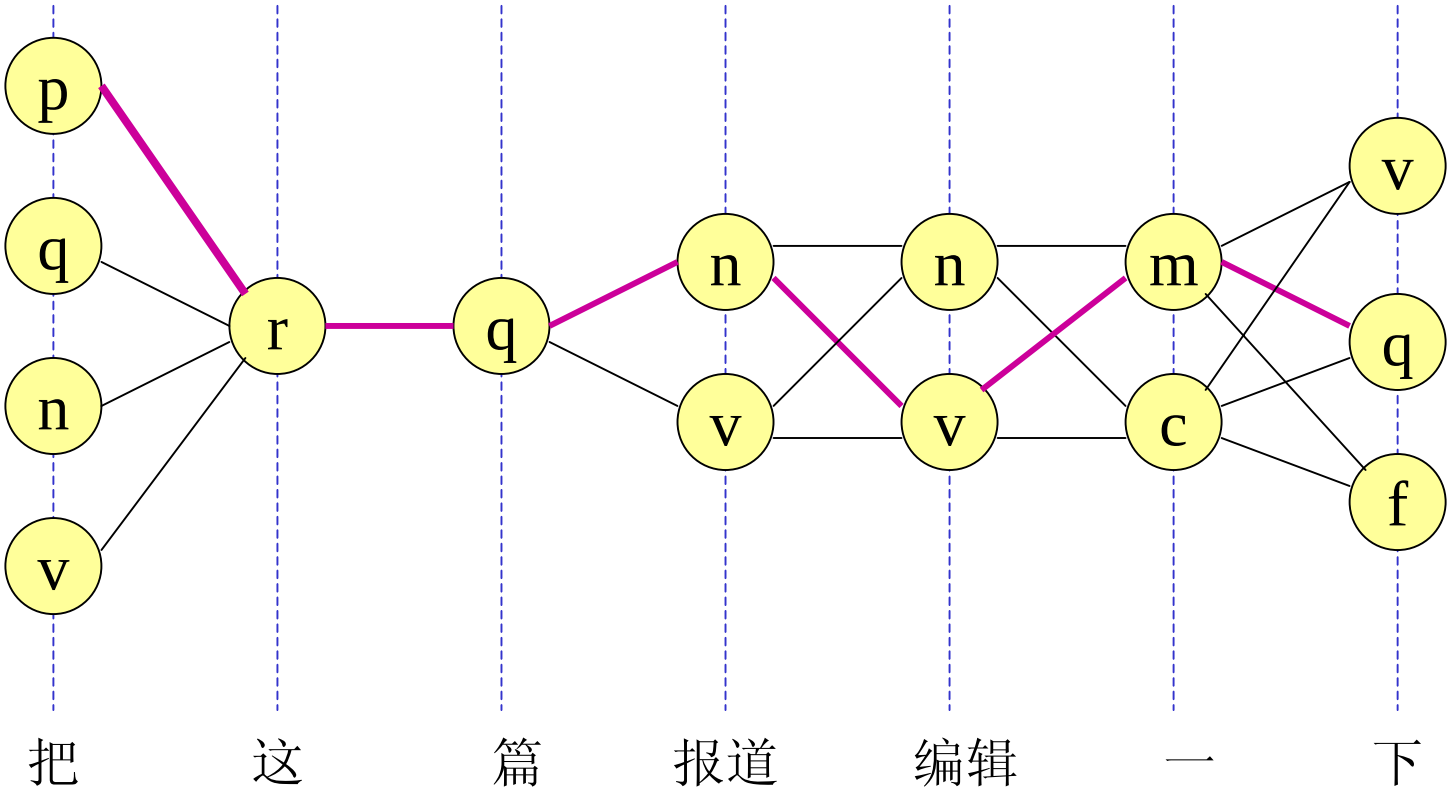
- For each clause, determine the semantic role played by each noun phrase that is an argument to the verb.

  agent **patient** **source** **destination** **instrument**

  – **张先生**驾驶**他的奥迪**把**夫人**从**家**送到**医院**.

- Also referred to a “case role analysis,” “thematic analysis,” and “shallow semantic parsing”

POS



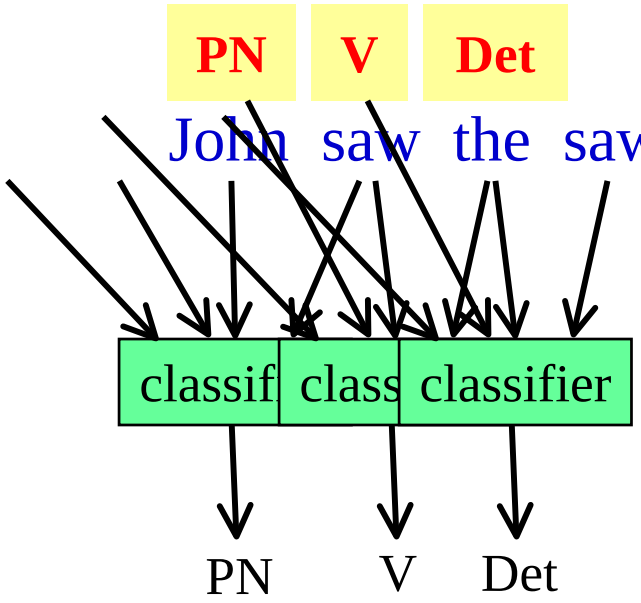
Sequence Labeling as Classification (from Raymond J. Mooney)

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window)

Sequence Labeling as Classification Using Outputs as Inputs

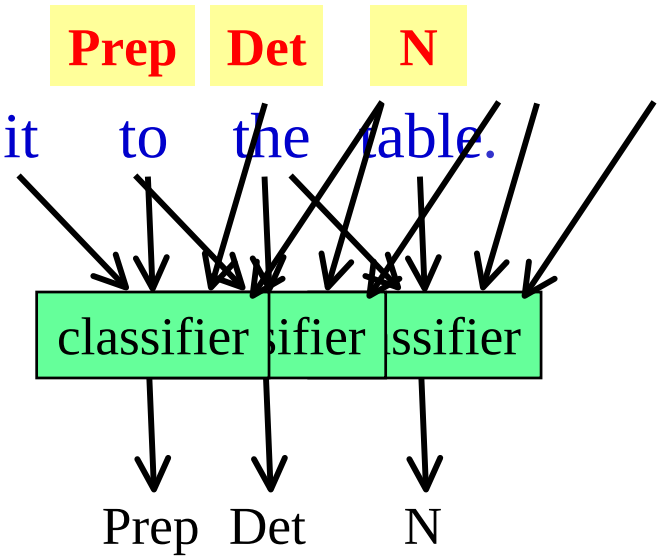
- Better input features are usually the categories of the surrounding tokens, but these are not available yet.
- Can use category of either the preceding or succeeding tokens by going forward or back and using previous output.

Forward Classification



Backward Classification

John saw the saw and decided to take it to the table.

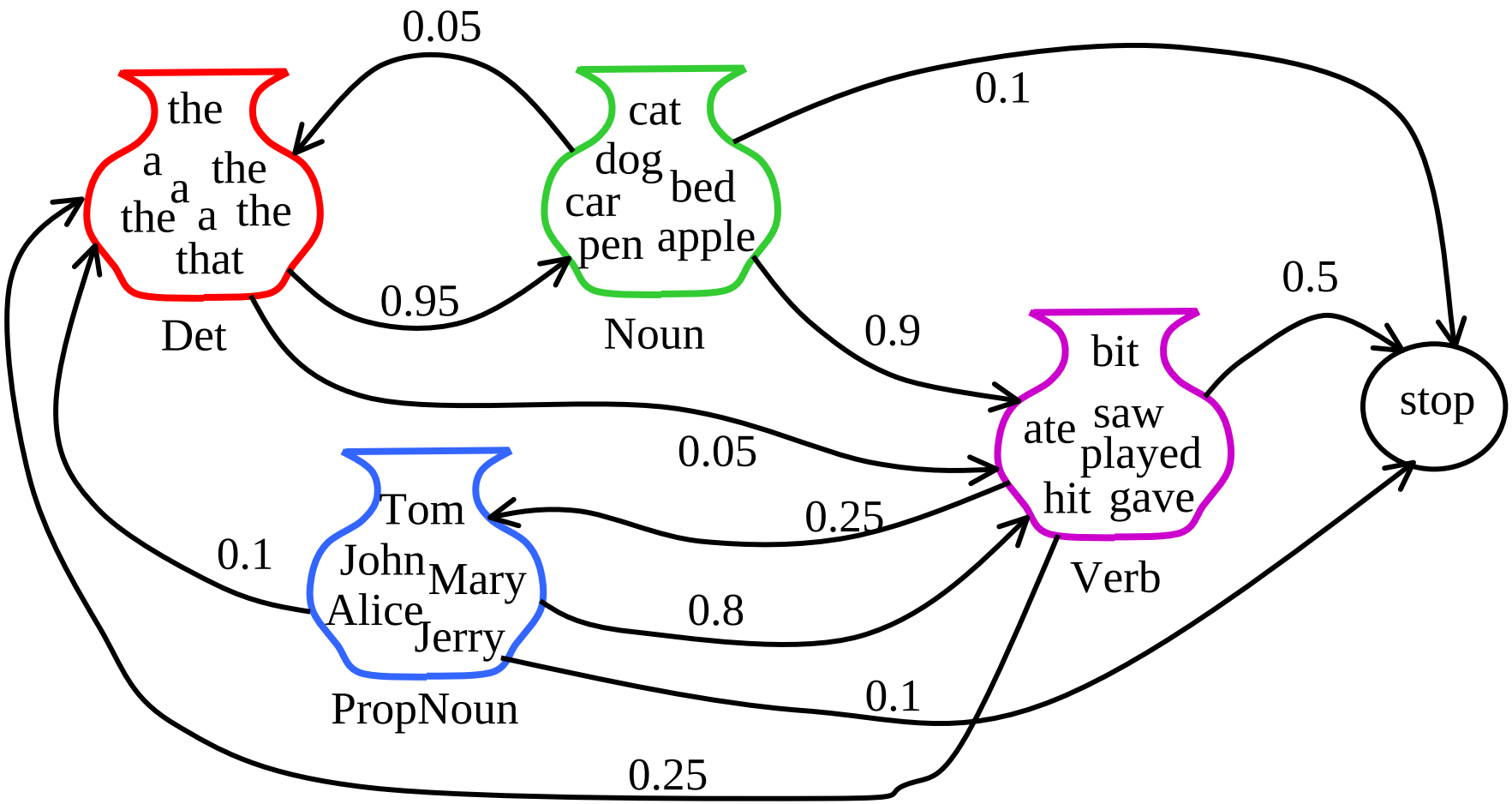


- Disambiguating “to” in this case would be even easier backward.

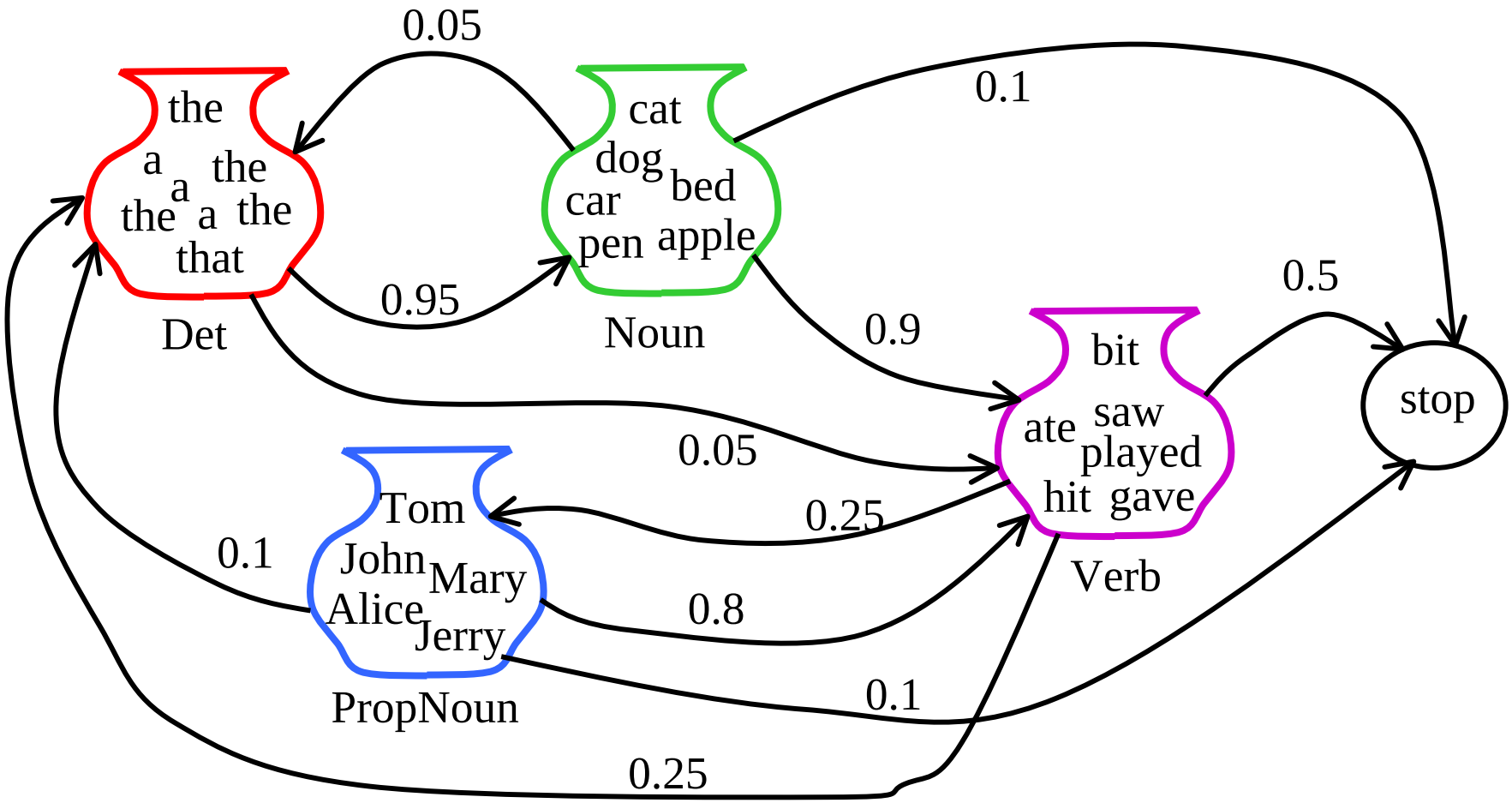
Probabilistic Sequence Models

- Probabilistic sequence models allow integrating uncertainty over multiple, interdependent classifications and collectively determine the most likely global assignment.
- Probabilistic Graphical models in this problem:
  - Hidden Markov Model (HMM): directed GM
  - Conditional Random Field (CRF): undirected GM

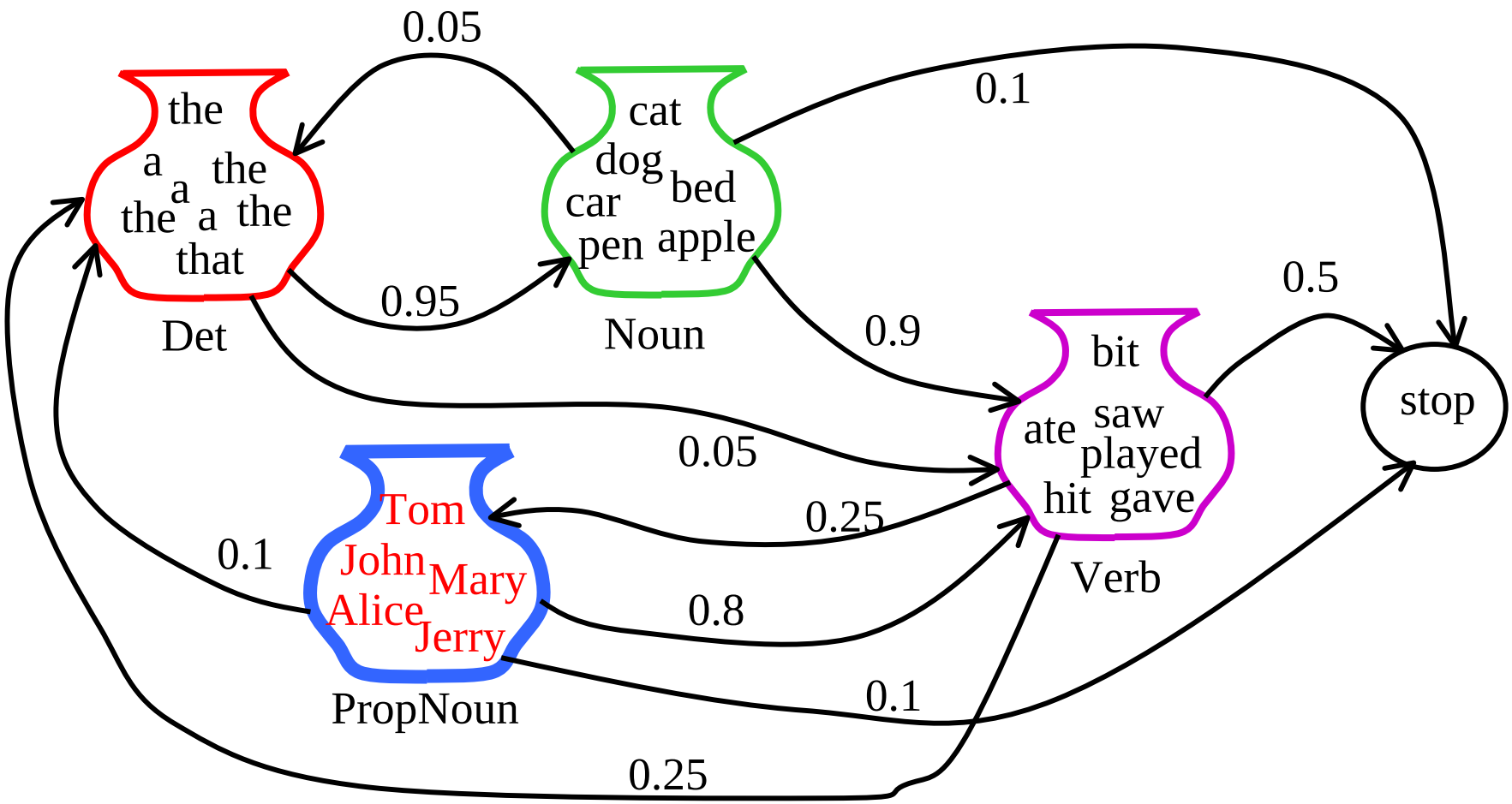
Sample HMM for POS



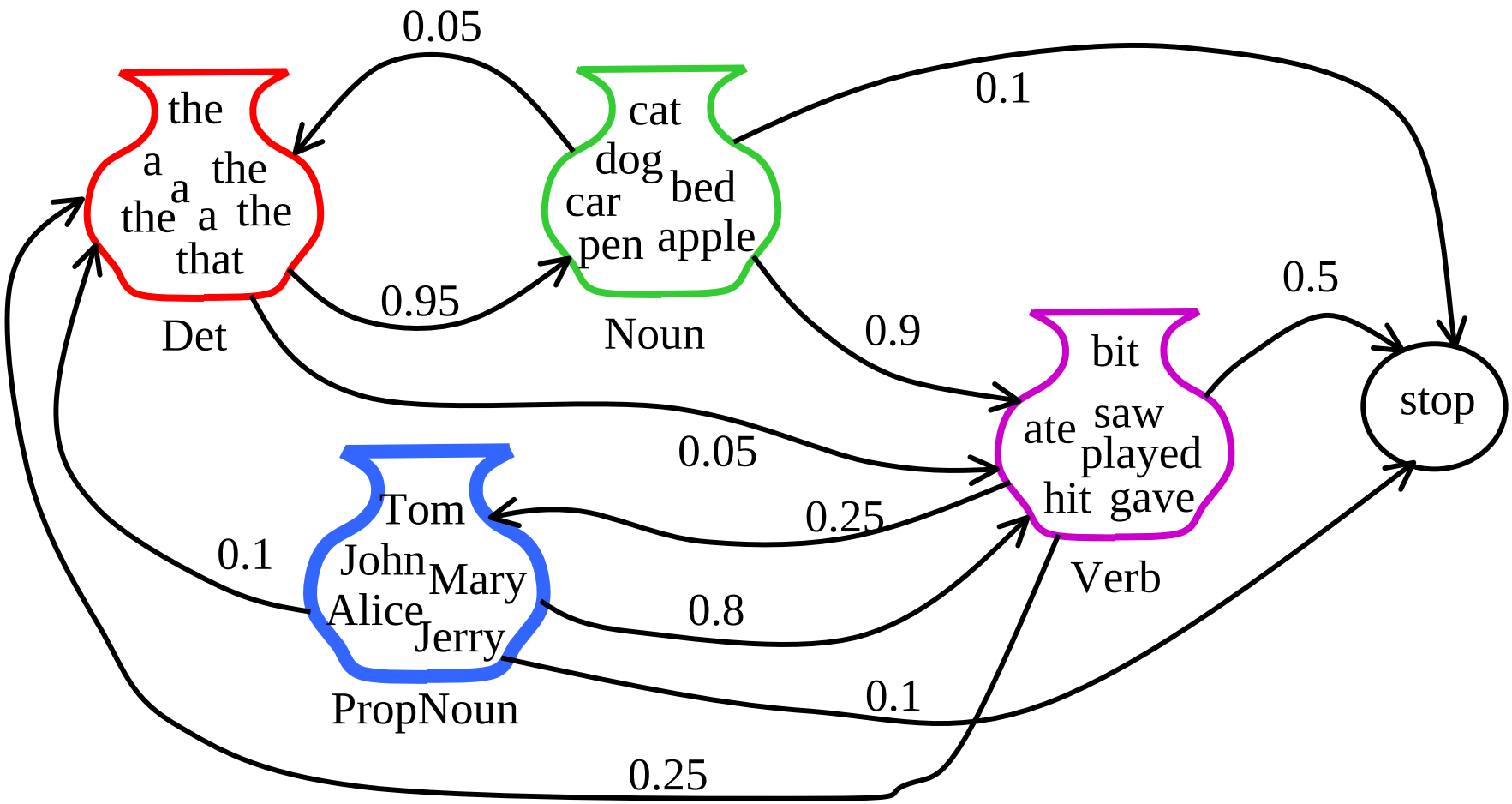
Sample HMM Generation



Sample HMM Generation

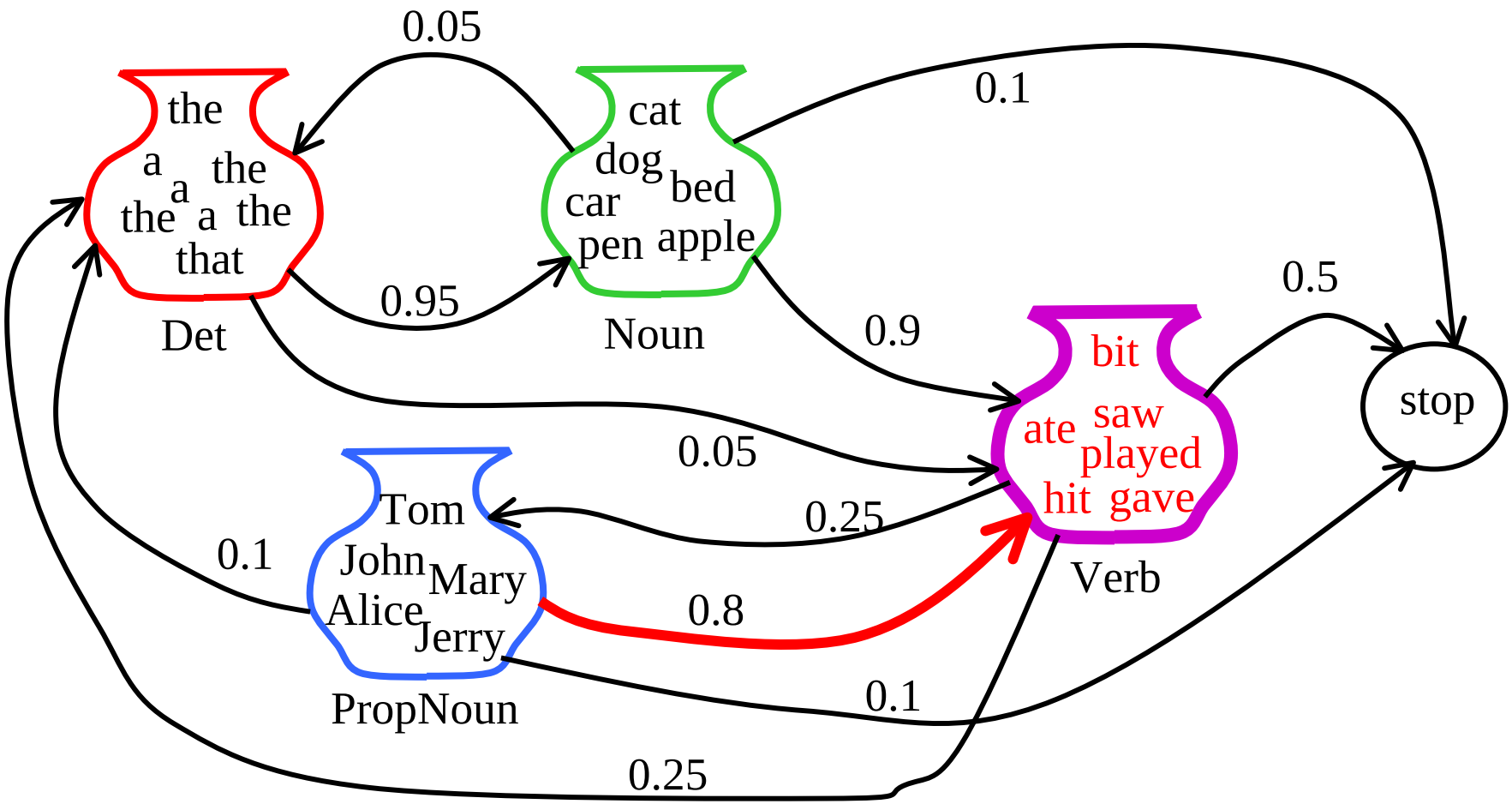


Sample HMM Generation



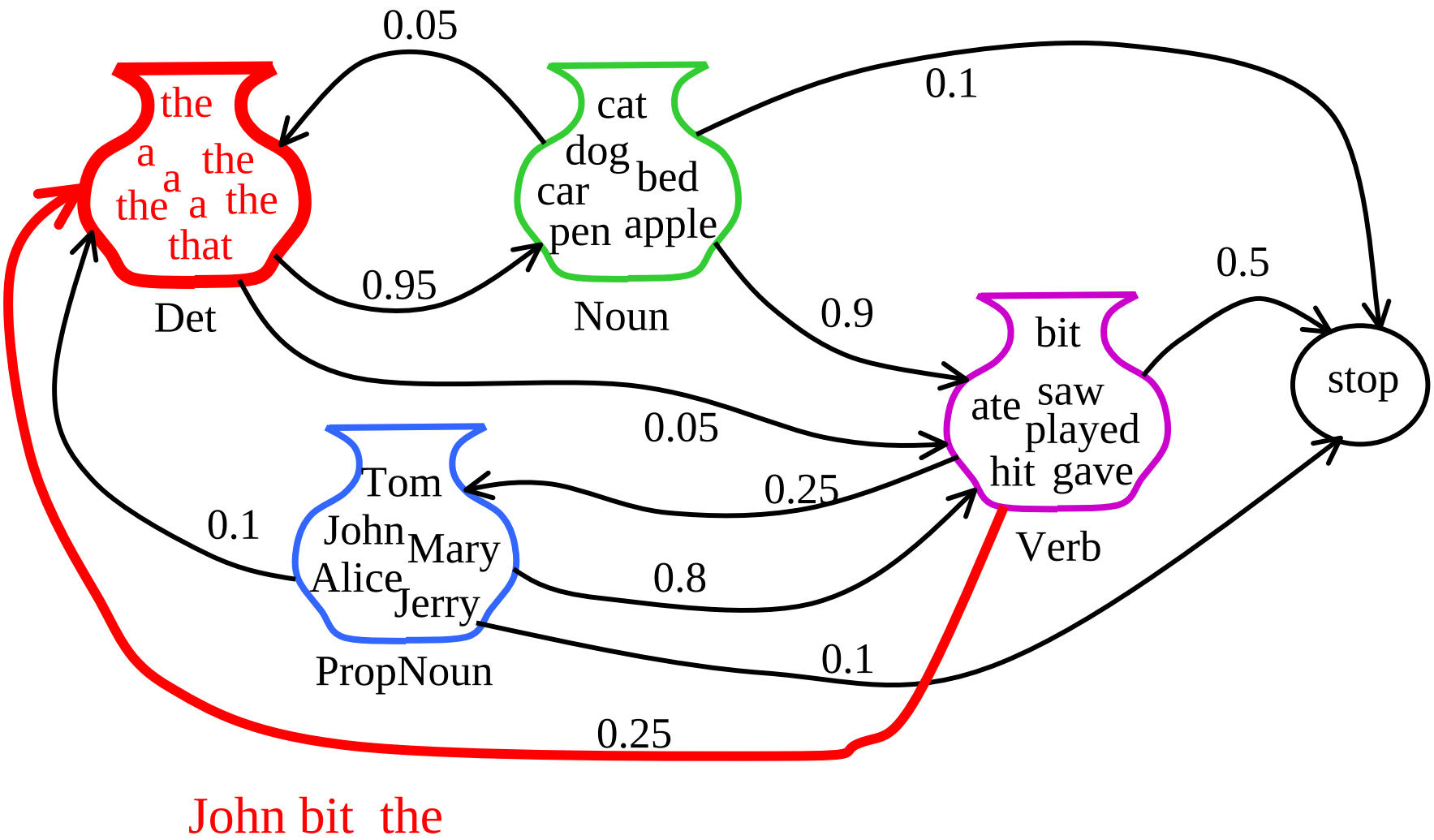
John

Sample HMM Generation

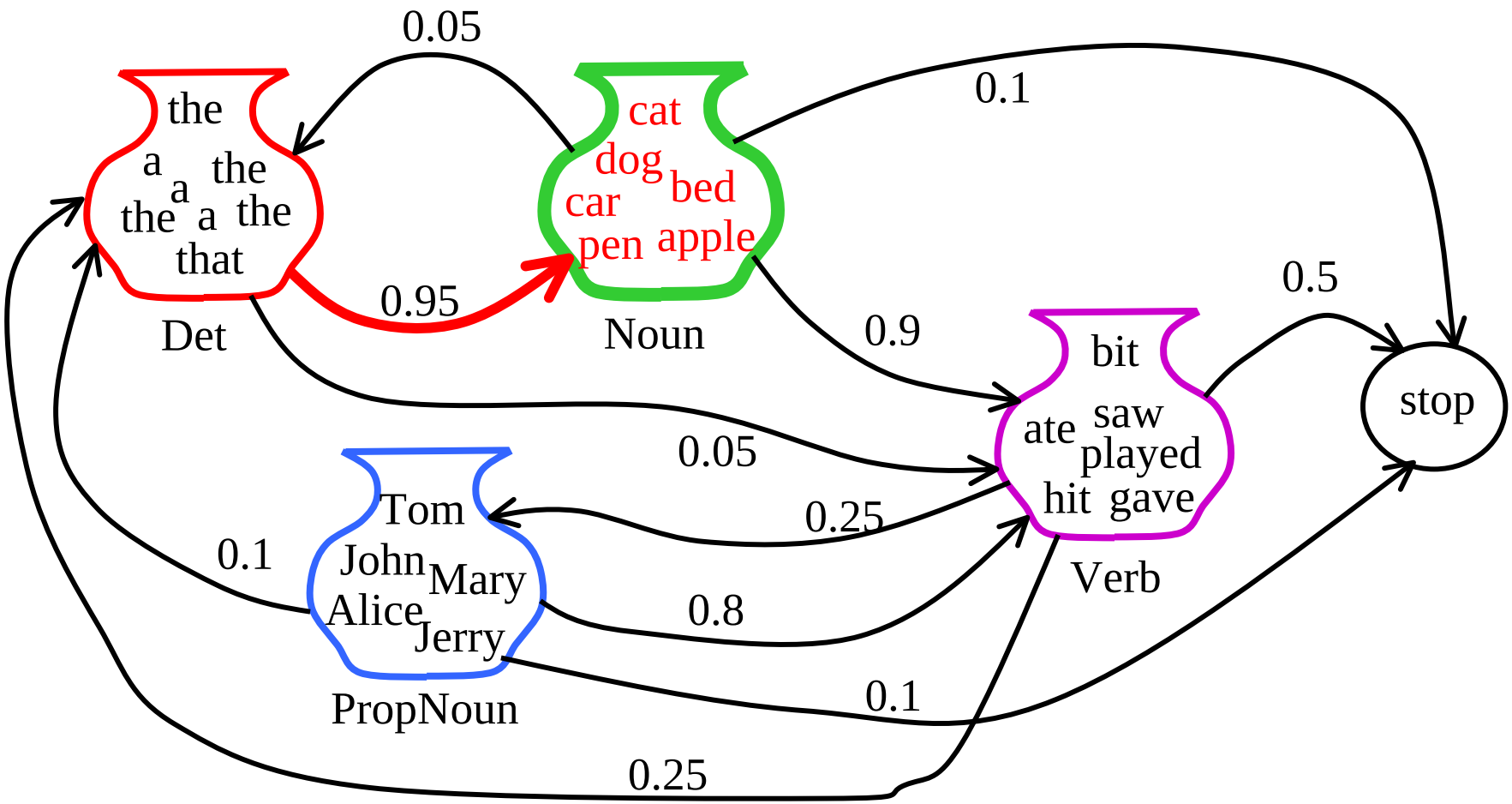


John bit

Sample HMM Generation

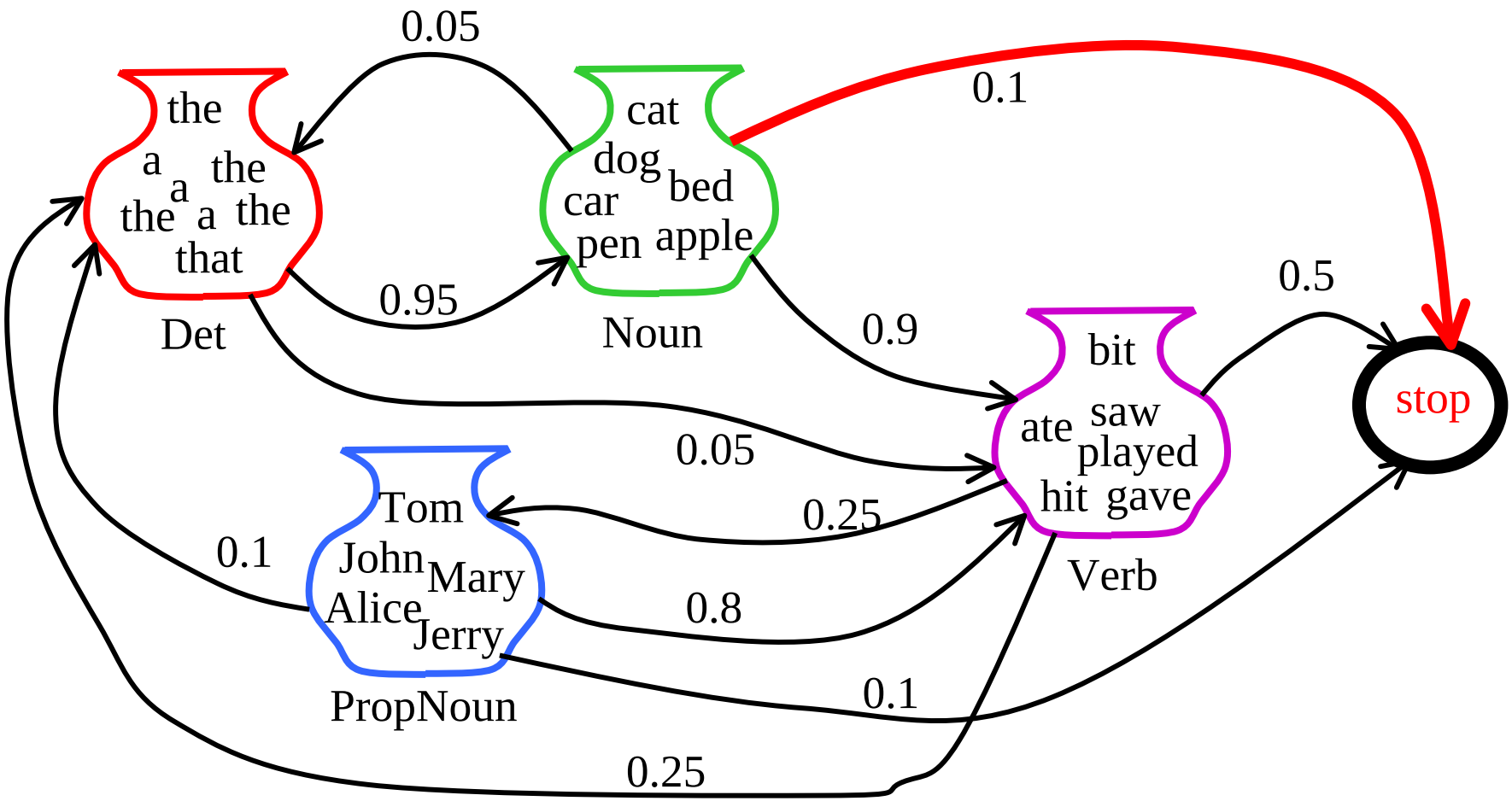


Sample HMM Generation



John bit the apple

Sample HMM Generation



John bit the apple

Structure label Problem

- Parsing

Parsing Disambiguation

Example	董永 和 七仙女 的 父亲 董永 和 七仙女 的 孩子
Pattern	N 和 N 的 N

One pattern vs. two Structures

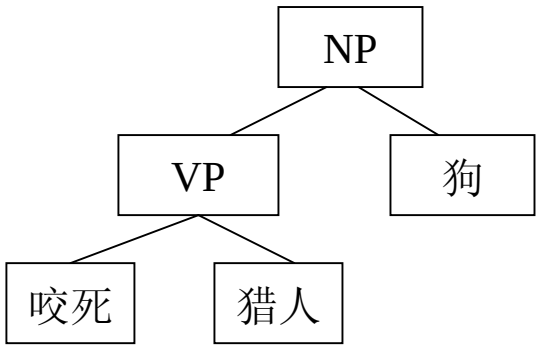
[N 和 [N 的 N]]

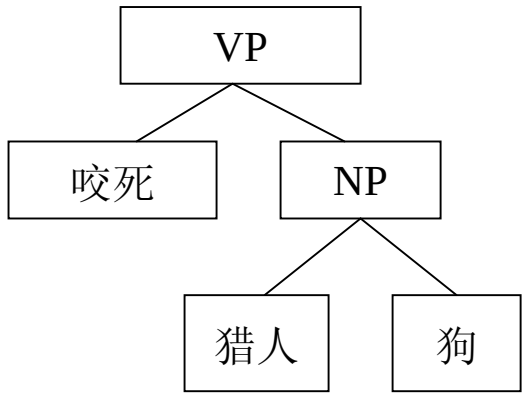
[[N 和 N] 的 N]

A famous example

I saw the boy with telescope
咬死猎人的狗 (two meanings)

VP → VP + NP | v
NP → NP + NP| NP + 的 + NP | VP + 的 + NP |n





ML is important in NLP?

- Relationships and correlations can be hidden within large amounts of data. Machine Learning may be able to find these relationships (e.g. word->POS, sentences->structure).
- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans(e.g. translation knowledge, parsing knowledge, etc.).
- Examples are easier to obtain than rules & Rule writers usually miss low frequency cases.
- Environments change over time(language evolve over time, or from one language to others).
- New knowledge about tasks is constantly being discovered by humans. It may be difficult to continuously re-design systems “by hand”

Outline

- Why is ML important
- **General Frame**
- Generalization & Over-fitting
- Experimental Evaluation

Machine Learning

The definition: Machine learning is programming computers to optimize a performance criterion using example data or past experience.

- Obtaining a description of the concept in some **representation** that explains observations and helps predicting new instances of the **same distribution**

"Learning denotes changes in the system that ... enable the system to do the same task ... more effectively the next time."

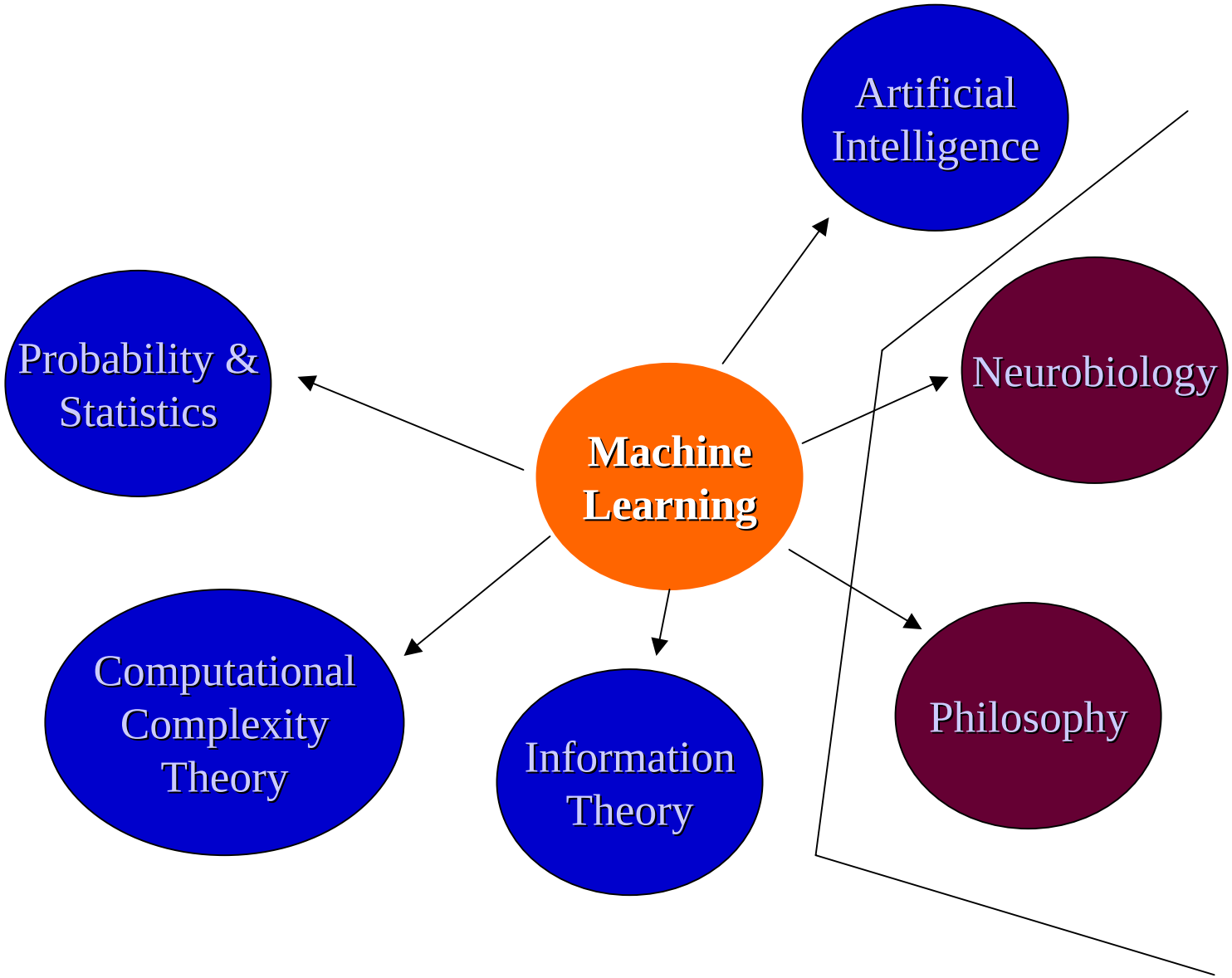
- Herbert Simon

"Learning is making useful changes in our minds."

- Marvin Minsky

A Distribution Assumption

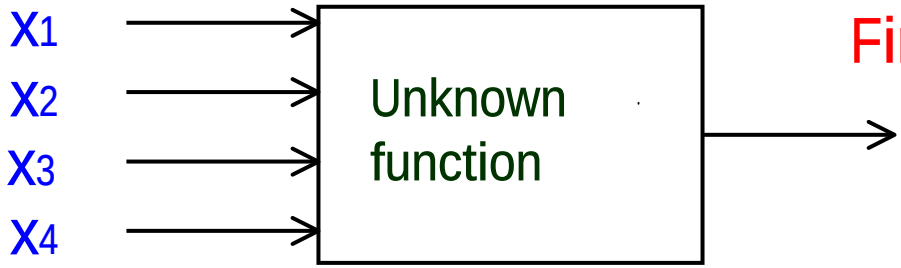
- Assuming that both training and test samples are generated from the same distribution *P(x; y)*
- *P(x; y)* is fixed, but **also unknown**
- ***Crucial point***: both training and test samples are drawn from the same distribution *P(x; y)*. This allows us to learn properties/functions from the training data which can be used to predict new, test examples
- IID: *independently identically distributed*


Artificial Intelligence
Probability & Statistics
Neurobiology
Machine Learning
Computational Complexity Theory
Information Theory
Philosophy

Machine Learning

- Three aspect:
  - What is the target?
  - The representation?
    - Data
    - Target function
  - Algorithms: Machine Learning Methods

A Learner



Find function $h \in H$

$y = h(x_1, x_2, x_3, x_4)$

features

Example	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

$y = h(x_1, x_2, x_3, x_4) = \neg x_2 \wedge x_4$?

Three Spaces

- **Input Space** – space used to describe each instance; often
  - Continuous: $X \subseteq \Re^n$;
  - Discrete (ordered and unordered): $X \subseteq N^n$;
  - Binary $X \subseteq \{0,1\}^n$;

Three Spaces

- **Output Space** – space of possible output labels; very dependent on problem
  - Continuous vs. discrete
  - Binary vs. multivalued
- **Hypothesis Space** – space of functions that can be selected by the machine learning algorithm (it is the set of all functions h that satisfy the goal).

A General Framework for Learning

- **Input space/feature space**: $X \subseteq \mathrm{R}^n$
  instance, x $\in X$, **x** $= (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n)$
- **Output space**: $Y = \{\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_l\}$
- **Example**: $(\mathbf{x}, y)$ with $\mathbf{x} \in X, y \in Y$
- **Training set**: a set of m examples D, generated i.i.d. according to an unknown real world **distribution** $P(\mathbf{x}, y)$:

$$\mathrm{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\} \subseteq (X \times Y)$$

A General Framework for Learning

- **Goal**: find a function ***h*** belonging to H, the **hypotheses space**, such that the expected error on new examples, is minimal

- What do we want ***h*(x)** to satisfy for real date?
  - Minimize the Loss (Risk):

$$R[h] = E(Loss(h(X),Y))$$
$$= \int Loss(h(X),Y)dP(X,Y)$$

  - Where,Loss Functions might be:

$$Loss(h(X),Y) = \begin{cases} 1, & h(X) \neq Y \\ 0, & h(X) = Y \end{cases}$$ Discrete Loss Function: 0-1 Loss

$$Loss(h(X),Y) = (h(X) - Y)^2$$ Squared Loss Function

A General Framework for Learning

- Minimize the Loss for **real data:**(Risk Function)

$$\min_h R[h] = E(Loss(h(X),Y)) = \int Loss(h(X),Y)dP(X,Y)$$

- However, we cannot do that.
- Instead, we try to minimize the **empirical classification error**.
- For a set of training examples $\{(x, y)_i\}_{i=1,m}$
- Try to minimize:

$$R_{emp}[h] = 1/m \sum_{i=1}^{m} \text{loss}\left(h(\mathbf{x}_i), y_i\right)$$

This is the average loss on the training samples

Learning=Minimizing Risk Function

- **Empirical Risk Minimization:**

$$\hat{h} = \underset{h}{\text{Argmin}} \quad R_{emp}[h]$$

- **Structure Risk Minimization:**

$$\hat{h} = \underset{h}{\text{Argmin}} \ (R_{emp}[h] + \lambda Q(h))$$

3 Problems

Accordance assumption

m examples chosen i.i.d. according to an unknown real world **distribution**

Modeling

For a **hypothese**, estimate the parameters based on training data and minimize loss

Generalization

Accuracy on real data, ie: training + test

types

Goal: find a function **h**: **Y=h(X),** where, D={(x,y)|x $\in X$, $y \in Y$} is training sample space.

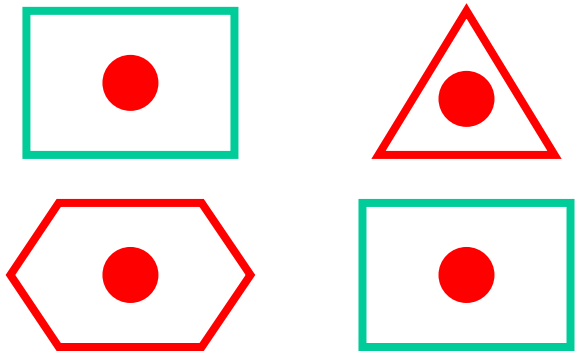
1. Y=∅： unsupervised learning;
2. Y is a set of integer： classification;
3. If |Y|=2, h is called a concept, and learning is **concept learning**
4. Y is a set of real: regression
5. Y are not given for some Ds： semi-supervised learning ;
6. Y is order set： Learning for Ranking;
7. …

Learning Paradigms

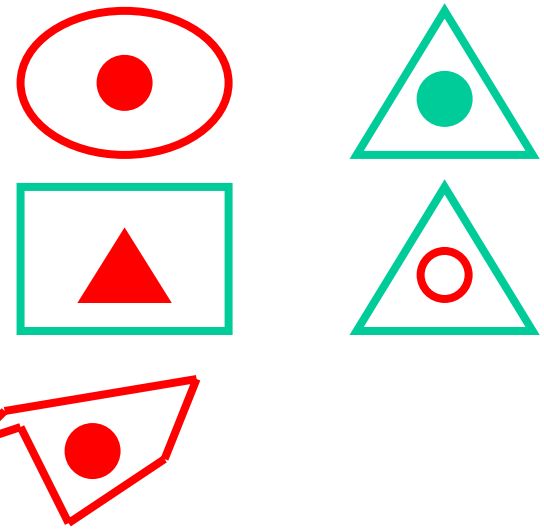
- **Inducing Functions from I/O Pairs**
  - Decision trees (e.g., Quinlan's C4.5 [1993])
  - Connectionism / neural networks (e.g., backprop)
  - Nearest-neighbor methods
  - Genetic algorithms
  - SVM's
  - Bayesian Methods
- **Learning without a Teacher(Unsupervised)**
  - clustering
  - others

Example

Positive Examples



Negative Examples

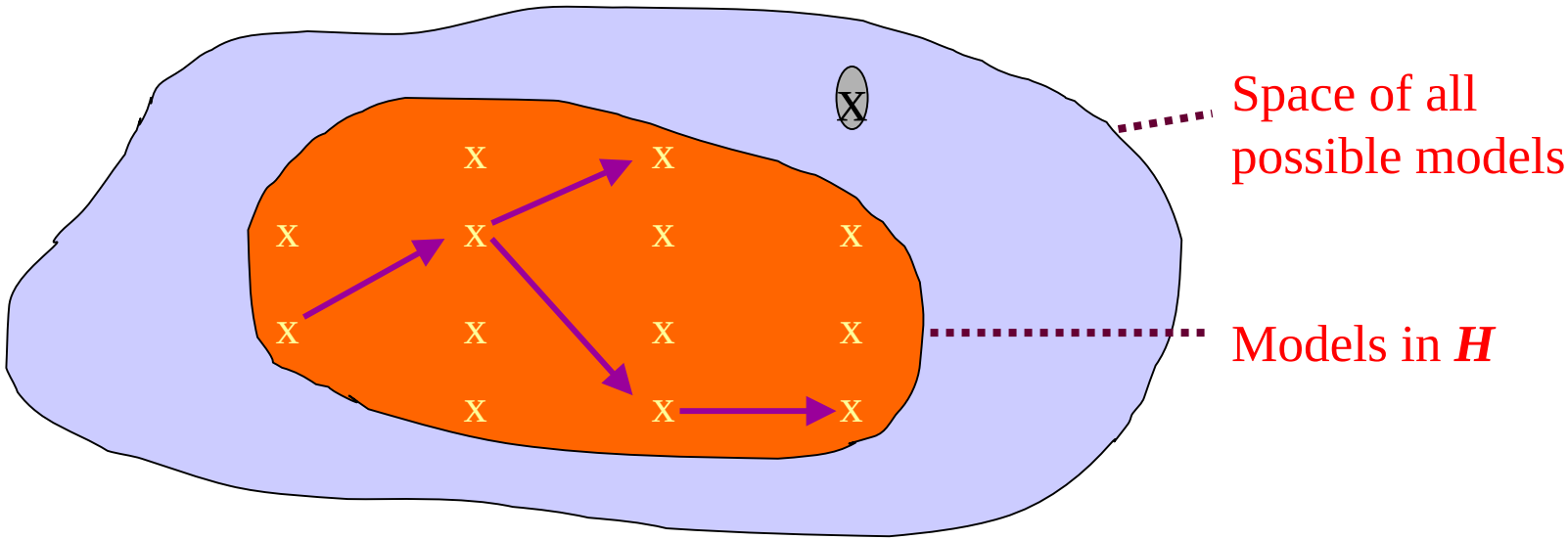


How does this symbol classify?

- Concept
  - Solid Red Circle in a (Regular?) Polygon

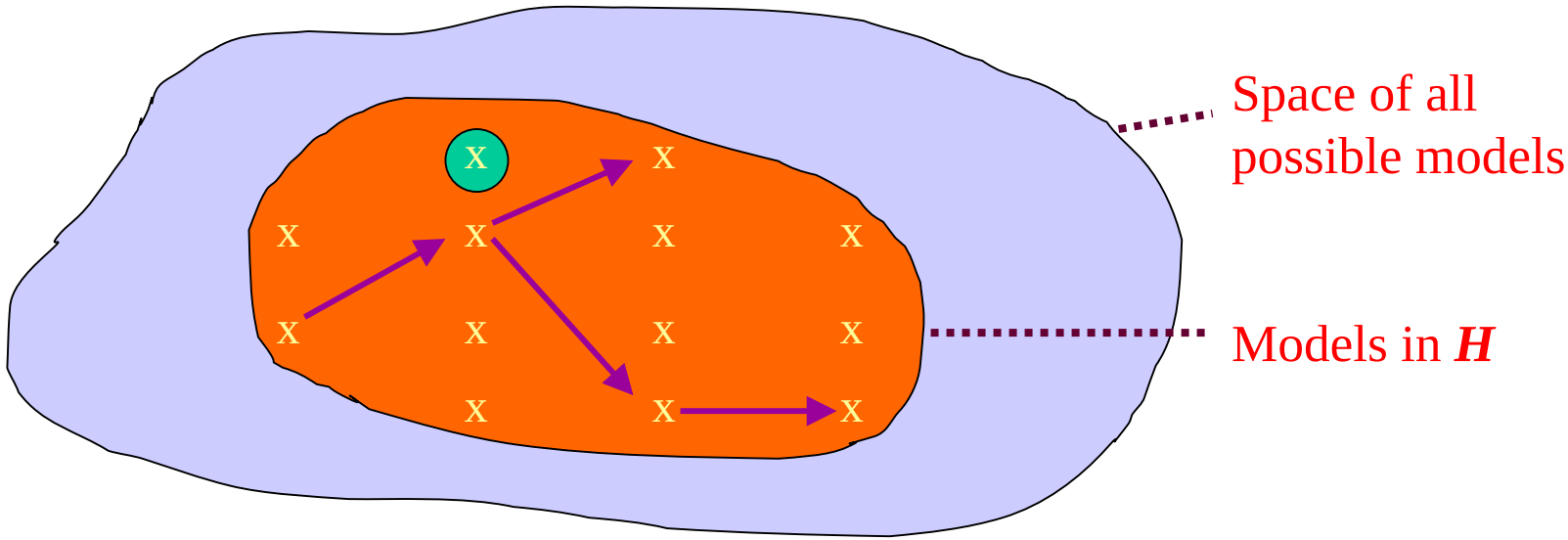
How can ML methods fail?

- Wrong Bias: best hypothesis is not in *H*
- Example:
  - In concept learning, the target function is usually represented as conjunctive descriptions, but in fact disjunctive descriptions should also be considered!
  - we look for hypothesises expressible as discrete decision trees but the domain is continuous



How can ML methods fail?

- Search Failure: best model is in *H* but search fails to examine it
- Example: greedy search fails to capture a strong effect



Inductive bias

- The constraints on the hypothesis space *H* is called the ***inductive bias***.
- Why ***inductive bias:***
  - *H* might contains an ***infinite*** number of functions that satisfy the goal;
  - It is necessary to find a way to constrain the search space of *h*
- There are two types of inductive bias:
  - The ***hypothesis space restriction bias***
  - The ***preference bias***

Outline

- Why is ML important
- General Frame
- ➢ **Generalization & Over-fitting**
- Experimental Evaluation

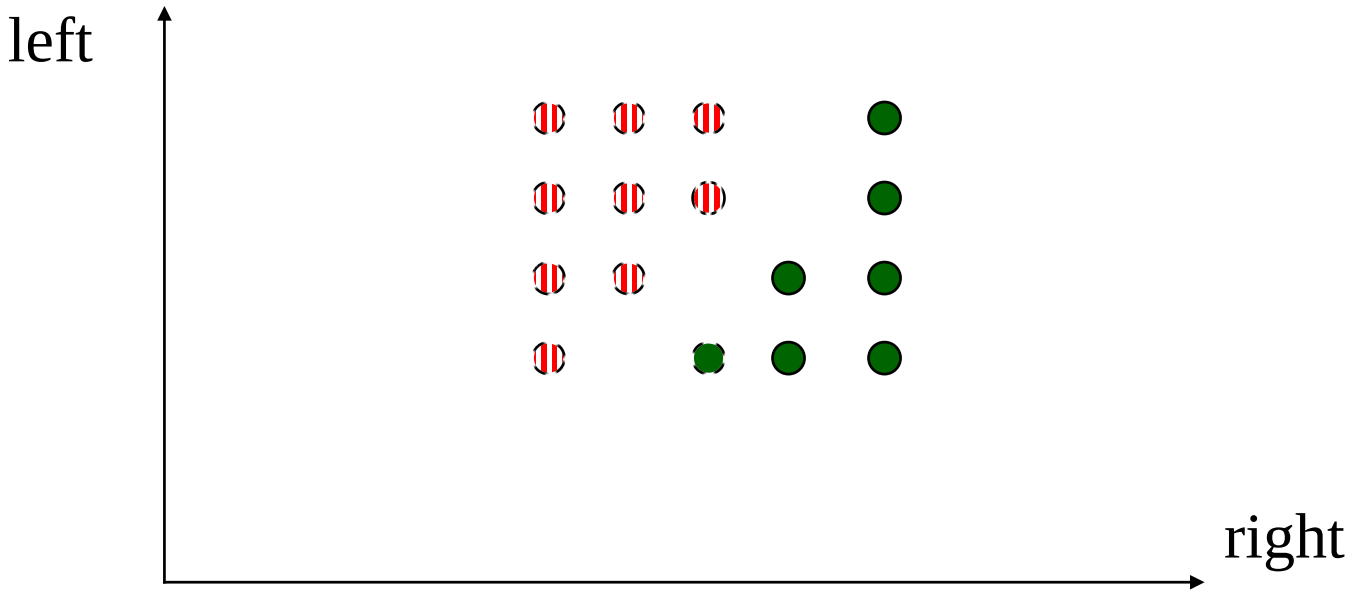
Generalization

- A good learning program learns something about the data beyond the specific cases that have been presented to it
- Classifier can minimize “i.i.d.” error, that is error over future cases (not used in training). Such cases contain both previously encountered as well as new cases.
- That is:

$$\min_h R[h] = E(Loss(h(X),Y)) = \int Loss(h(X),Y)dP(X,Y)$$

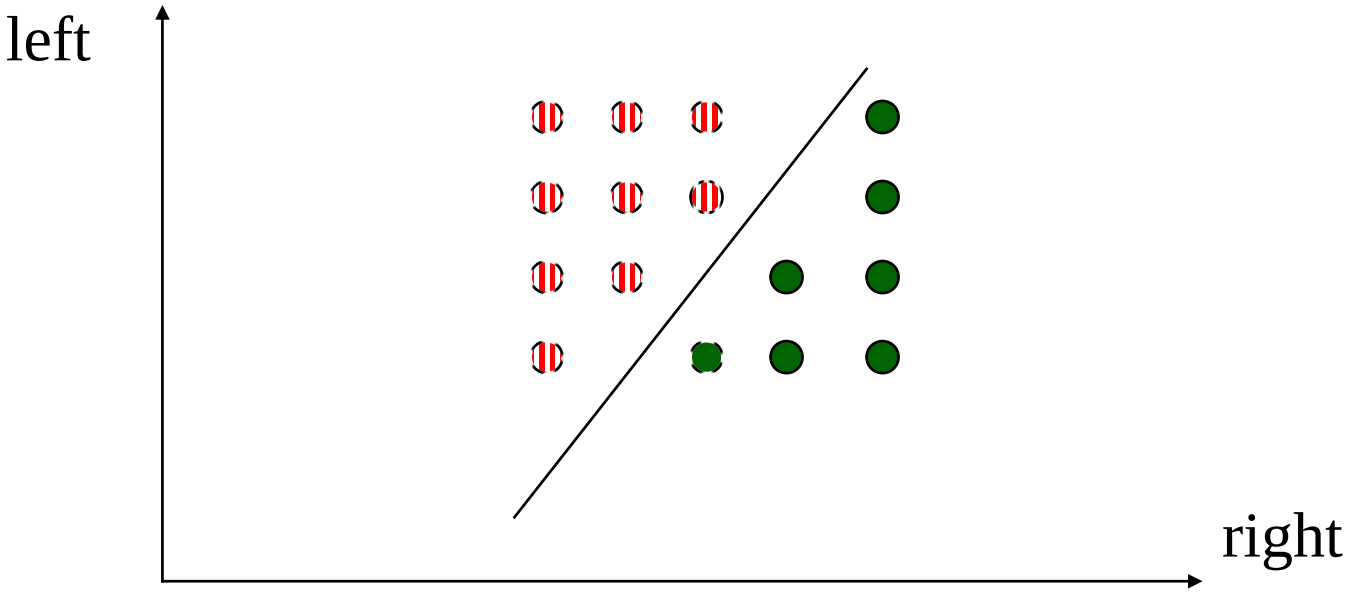
An Example

- Metaphor Recognition: classify a word in Natural Language as metaphor (**red/vertical pattern**) or not (**green**) based on the left word and the right one.



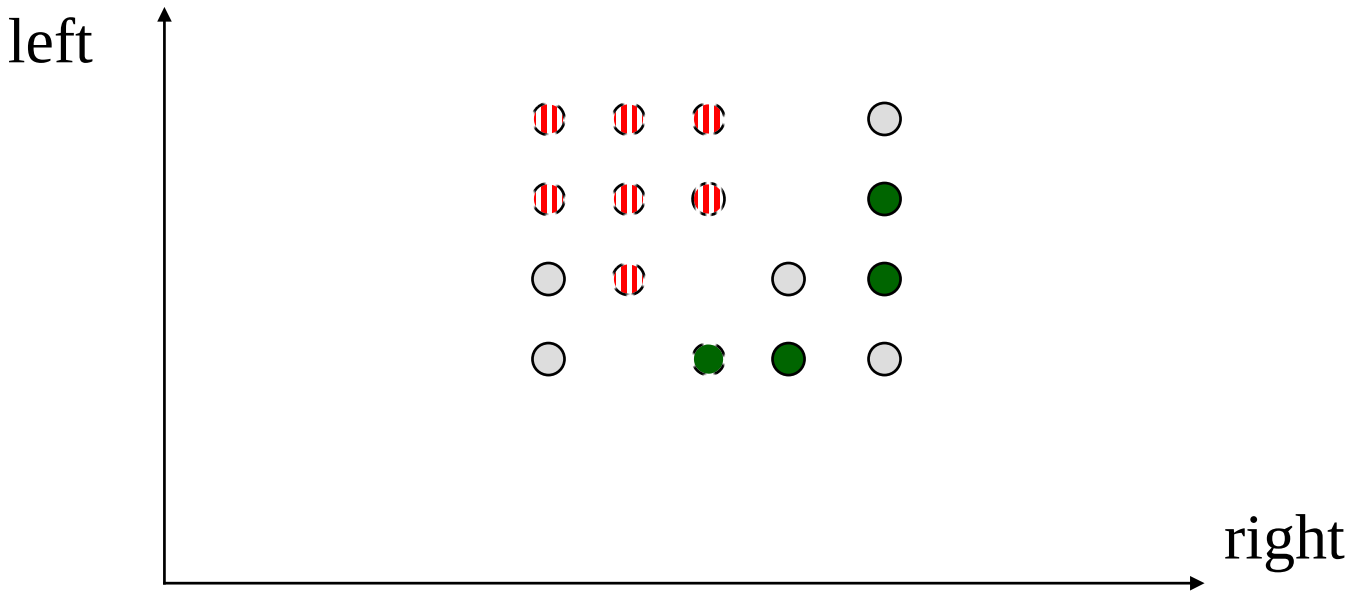
An Example

- Metaphor Recognition: classify an object as ‘A’ (red/vertical pattern) or ‘B’ (green) based on **the left and the right**.
- The line represents a perfect classifier for this problem:



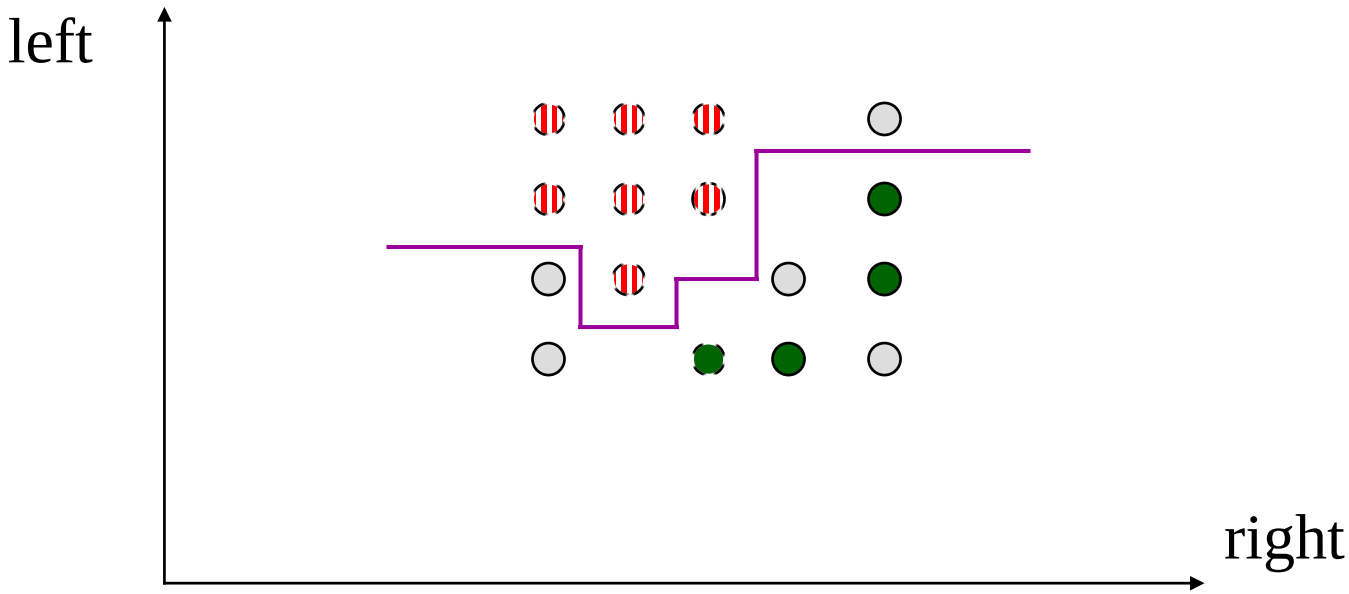
Train on a small sample

- One possible small sample is:
- Notice: grey points are not given in training set



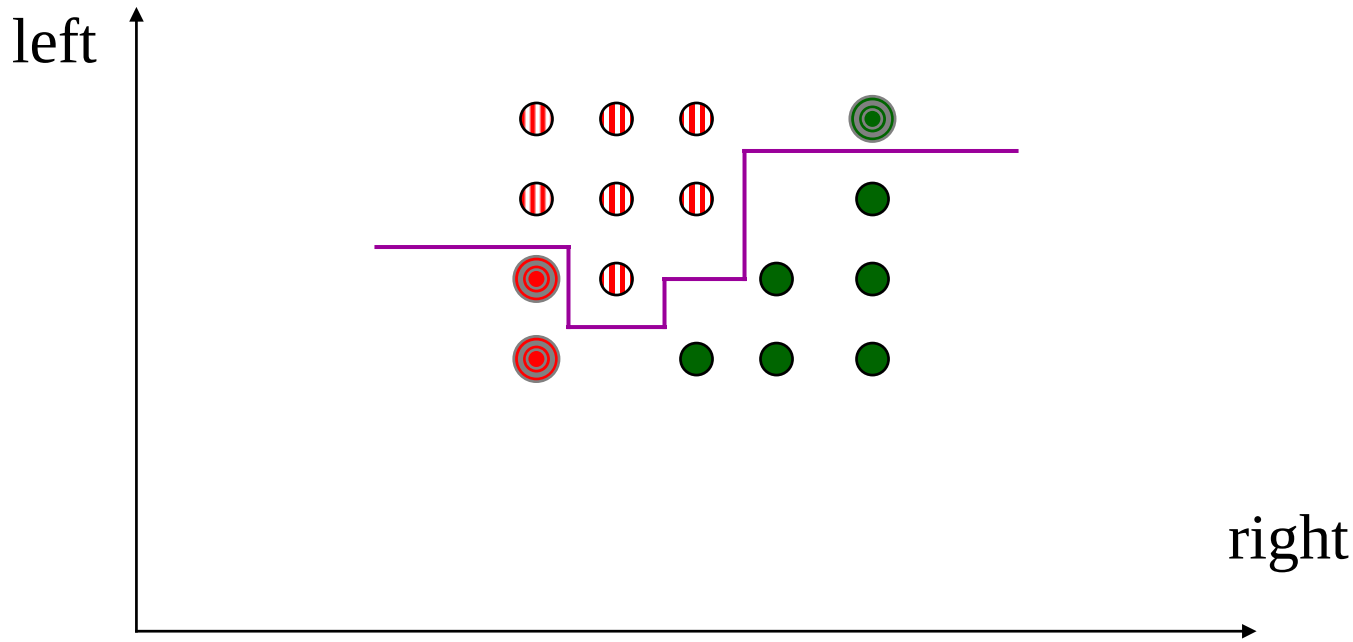
A solution

- A fairly complicated line can be learned as follows:



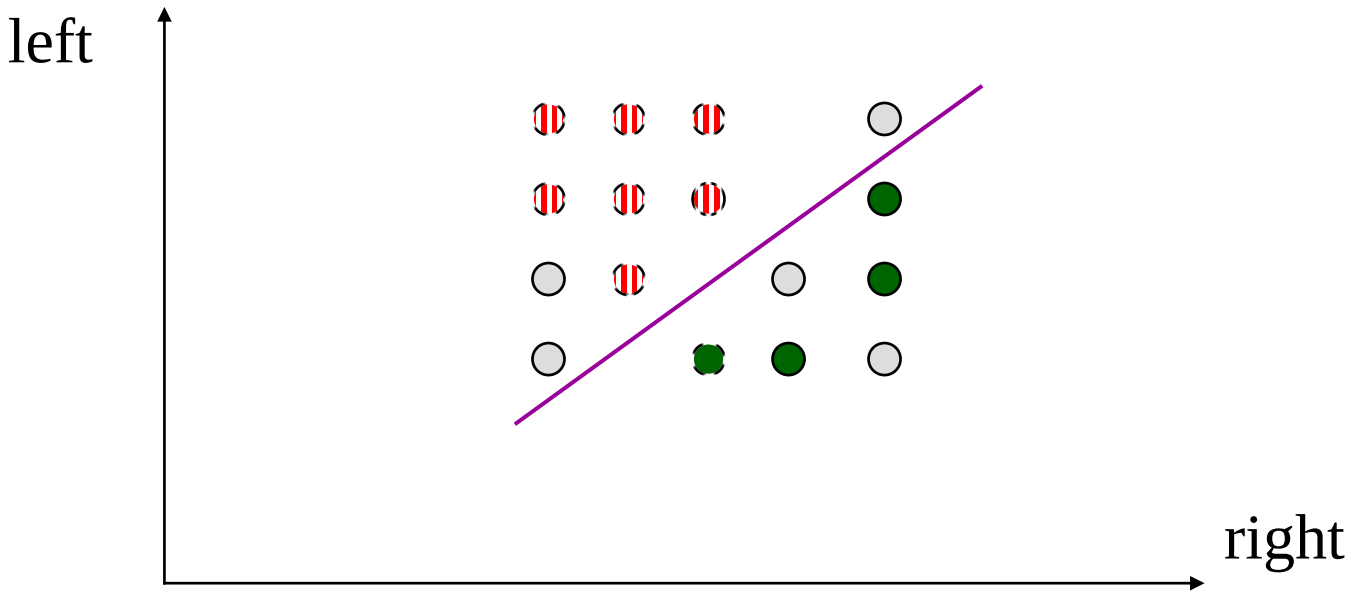
Errors

- Several errors occur:



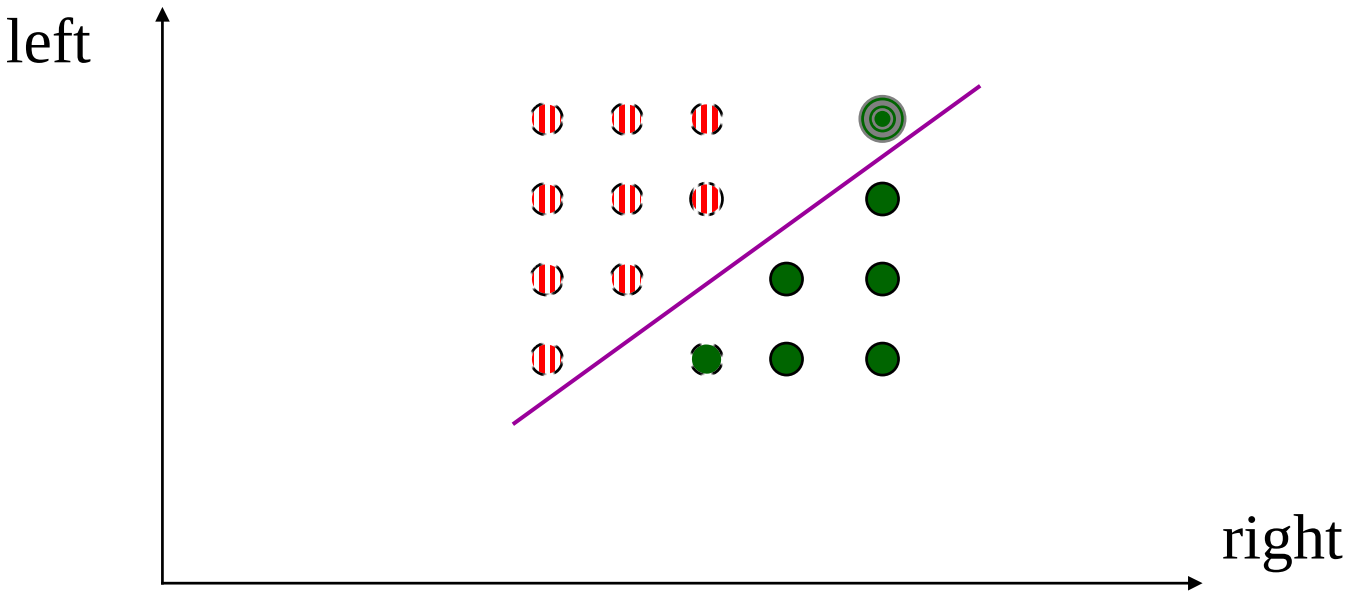
Another Solution

- A simple line can be learned as follows:



Errors

- A much small errors:



Generalization & Over-fitting

- In general, over-fitting a model to the data means that we learn non-representative properties of the sample data;
- Over-fitting and poor generalization are synonymous as long as we have learned the training data well.
- Over-fitting is not only affected by the “simplicity” of the classifier (e.g., straight vs wiggly line) but also by:
  - the size of the sample,
  - the complexity of the function we wish to learn from data,
  - the amount of noise, and
  - the number of the variables.

Avoid over-fitting

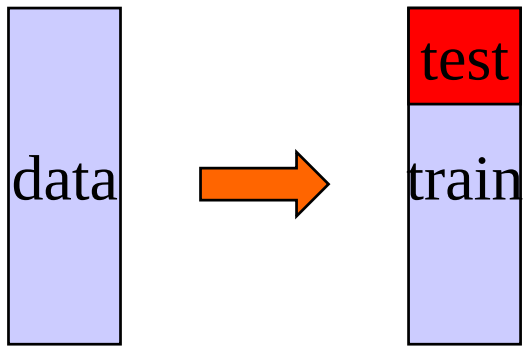
- Via a variety of approaches:
  - Use learning algorithms that intrinsically (by design) generalize well
  - Pursue simple classifiers for small samples
  - …

Outline

- Why is ML important
- General Frame
- Generalization & Over-fitting
- **Experimental Evaluation**

Hold-out Validation

- Split data into Train and Test data(usually 2/3 for training and 1/3 for testing);
- Problem: the samples might not be representative. For example, some classes might be represented with very few instance or even with no instances at all;
- Solution: ***stratification*** - sampling for training and testing within classes. This ensures that each class is represented with approximately equal proportions in both subsets



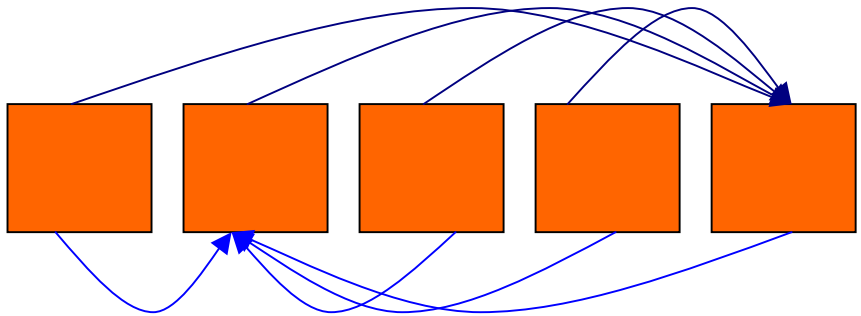
Repeated Hold-out

- **Idea**: repeat hold-out process with different subsamples
- **Step**: repeat m times
  - A certain proportion is randomly selected for training (possibly with stratification) while remaining for testing;
  - The accuracy on the different iterations are averaged to yield an overall accuracy rate.
- **Problem**: the different test sets may overlap

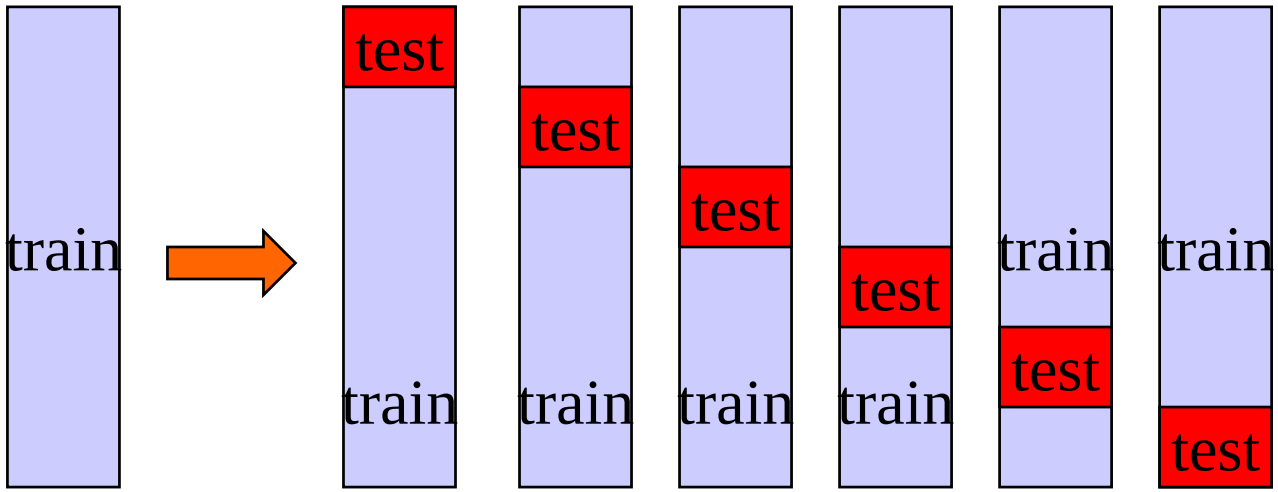
N-Fold Cross-Validation

- Start with a dataset of labeled example **D**;
- Randomly partition into *N* groups: **N-fold**; $P_1 \ldots P_N$
- For *i* from 1 to *N* do :
  - $S_i = (D - P_i)$: **train set**;
  - $h = L(S_i)$: train classifier on train set S_i;
  - $\text{error}_{Pi}(h)$: measure accuracy on **test set** P_i
- Average *n* errors.

$$Er = \frac{1}{N}\sum_{i=1}^{N} error_{Pi}(h)$$



Cross-Validation



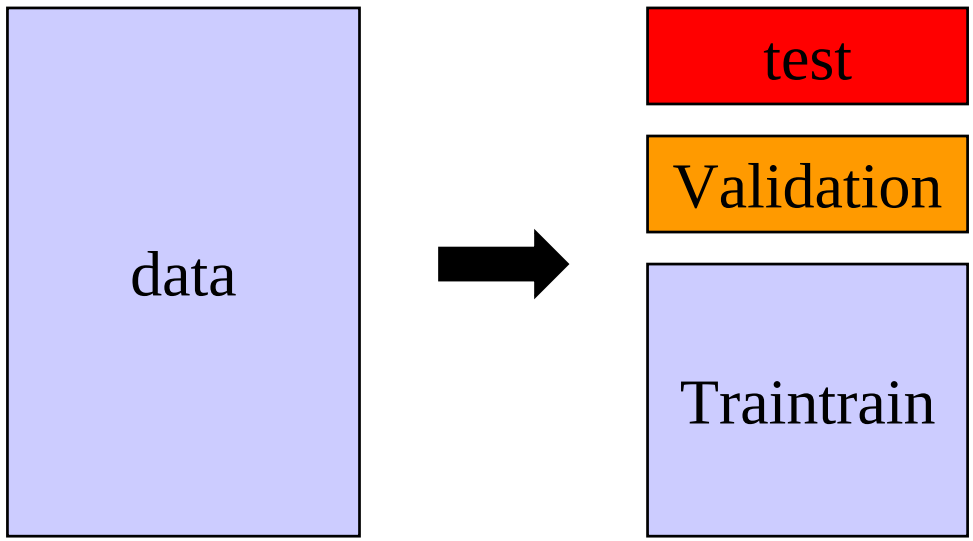
Hold-one-out Cross-Validation

(LOO CV)

- LOO CV is a *N*-fold cross-validation, where *N* is the number of training examples;
- Every examples gets used as **a test example** exactly **once** and as a training example N-1 times;
- No random subsampling is involved ;
- Problems:
  - LOO CV is very computationally expensive;
  - Stratification is not possible. only one example for testing).
- Worst case example: assume a *completely random* dataset with two classes each represented by 50% of the instances. The best classifier for this data is the majority predictor. LOO CV will predict 100% error (!) rate for this classifier (explain why?).

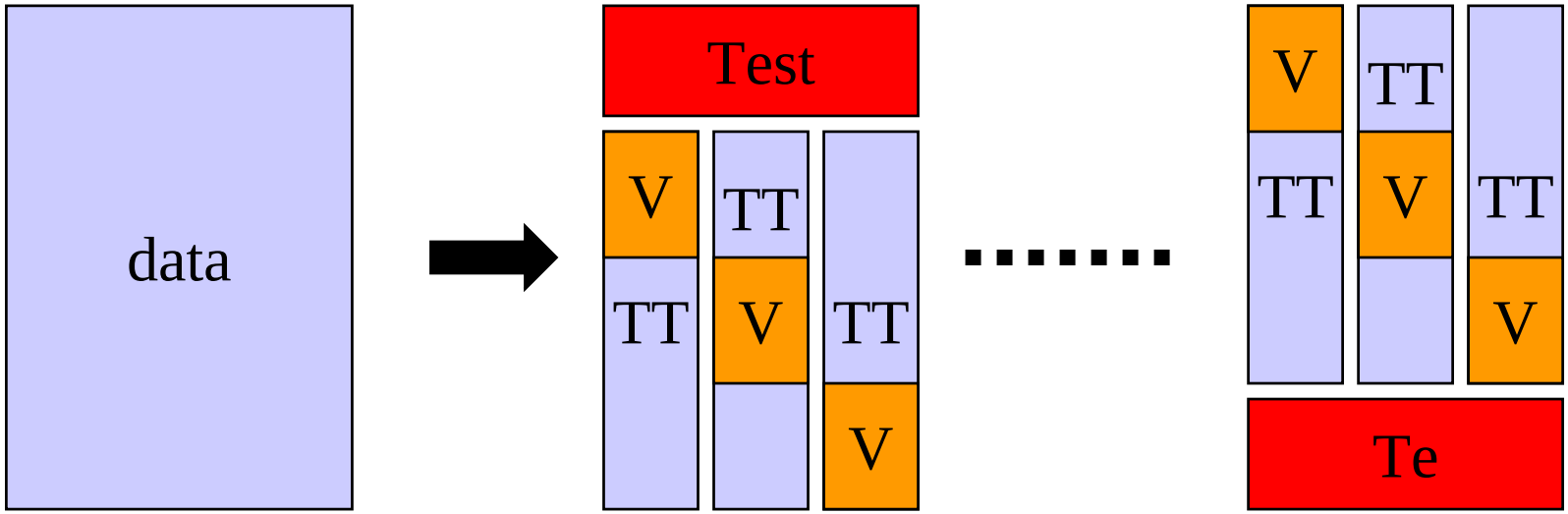
Train-Validation -Test

- Split the Train data into two (Traintrain and Validation);
- Use the validation set to find the best parameters;
- Use the test set to estimate the true error.



Nested N-Fold CV

- If the sample is small, the nesting can be repeated with different assignment of the test set (i.e., nested n-fold C.V.).



Bootstrapping

- CV uses sampling without replacement. That is, the same example, once selected, can not be selected again for a particular training/test set;
- The bootstrap is an estimation method that uses sampling with replacement to form the training set;

The general bootstrap algorithm

Let the original sample be $S=(x_1,x_2,\ldots,x_n)$

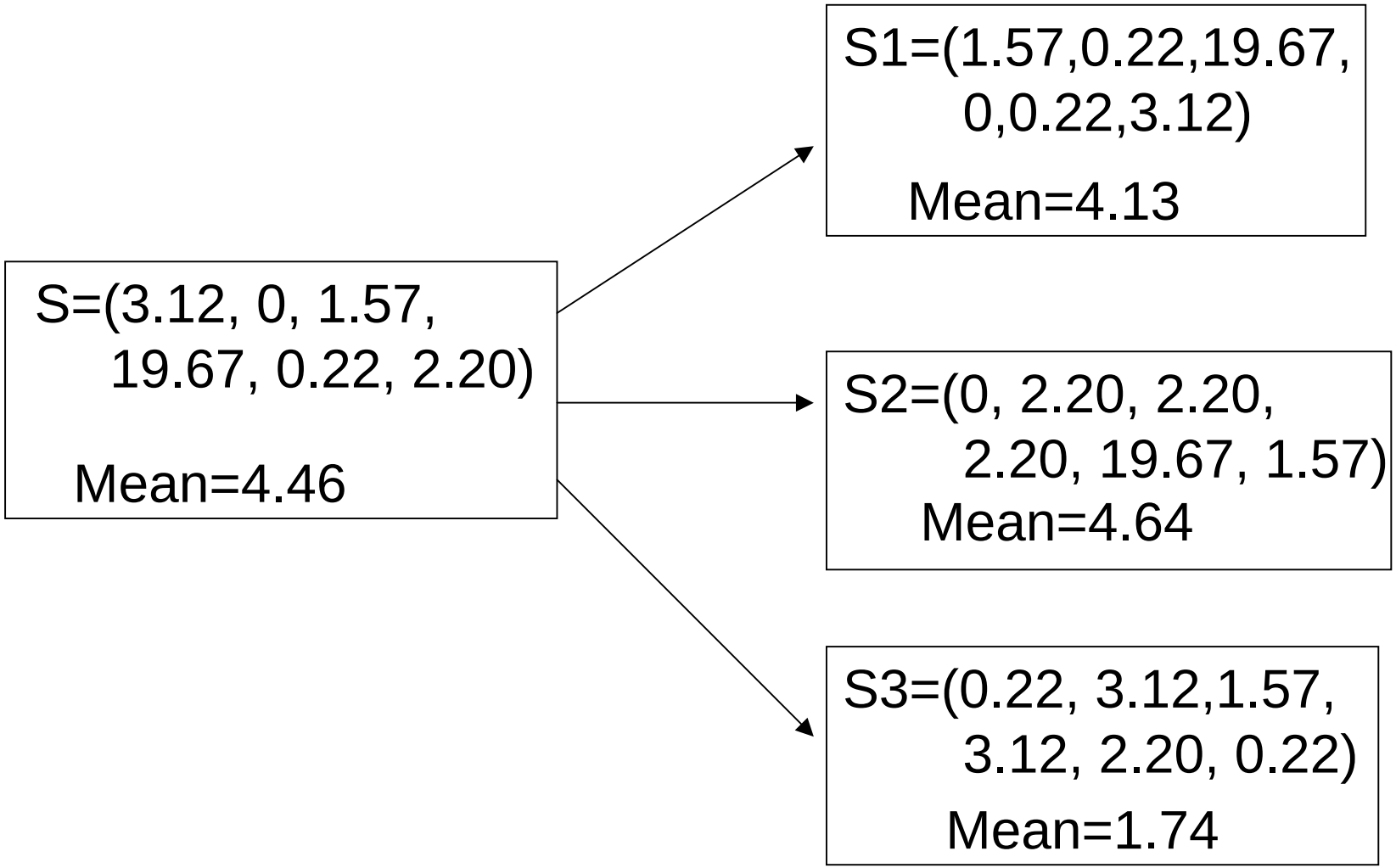
- Repeat B time:
  - Generate a sample S_k of size n from S by sampling with replacement.
  - Compute $\hat{\theta}*$ for x*.

➔ Now we end up with bootstrap values

$$\hat{\theta}* = (\hat{\theta}_1^*, ..., \hat{\theta}_B^*)$$

- Use these values for calculating all the quantities of interest (e.g., standard deviation, confidence intervals)

An example


S=(3.12, 0, 1.57,
19.67, 0.22, 2.20)
Mean=4.46
S1=(1.57,0.22,19.67,
0,0.22,3.12)
Mean=4.13
S2=(0, 2.20, 2.20,
2.20, 19.67, 1.57)
Mean=4.64
S3=(0.22, 3.12,1.57,
3.12, 2.20, 0.22)
Mean=1.74

Bootstrap distribution

- The bootstrap does not replace or add to the original data.
- We use bootstrap distribution as a way to estimate the variation in a statistic based on the original data.

Cases where bootstrap does not apply

- Too small data sets: the original sample is not a good approximation of the population
- Dirty data: outliers add variability in our estimates.
- Dependence structures (e.g., time series, spatial problems): Bootstrap is based on the assumption of independence.
- …

Bootstrapping for evaluation

- CV uses sampling without replacement. That is, the same example, once selected, can not be selected again for a particular training/test set;
- The bootstrap is an estimation method that uses sampling with replacement to form the training set;
- **Training set**: a dataset of *n* examples is sampled with replacement *n* times with replacement to form the training set of *n* examples (possibly with repetitions);
- **Test set**: the examples from the original dataset that don't occur in the training set.

0.632-bootstrap

- A particular example has a probability of (1-1/n) of not being selected for the training set. Thus, an instance will fall in the test set with probability:

$$\lim_{x \to \infty} (1 - \frac{1}{n})^n = \frac{1}{e} = 0.368$$

- This means that the training data will contain approximately 63.2% of the examples.

Evaluation Metrics

	Reference+	Reference-	
Testing+	a	b	a+b
Testing-	c	d	c+d
	a+c	b+d	a+b+c+d

- Accuracy (0/1 loss): $$\frac{\text{Number of correct classifications}}{\text{Number of total classifications}} = \frac{a+d}{a+b+c+d}$$
- Sensitivity: proportion of true positives identified : $\frac{a}{a+c}$
- Specificity: proportion of true negatives identified : $\frac{d}{b+d}$

Evaluation Metrics

- Positive predictive value (PPV): proportion of true positives over test positives : $\frac{a}{a+b}$
- Negative predictive value (NPV): proportion of true negatives over test negatives : $\frac{d}{c+d}$
  - "Precision" is the name for "PPV" and
  - "Recall" is the name for "Sensitivity"

Question

- How to represent the training data for the following problem:
  - Word Segmentation
  - POS tagging
  - Named Entities Recognition
  - WSD
  - Coreference Resolution