Machine Learning Department
School of Computer Science
Carnegie Mellon University

From Binary to Extreme Classification

Matt Gormley
Lecture 2
Aug. 28, 2019

Q&A

Q: How do I get into the online section?

A: Sorry! I erroneously claimed we would automatically add you to the online section. Here's the correct answer:

To join the online section, **email Dorothy Holland-Minkley at dfh@andrew.cmu.edu stating that you would like to join the online section.**

Why the extra step? We want to make sure you've seen the **non-professional video recording** and are okay with the quality.

Q&A

Q: Will I get off the waitlist?

A: Don't be on the waitlist. Just email Dorothy to join the online section instead!

Q&A

Q: Can I move between 10-418 and 10-618?

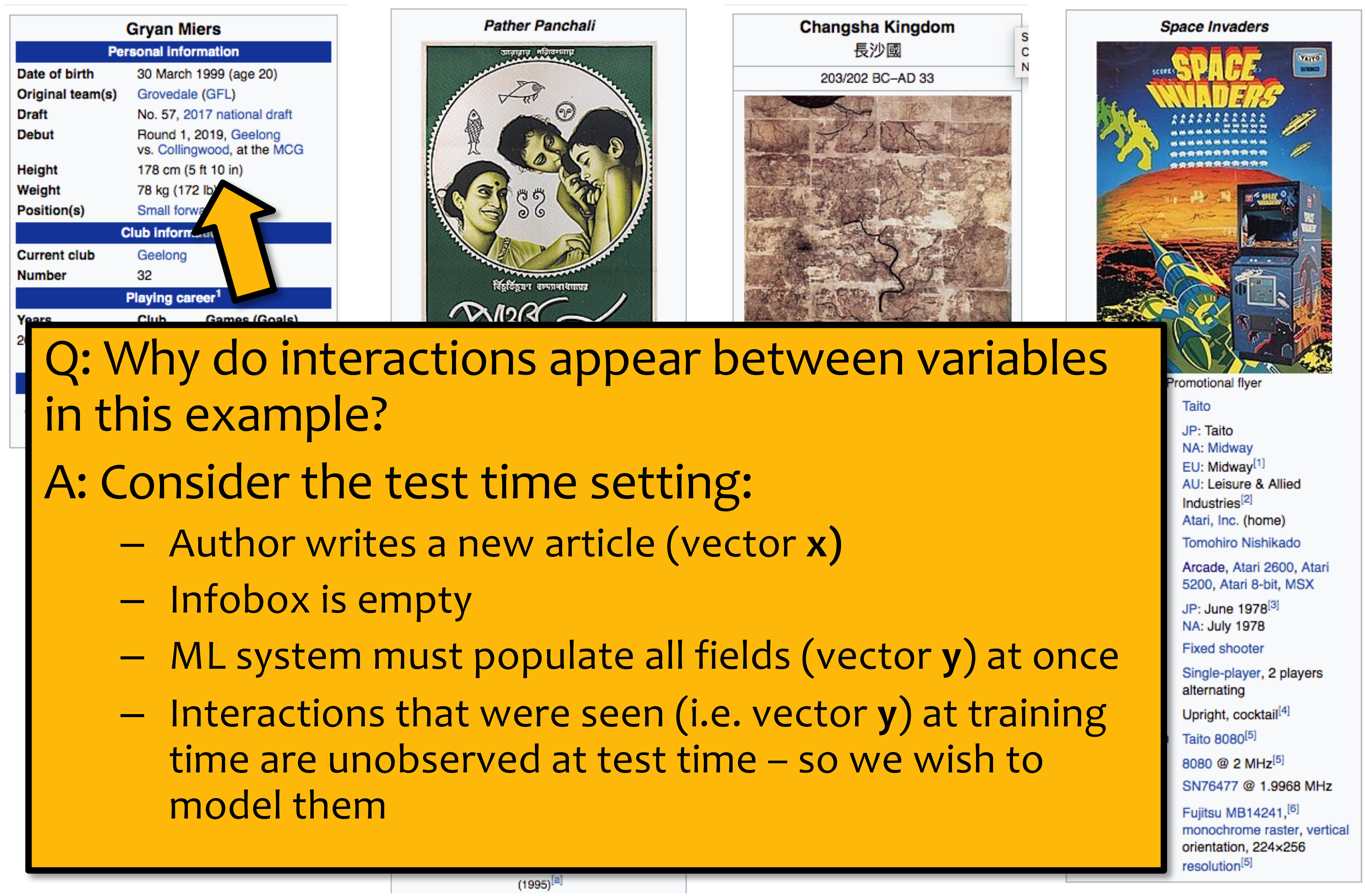
A: Yes. Just email Dorothy Holland-Minkley at dfh@andrew.cmu.edu to do so.

Q: When is the last possible moment I can move between 10-418 and 10-618?

A: I'm not sure. We'll announce on Piazza once I have an answer.

QnA

Populating Wikipedia Infoboxes



Q: Why do interactions appear between variables in this example?

A: Consider the test time setting:

- Author writes a new article (vector **x**)
- Infobox is empty
- ML system must populate all fields (vector **y**) at once
- Interactions that were seen (i.e. vector **y**) at training time are unobserved at test time – so we wish to model them

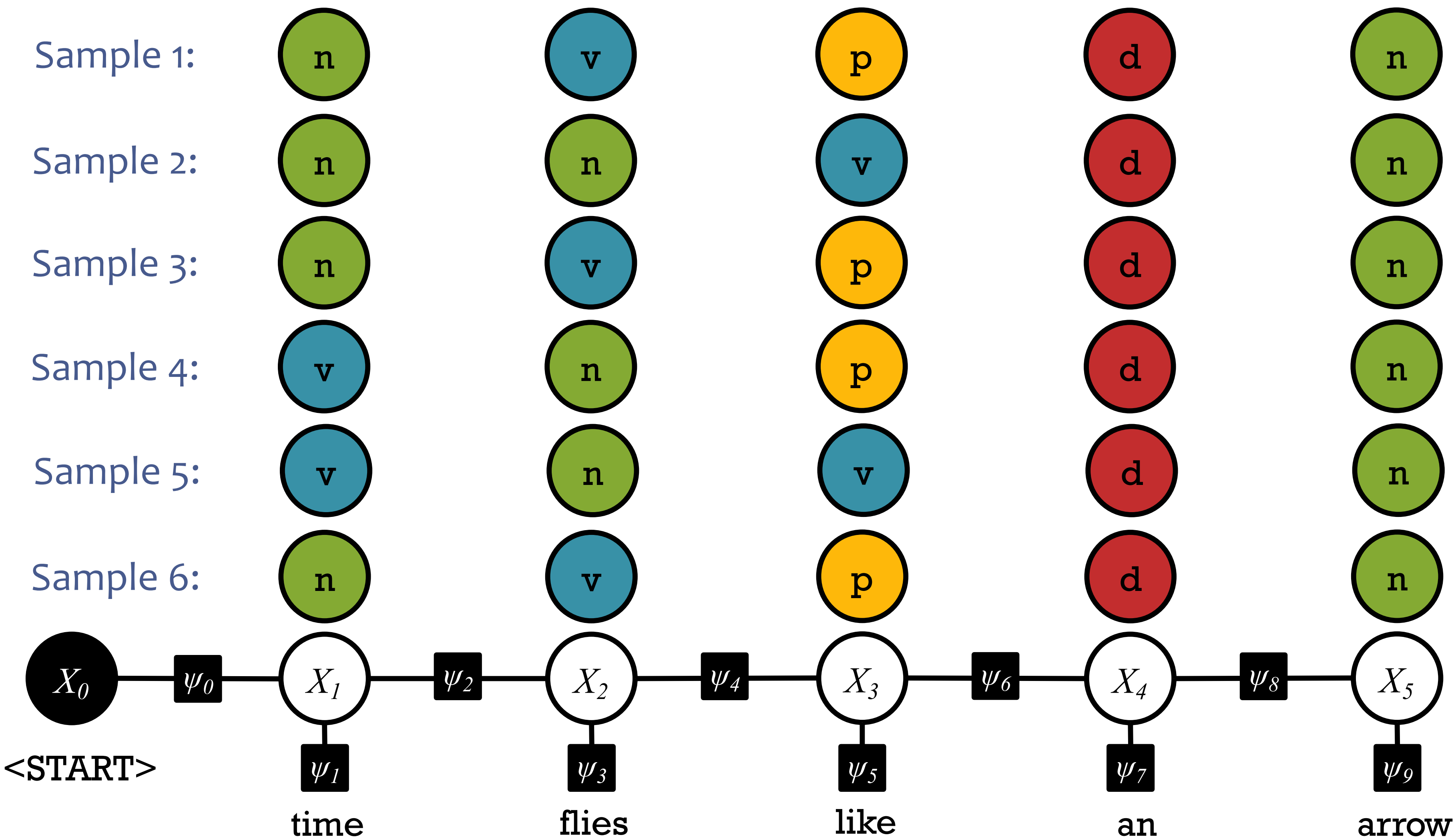
ROADMAP

How do we get from Classification to Structured Prediction?

1. We start with the simplest decompositions (i.e. **classification**)
2. Then we formulate structured prediction as a **search problem** (decomposition of into a sequence of **decisions**)
3. Finally, we formulate structured prediction in the framework of **graphical models** (decomposition into **parts**)

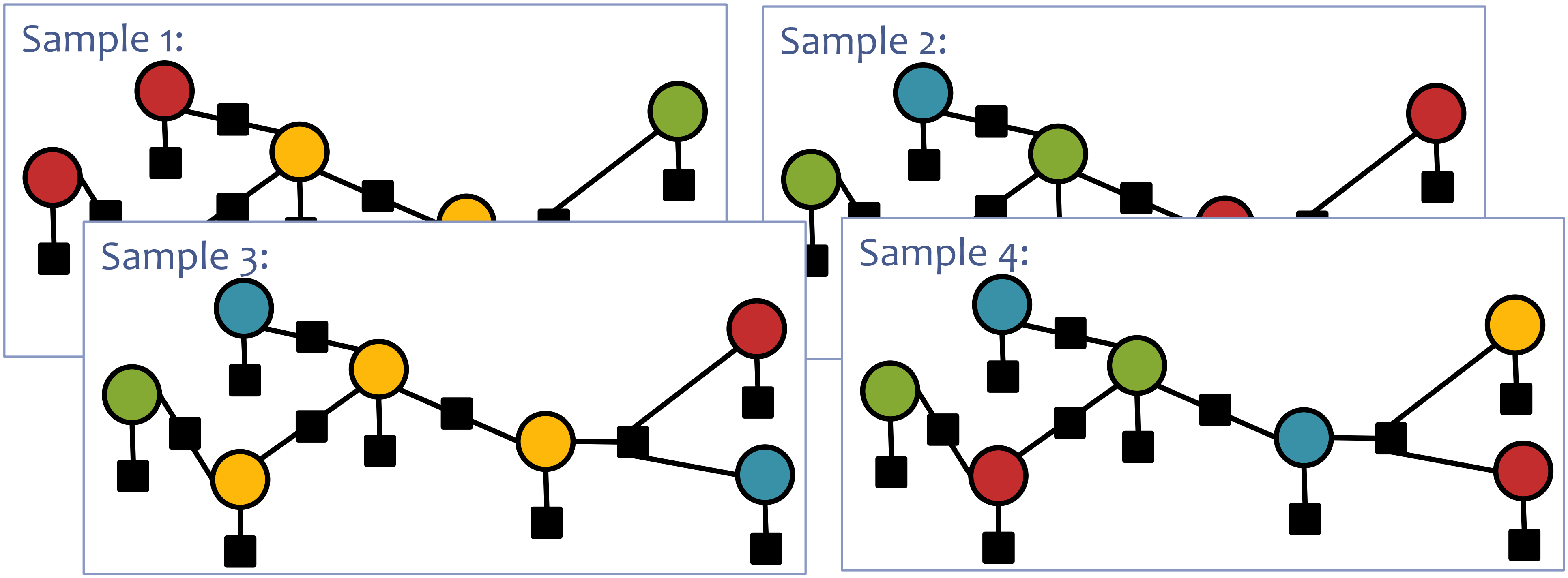
Sampling from a Joint Distribution

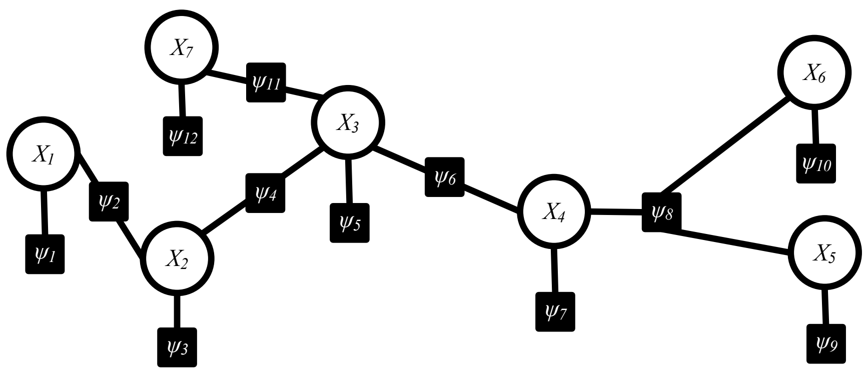
A **joint distribution** defines a probability $p(\boldsymbol{x})$ for each assignment of values $\boldsymbol{x}$ to variables $\boldsymbol{X}$. This gives the **proportion** of samples that will equal $\boldsymbol{x}$.



Sampling from a Joint Distribution

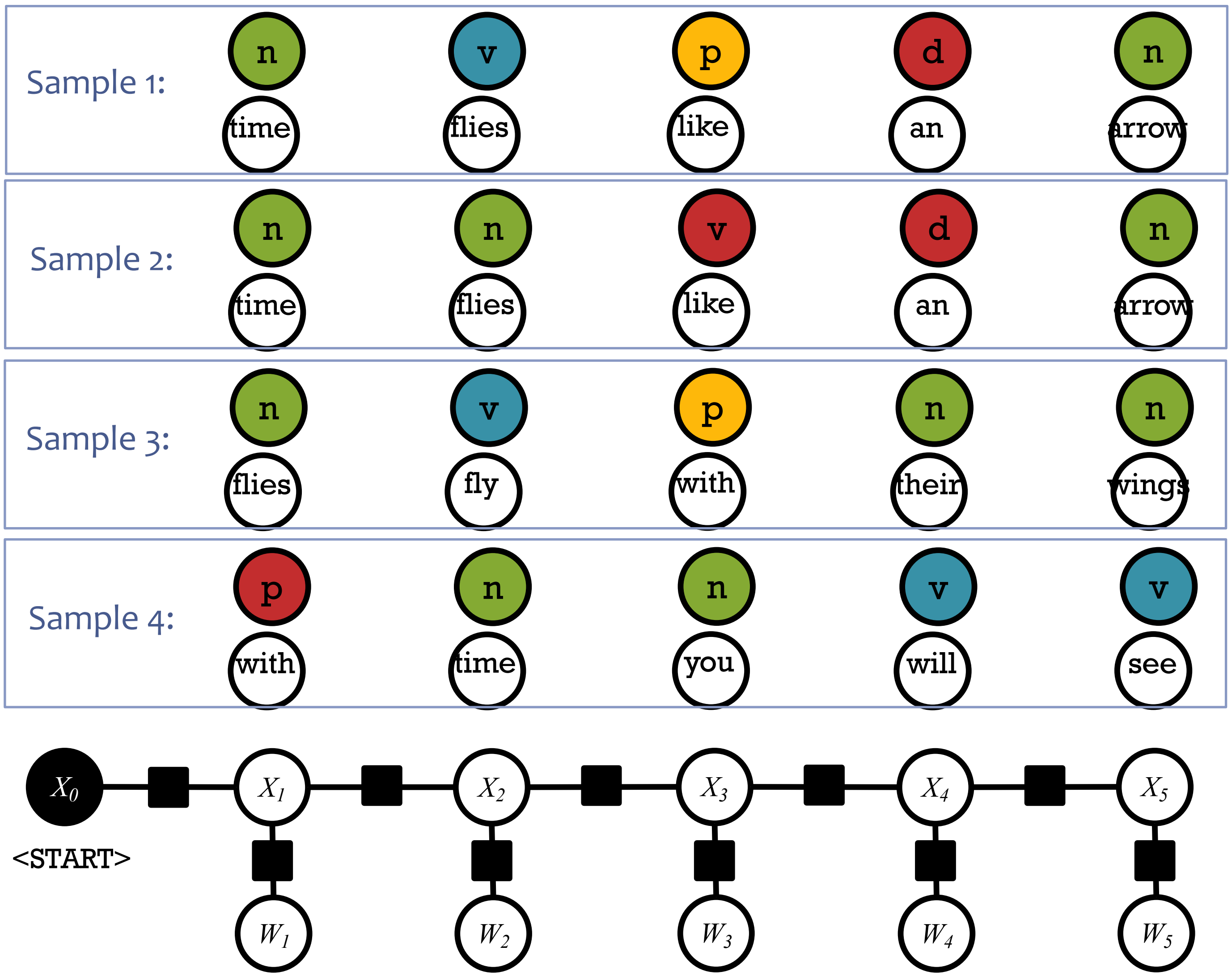
A **joint distribution** defines a probability $p(\boldsymbol{x})$ for each assignment of values $\boldsymbol{x}$ to variables $\boldsymbol{X}$.
This gives the **proportion** of samples that will equal $\boldsymbol{x}$.





Sampling from a Joint Distribution

A **joint distribution** defines a probability $p(\boldsymbol{x})$ for each assignment of values $\boldsymbol{x}$ to variables $\boldsymbol{X}$. This gives the **proportion** of samples that will equal $\boldsymbol{x}$.



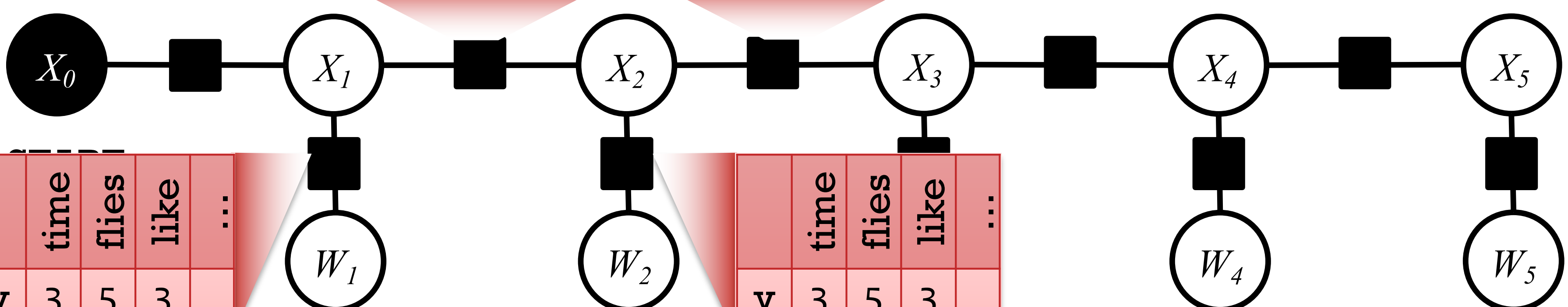
Factors have local opinions (≥ 0)

Each black box looks at *some* of the tags X_i and words W_i

Note: We chose to reuse the same factors at different positions in the sentence.

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0



	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

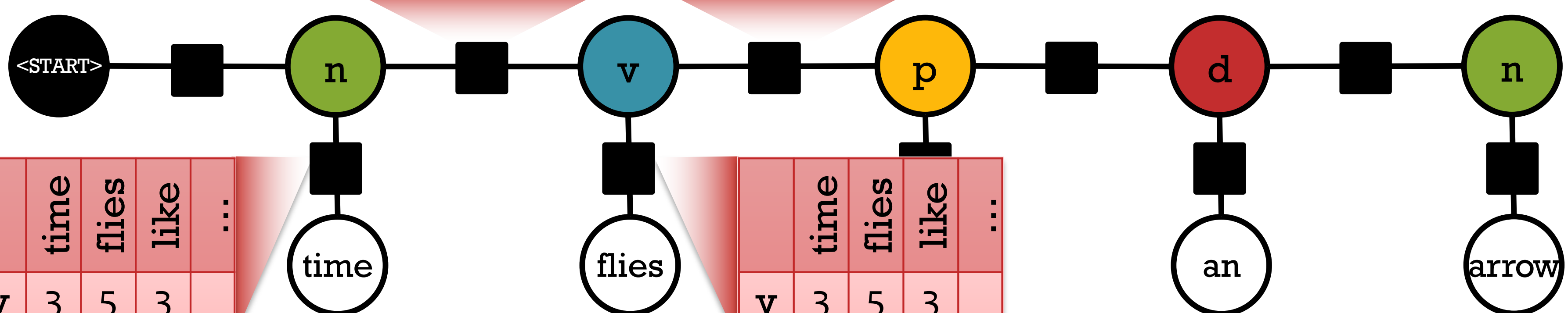
Factors have local opinions (≥ 0)

Each black box looks at *some* of the tags X_i and words W_i

$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) \quad = \quad ?$$

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0



	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

Global probability = product of local opinions

Each black box looks at *some* of the tags X_i and words W_i

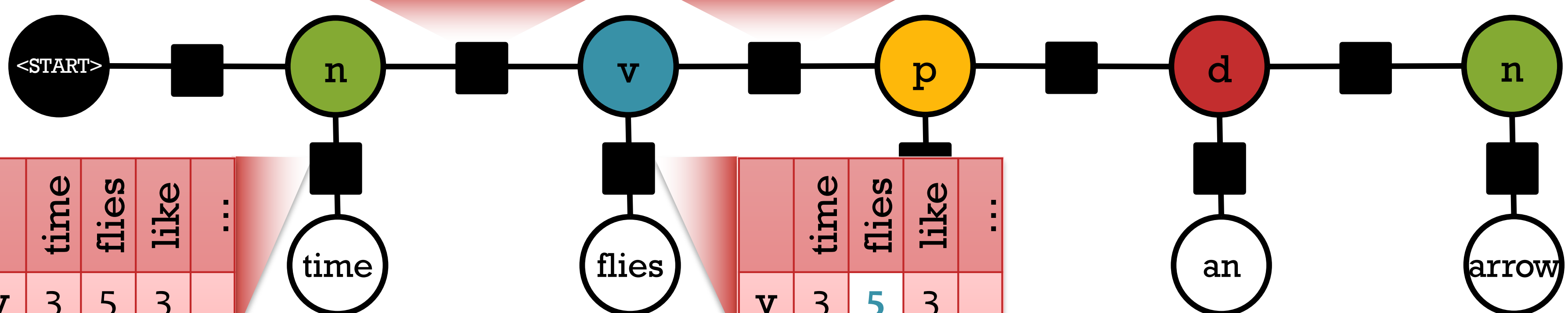
$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) = \frac{1}{Z}(4 * 8 * 5 * 3 * \ldots)$$

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

Uh-oh! The probabilities of the various assignments sum up to Z > 1.

So divide them all by Z.



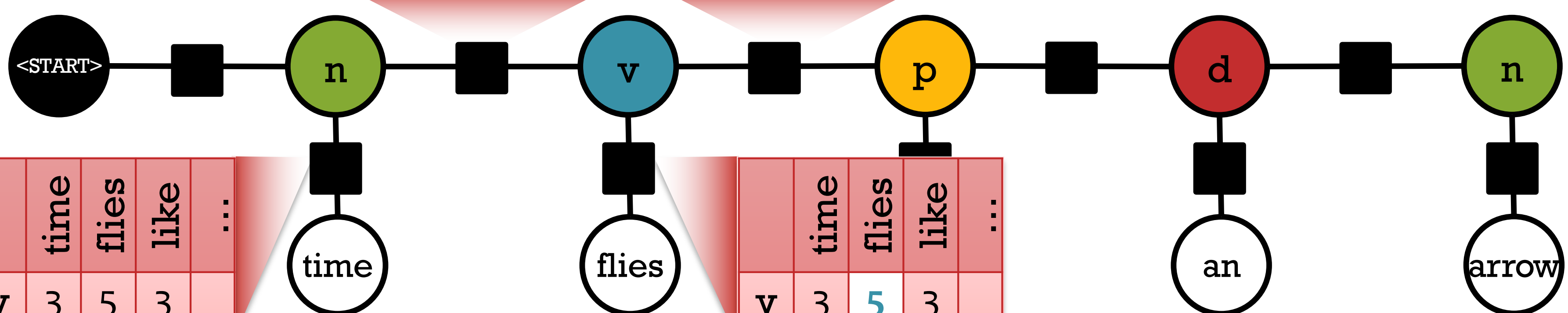
	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

Markov Random Field (MRF)

Joint distribution over tags X_i <u>and</u> words W_i
The individual factors aren't *necessarily* probabilities.

$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) = \frac{1}{Z}(4 * 8 * 5 * 3 * \ldots)$$



	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

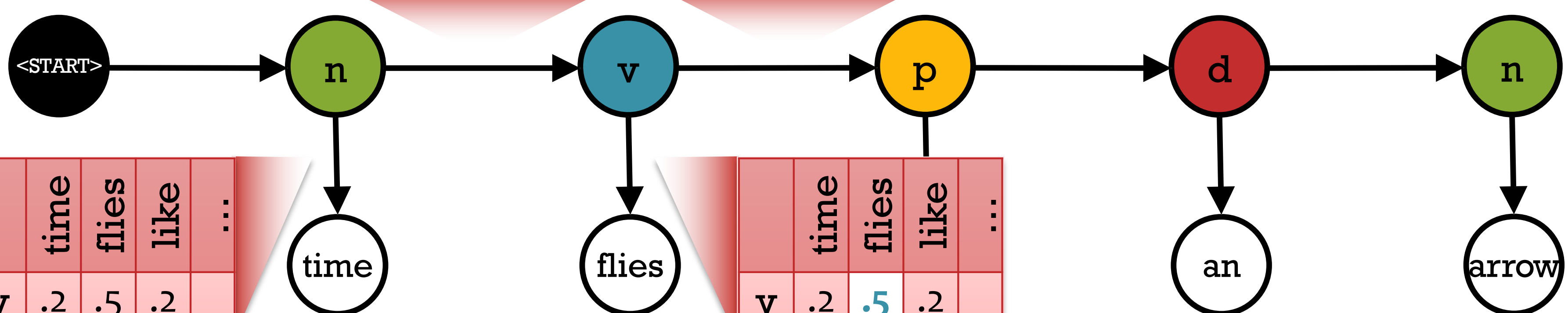
	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

Hidden Markov Model

But sometimes we *choose* to make them probabilities.
Constrain each row of a factor to sum to one. Now $Z = 1$.

$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) = \cancel{\frac{1}{Z}}(.3 * .8 * .2 * .5 * \dots)$$



	v	n	p	d
v	.1	.4	.2	.3
n	.8	.1	.1	0
p	.2	.3	.2	.3
d	.2	.8	0	0

	v	n	p	d
v	.1	.4	.2	.3
n	.8	.1	.1	0
p	.2	.3	.2	.3
d	.2	.8	0	0

	time	flies	like	…
v	.2	.5	.2	
n	.3	.4	.2	
p	.1	.1	.3	
d	.1	.2	.1	

	time	flies	like	…
v	.2	.5	.2	
n	.3	.4	.2	
p	.1	.1	.3	
d	.1	.2	.1	

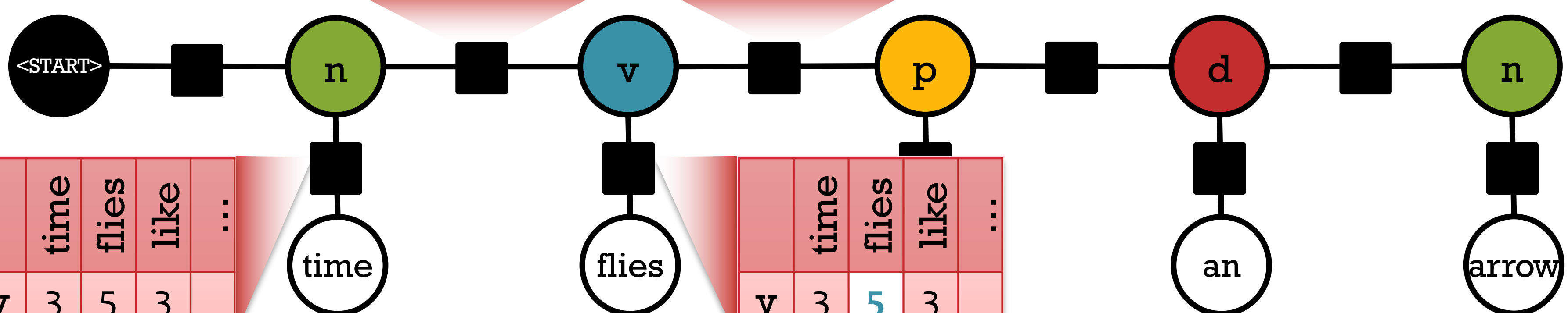
Markov Random Field (MRF)

Joint distribution over tags X_i and words W_i

$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) = \frac{1}{Z}(4 * 8 * 5 * 3 * \ldots)$$

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0



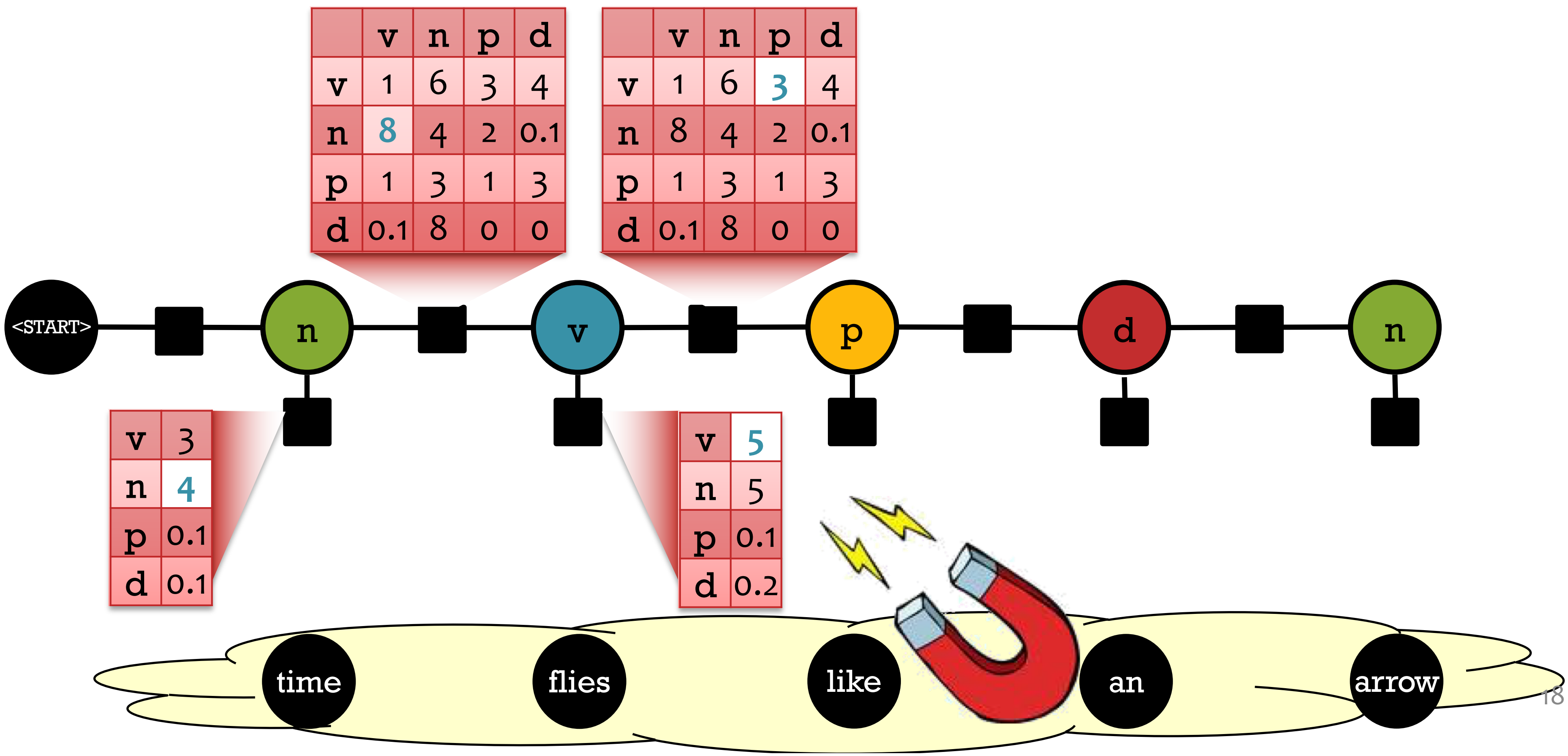
	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

Conditional Random Field (CRF)

Conditional distribution over tags X_i given words w_i.
The factors and Z are now specific to the sentence w.

$$p(\text{n, v, p, d, n} \mid \text{time, flies, like, an, arrow}) \quad = \frac{1}{Z}(4 * 8 * 5 * 3 * \ldots)$$



	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

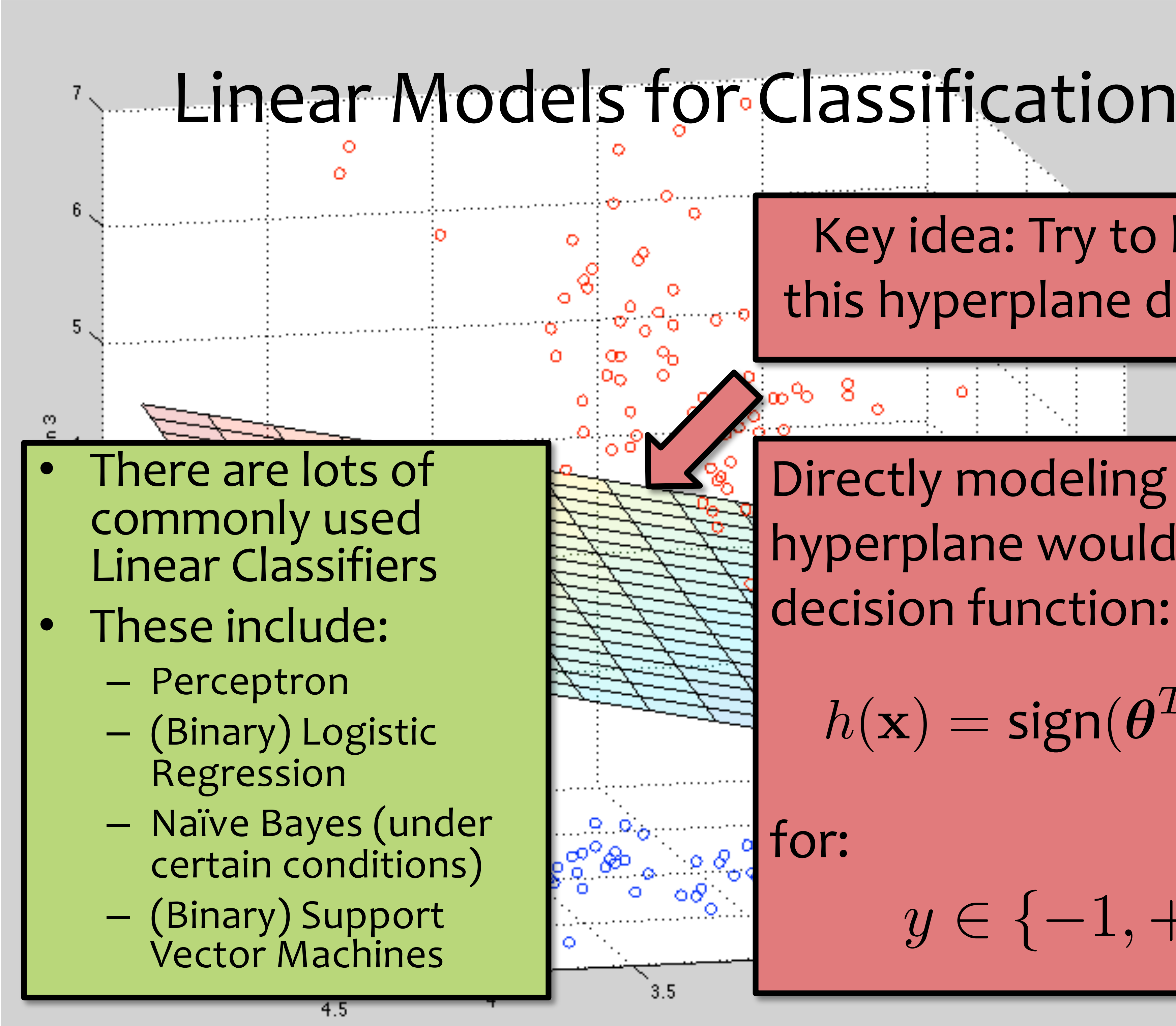
	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

v	3
n	4
p	0.1
d	0.1

v	5
n	5
p	0.1
d	0.2

BACKGROUND: BINARY CLASSIFICATION

Linear Models for Classification



Key idea: Try to learn this hyperplane directly

- There are lots of commonly used Linear Classifiers
- These include:
  - Perceptron
  - (Binary) Logistic Regression
  - Naïve Bayes (under certain conditions)
  - (Binary) Support Vector Machines

Directly modeling the hyperplane would use a decision function:

$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$

for:

$$y \in \{-1, +1\}$$

(Online) Perceptron Algorithm

Data: Inputs are continuous vectors of length *M*. Outputs are discrete.

$$(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots$$

where $\mathbf{x} \in \mathbb{R}^M$ and $y \in \{+1, -1\}$

Prediction: Output determined by hyperplane.

$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$

$$\text{sign}(a) = \begin{cases} 1, & \text{if } a \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

Assume $\boldsymbol{\theta} = [b, w_1, \ldots, w_M]^T$ and $x_0 = 1$

Learning: Iterative procedure:

- **initialize** parameters to vector of all zeroes
- **while** not converged
  - **receive** next example ($\mathbf{x}^{(i)}$, $y^{(i)}$)
  - **predict** y' = h($\mathbf{x}^{(i)}$)
  - **if** positive mistake: **add** $\mathbf{x}^{(i)}$ to parameters
  - **if** negative mistake: **subtract** $\mathbf{x}^{(i)}$ from parameters

(Binary) Logistic Regression

Data: Inputs are continuous vectors of length M. Outputs are discrete.

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N \text{ where } \mathbf{x} \in \mathbb{R}^M \text{ and } y \in \{0, 1\}$$

Model: Logistic function applied to dot product of parameters with input vector.

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

Learning: finds the parameters that minimize some objective function. $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta})$

Prediction: Output is the most probable class.

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}}(y|\mathbf{x})$$

Support Vector Machines (SVMs)

Hard-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \ldots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

Soft-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\left(\sum_{i=1}^{N} e_i\right)$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - e_i, \quad \forall i = 1, \ldots, N$$

$$e_i \geq 0, \quad \forall i = 1, \ldots, N$$

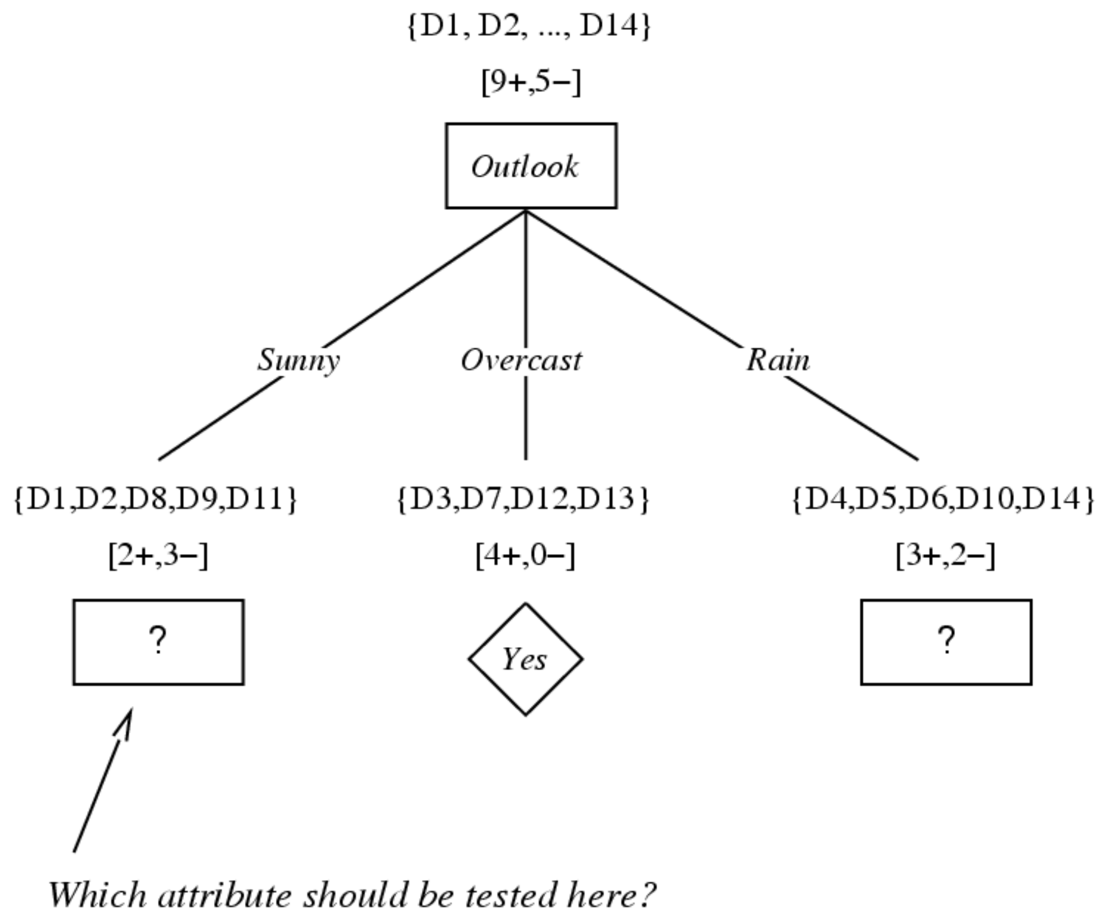
Soft-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } 0 \leq \alpha_i \leq C, \quad \forall i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

Decision Trees



S_{sunny} = {D1,D2,D8,D9,D11}

$Gain(S_{sunny}, Humidity) = .970 - (3/5)\,0.0 - (2/5)\,0.0 = .970$

$Gain(S_{sunny}, Temperature) = .970 - (2/5)\,0.0 - (2/5)\,1.0 - (1/5)\,0.0 = .570$

$Gain(S_{sunny}, Wind) = .970 - (2/5)\,1.0 - (3/5)\,.918 = .019$

Figure from Tom Mitchell

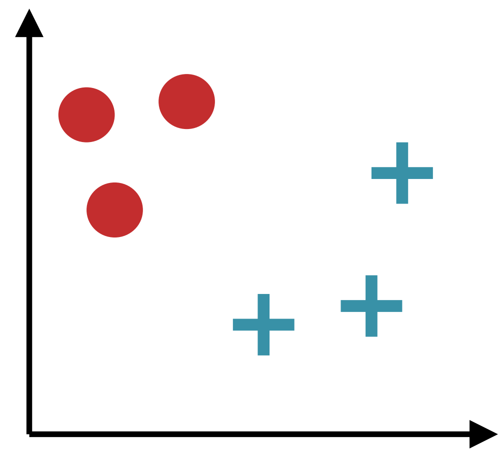
Binary and Multiclass Classification

Supervised Learning:

$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}$ $\quad\quad$ $\mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$

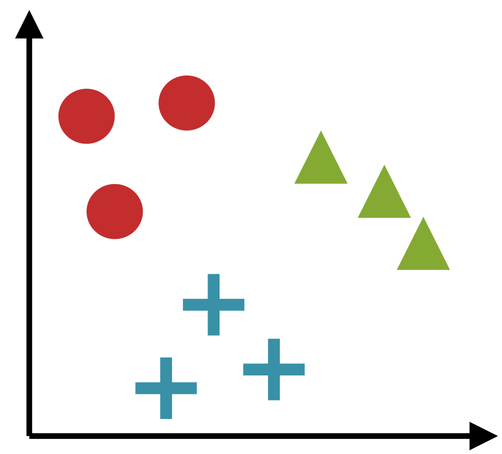
Binary Classification:

$$y^{(i)} \in \{+1, -1\}$$



Multiclass Classification:

$$y^{(i)} \in \{1, \ldots, K\}$$



Outline

Reductions
(Binary → Multiclass)

1. one-vs-all (OVA)
2. all-vs-all (AVA)
3. classification tree
4. error correcting output codes (ECOC)

Settings

A. Multiclass Classification
B. Hierarchical Classification
C. Extreme Classification

Why?

– multiclass is the simplest structured prediction setting
– key insights in the simple reductions are analogous to later (less simple) concepts

REDUCTIONS OF MULTICLASS TO BINARY CLASSIFICATION

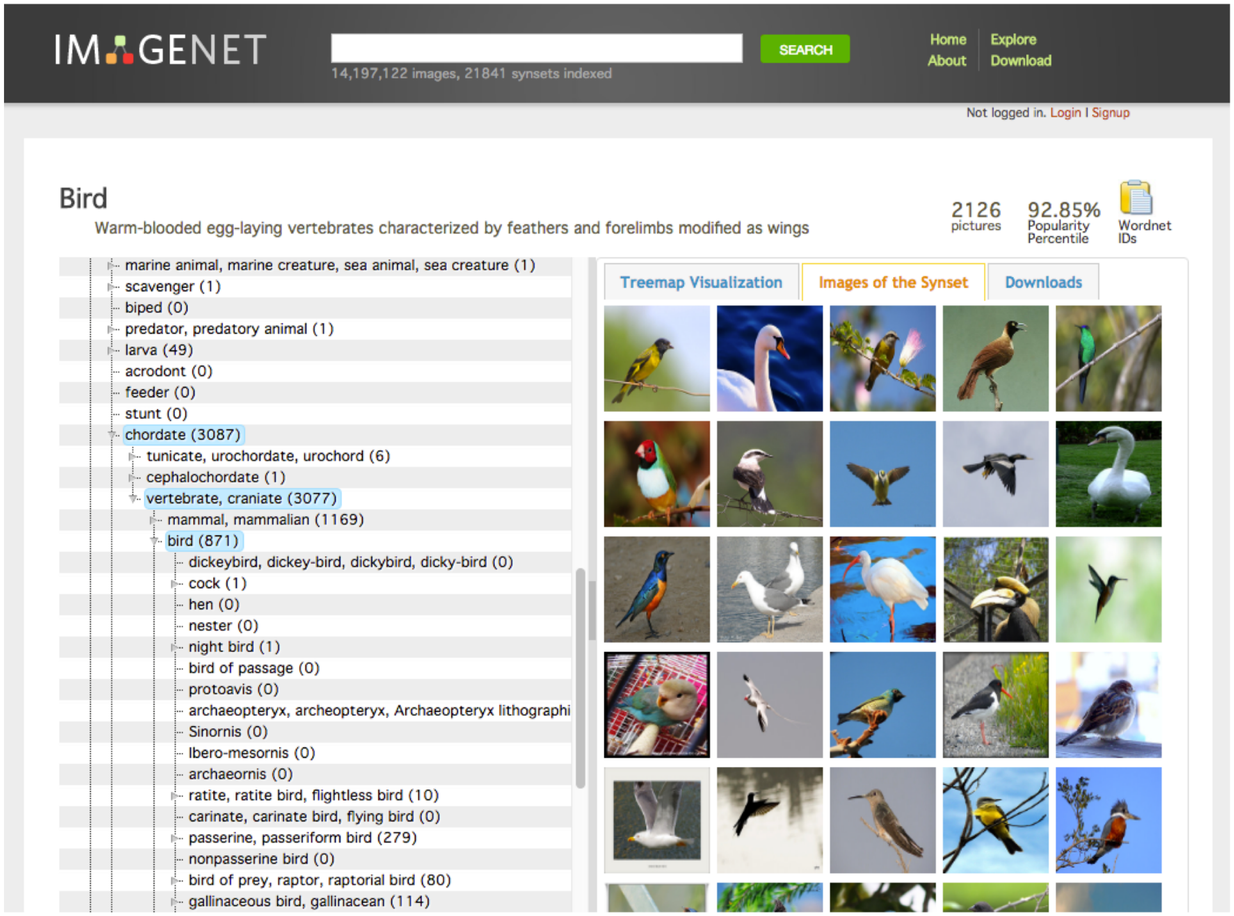
Reductions to Binary Classification

***Whiteboard*:**

- Setting for multiclass to binary reductions
- Reduction 1: One-vs-All (OVA)
- Reduction 2: All-vs-All (AVA)
- Reduction 3: Classification Tree

HIERARCHICAL CLASSIFICATION

Hierarchical Classification



Setting:

- **Given hierarchy** over output labels
- Otherwise, the **same as multiclass** classification
- Each **leaf node is a label**

Hierarchical Classification

9					
10		**2010 Standard Occupational Classification**			
11					
12	Major Group	Minor Group	Broad Group	Detailed Occupation	
13					

Row	Major Group	Minor Group	Broad Group	Detailed Occupation	Title
960				45-4022	Logging Equipment Operators
961				45-4023	Log Graders and Scalers
962				45-4029	Logging Workers, All Other
963	**47-0000**				**Construction and Extraction Occupations**
964		**47-1000**			**Supervisors of Construction and Extraction Workers**
965			47-1010		First-Line Supervisors of Construction Trades and Extraction Workers
966				47-1011	First-Line Supervisors of Construction Trades and Extraction Workers
967		**47-2000**			**Construction Trades Workers**
968			47-2010		Boilermakers
969				47-2011	Boilermakers
970			47-2020		Brickmasons, Blockmasons, and Stonemasons
971				47-2021	Brickmasons and Blockmasons
972				47-2022	Stonemasons
973			47-2030		Carpenters
974				47-2031	Carpenters
975			47-2040		Carpet, Floor, and Tile Installers and Finishers
976				47-2041	Carpet Installers
977				47-2042	Floor Layers, Except Carpet, Wood, and Hard Tiles
978				47-2043	Floor Sanders and Finishers
979				47-2044	Tile and Marble Setters
980			47-2050		Cement Masons, Concrete Finishers, and Terrazzo Workers
981					
982					
983		**47-3000**			
984			47		
985					
986					
987					
988					
989					
990					
991					
992		**47-4000**			
993			47		
994					
995			47		
996					
997			47		
998					
999			47		
1000					
1001			47		
1002					
1003			47		
1004					

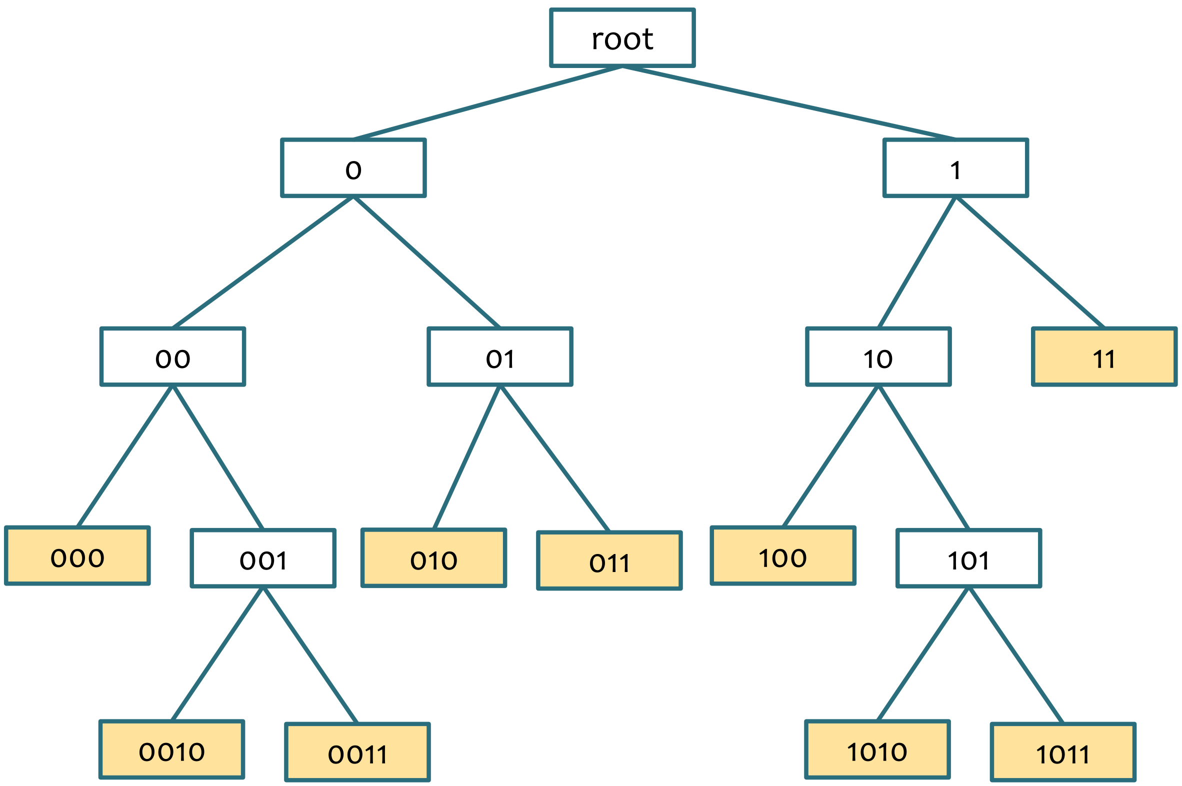
Training Data: pairs of occupation descriptions and their SOC code

- 9560,Rigging up man
- 5900,Mimeographer
- 3040,Doctor of optometry
- 8310,Wool presser
- 8720,Compress machine operator
- 9640,Pretzel packer
- 9260,Hot box spotter

Setting:

- **Given hierarchy** over output labels
- Otherwise, the **same as multiclass** classification
- Each **leaf node is a label**

Hierarchical Classification



Setting:

- **Given hierarchy** over output labels
- Otherwise, the **same as multiclass** classification
- Each **leaf node is a label**

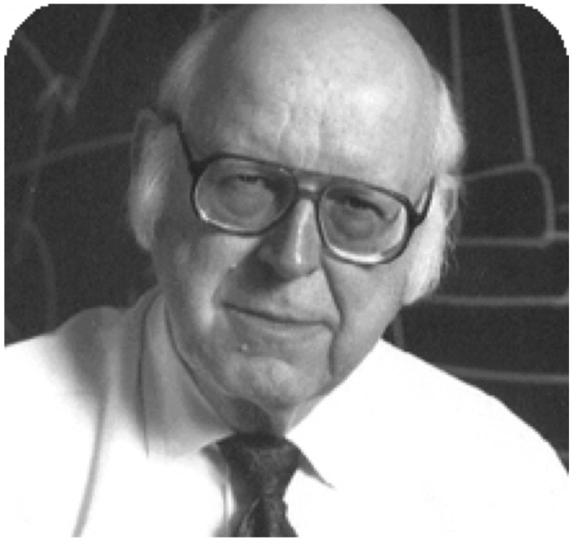
Reductions to Binary Classification

***Whiteboard*:**

- Hierarchical classification: how to build an appropriate classifier?
- Features of input vector and label
- Reduction 4: Error Correcting Output Codes (ECOC)

EXTREME CLASSIFICATION

Extreme Classification






Example adapted from Paul Miniero's ICML 2017 talk

Extreme Classification

Setting:

- Output label set is **extremely large** (e.g. millions of labels)
- Otherwise, the **same as multiclass** classification

Example Tasks:

- Large-scale facial recognition (billions?)
- Predicting Amazon product categories (3 million)
- Recommending Amazon items (100 million products)
- Predicting Wikipedia tags (2 million)
- Predicting Flick image tags
- Language modeling (millions of words)

Figure from Daumé III et al., (2017)

Logarithmic-time One-Against-Some

Key idea behind this algorithm:

- build a **Recall Tree** where
  - each leaf node contains a set S of labels where $|S| \leq \log_2(K)$
  - depth of tree is $d \leq \log_2(K)$
- learn **one binary classifier per internal node** to route an instance (vector **x**) to a leaf node
- learn **one multiclass classifier per leaf** over the set of labels S which restricts the label set for instances **x** routed there
- given a new instance, **predict one of the |S| labels** at the leaf to which the instance was routed

An example Recall Tree:

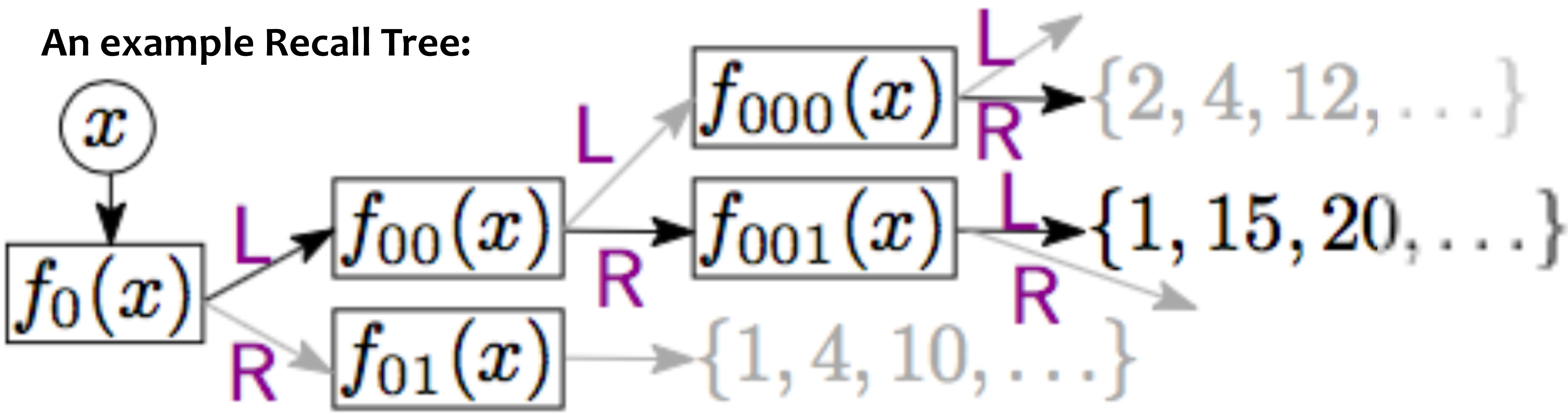


Figure from Daumé III et al., (2017)

Logarithmic-time One-Against-Some

Properties:

1. Competes with one-against-all (i.e. standard multiclass classifier) on **benchmark** datasets
2. **Speed**: O(log K) training and prediction
3. **Space**: O(K), same as one-against-all
4. **Online** learning!

An example Recall Tree:

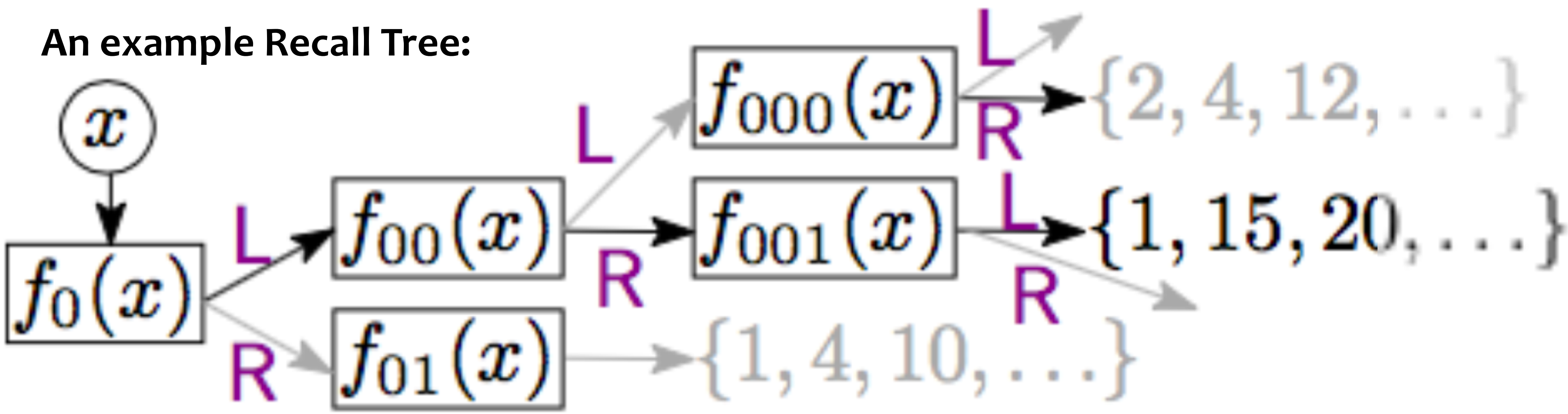
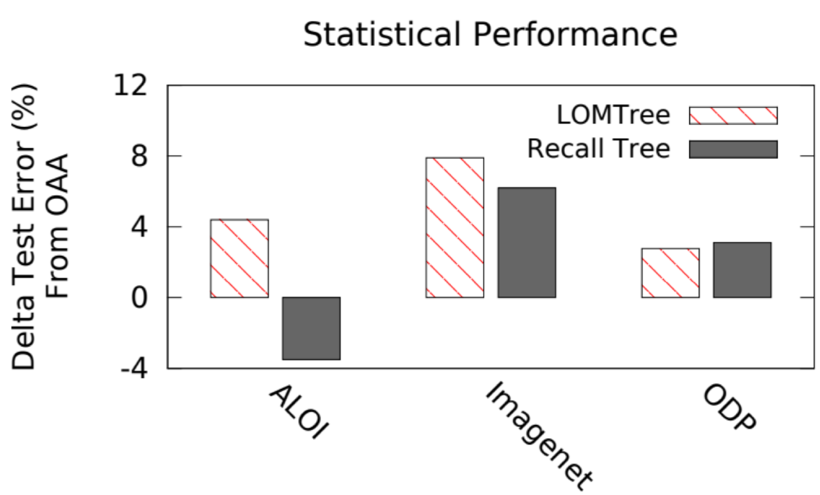


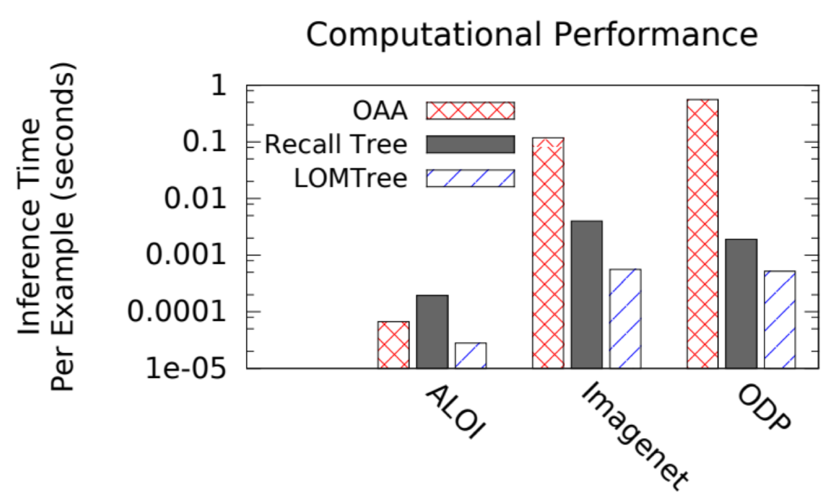
Figure from Daumé III et al., (2017)

Logarithmic-time One-Against-Some

Experiments:

Dataset	Task	Classes	Examples
ALOI[10]	Visual Object Recognition	$1k$	10^5
Imagenet[19]	Visual Object Recognition	$\approx 20k$	$\approx 10^7$
LTCB[14]	Language Modeling	$\approx 80k$	$\approx 10^8$
ODP[2]	Document Classification	$\approx 100k$	$\approx 10^6$





Learning Objectives

From Binary to Multiclass Classification

You should be able to…

1. Reduce the multiclass classification problem to a collection of binary classification problems
2. Identify the advantages and deficiencies of different multiclass-to-binary reductions
3. Implement one-vs-all, all-vs-all, classification tree, error correcting output codes
4. Differentiate multiclass, hierarchical, and extreme classification settings