Language Technologies Institute

Carnegie Mellon University

Multimodal Affective Computing

Lecture 9: Probabilistic Predictive Modeling

Louis-Philippe Morency
Jeffrey Girard

Originally developed with help from Stefan Scherer and Tadas Baltrušaitis

Outline

- Basic concepts of machine learning
  - Definitions and types of algorithms
  - Linear regression and classification
- Joint probability distribution
- Probabilistic graphical models
  - Independence and Conditional independence
  - Example: modeling affect during learning
- Bayesian networks
  - Conditional probability distribution
  - Dynamic Bayesian Network
  - Naïve Bayes classifier
- Evaluation methods and error measures

Upcoming Deadlines and Course Schedule

- **Thursday, April 4th 4:30pm-6pm**
  - Midterm presentations
- **Sunday, April 7th at 11:59pm**
  - Midterm report deadline
- **Tuesday April 30th – NO CLASS**
  - Preparation for final report and presentation
- **Thursday, May 2nd 4:30pm-6pm**
  - Final presentations
- **Tuesday, May 7th at 11:30pm**
  - Final report deadline

Basic Concepts of Machine Learning

What is Machine Learning?

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

- Tom Mitchell

Machine learning algorithms originate from 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!

Types of Learning Algorithms

- Supervised learning: classification is seen as supervised learning from examples.
  - Supervision: The data (observations, measurements, etc.) are labeled with pre-defined classes. It is like that a "teacher" gives the classes (supervision).
- Unsupervised learning (clustering)
  - Class labels of the data are unknown
  - Given a set of data, the task is to establish the existence of classes or clusters/groupings in the data
- Reinforcement learning

Variables Involved in Supervised Learning

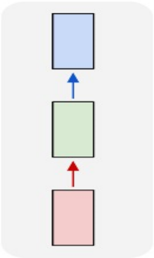
- **Input:** evidences, independent variables, observations
- **Output:** labels, outcome variables, dependent variable, non-observed
- **Hidden:** latent variables, intermediate representations

Types of Supervised Learning Algorithms

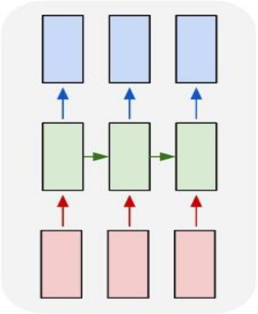
- Classification: categorize an example
  - Binary - {depressed, non-depressed}
  - Multi-class – {happy, sad, angry, neutral}
  - Recall nominal data from last week
- Regression:
  - Predict the intensity – how depressed a person is
  - Recall interval and ordinal data from last week

Supervised Learning with Sequential Data

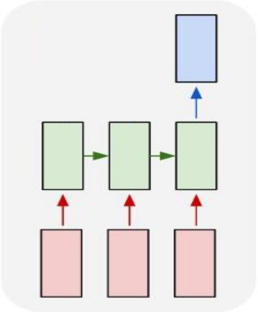
- Prediction with summary features
  - Input $\boldsymbol{x_i}$ represents the whole sequence
  - Output y_i summarizes the sequence



- Sequential Labeling
  - Input $\boldsymbol{X_i} = \{\boldsymbol{x}_1, \boldsymbol{x}_2 \dots, \boldsymbol{x}_k\}$ represent a sequence i of length *k*
  - Output $\boldsymbol{Y_i} = \{y_1, y_2 \dots, y_k\}$ represents the labels for the sequence



- Sequence Prediction
  - Input $\boldsymbol{X_i} = \{\boldsymbol{x}_1, \boldsymbol{x}_2 \dots, \boldsymbol{x}_k\}$ represent a sequence i of length *k*
  - Output y_i summarizes the sequence



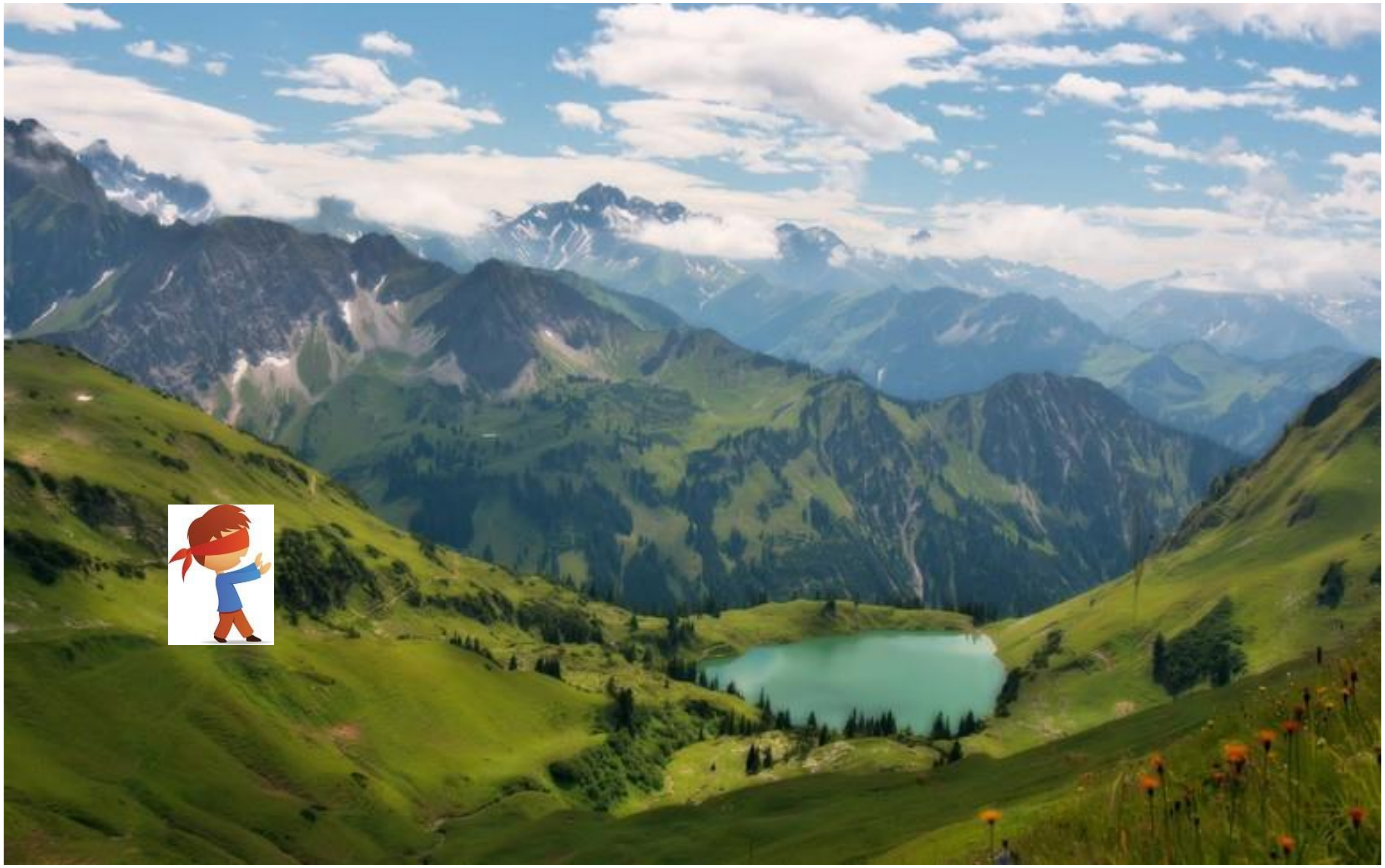
The Supervised Learn Problem

- Given
  - $D = \{(\boldsymbol{x}_1, y_1) \ldots (\boldsymbol{x}_n, y_n)\}$
  - a training set of samples of input variables $\boldsymbol{x}$ and output variables y
- Learn
  - $\hat{y} = f(\boldsymbol{x}; \boldsymbol{W})$
  - a function parametrized by $\boldsymbol{W}$ that predicts the output class ($y = c$) from the input variables

Main Ingredients of most Supervised Learning Algorithms

1. **Score function:** $\hat{y} = f(\boldsymbol{x}; \boldsymbol{W})$
   - Perform inference using current parameters $\boldsymbol{W}$
2. **Loss function:** $\boldsymbol{L}(\boldsymbol{W}; \boldsymbol{D})$
   - Goal: How to assign only one number representing how "unhappy" we are about these scores?
3. **Optimization**
   - Adjust the model parameters $\boldsymbol{W}$ to best minimize the loss function on the training data $\boldsymbol{D}$

Optimization



Linear Regression Model (from previous lectures)

- Score?
  - $f(\boldsymbol{x_i}; \boldsymbol{w}) = w_0 + \boldsymbol{w_1}\boldsymbol{x_i} + \varepsilon$
- Loss?
  - $Absolute\ Error$: $\sum_i |y_i - f(\boldsymbol{x_i}; \boldsymbol{w})|$
  - $Squared\ Error$: $\sum_{\boldsymbol{i}} \left(y_i - f(\boldsymbol{x_i}; \boldsymbol{w})\right)^{\boldsymbol{2}}$
- Optimization?
  - Least square (close form solution)

What if the relationship between x and y is non-linear?

Generalized Linear Regression Model

Still linear with respect to w

- Score?
  - $f(\boldsymbol{x_i}; \boldsymbol{w}) = \sum_j \boldsymbol{w_j} \varphi_j(\boldsymbol{x_i}) + \varepsilon$
  - where $\varphi_j(\boldsymbol{x_i})$ is a non-linear function of $\boldsymbol{x_i}$
- Loss?
  - $Absolute\ Error\ \sum_i |y_i - f(\boldsymbol{x_i}; \boldsymbol{w})|$
  - $Squared\ Error\ \sum_{\boldsymbol{i}} \left(y_i - f(\boldsymbol{x_i}; \boldsymbol{w})\right)^{\boldsymbol{2}}$
- Optimization?
  - Gradient descent
  - Ordinary least square

Basis functions

Polynomial:

$$\varphi_j(x) = x^k$$

Gaussian:

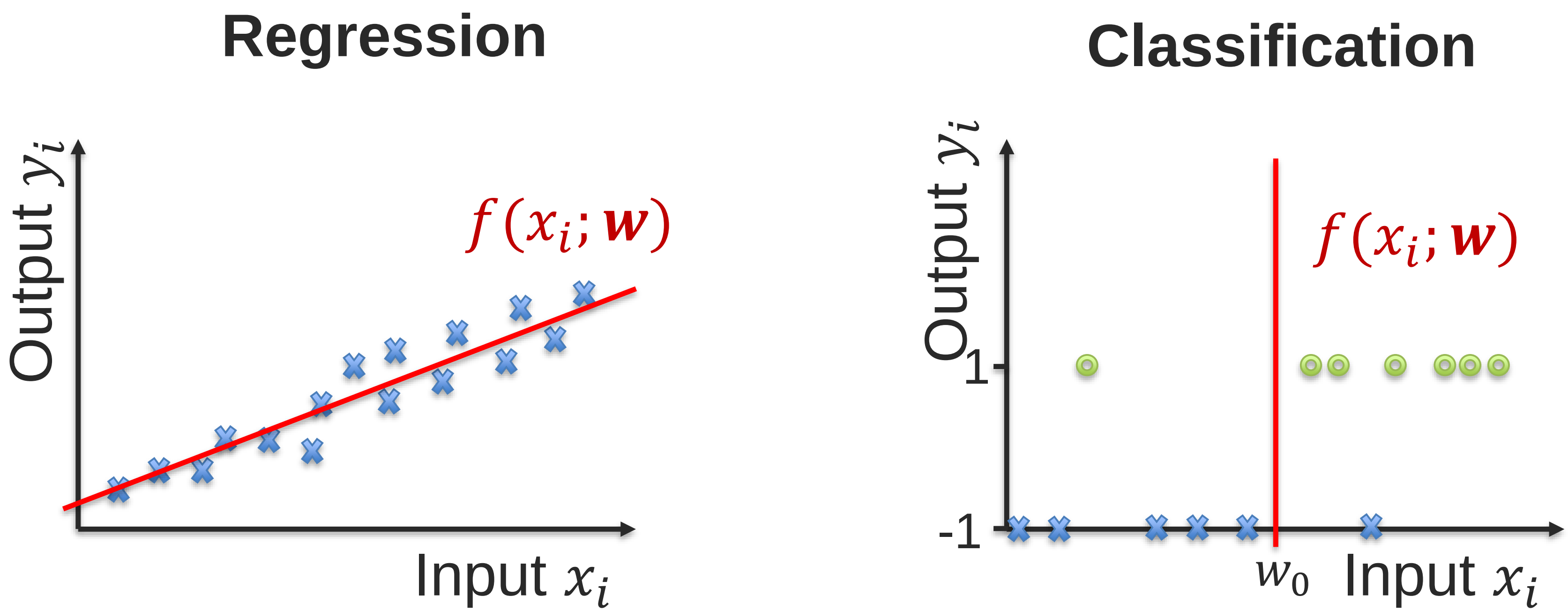
$$\varphi_j(x) = \frac{(x - \mu_j)}{2\sigma_j^2}$$

Logs:

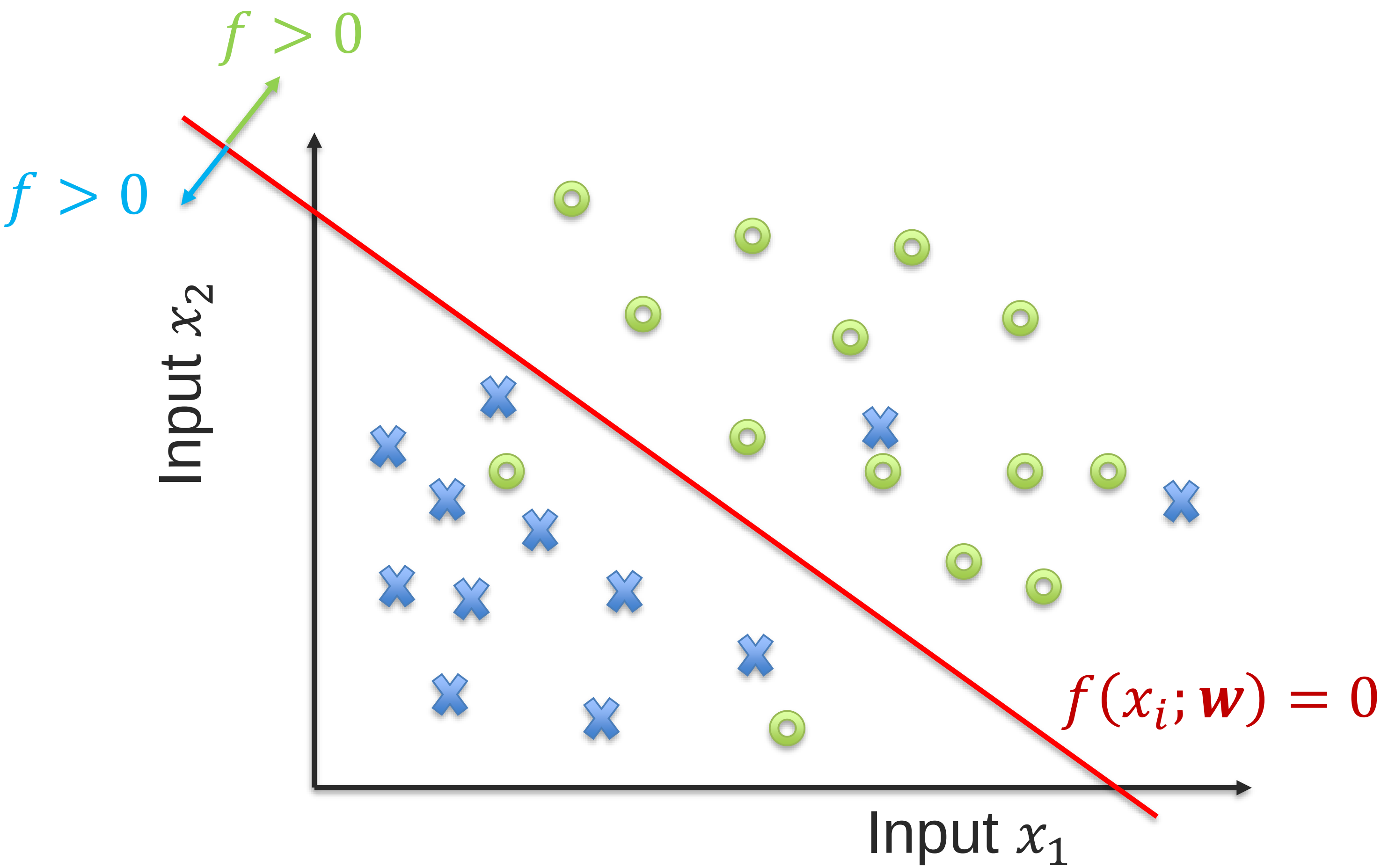
$$\varphi_j(x) = \log(x + 1)$$

But what if the output y_i is binary or discrete?

Classification vs Regression – 1D Visualization



Classification – 2D Visualization

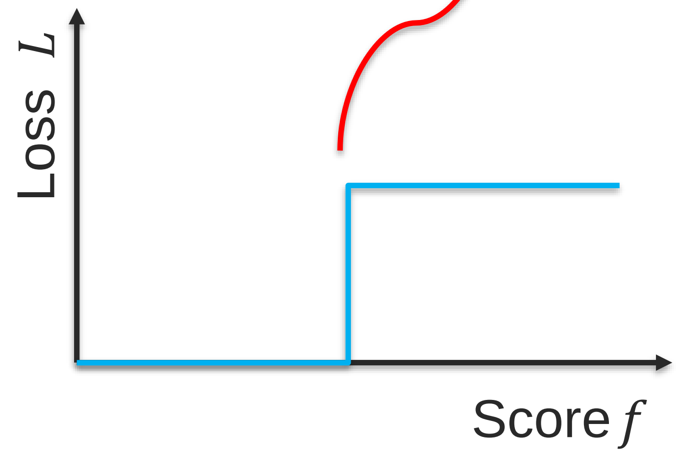


Linear Classifier

- Score?
  - $f(\boldsymbol{x_i}; \boldsymbol{w}) = w_0 + \boldsymbol{w_1 x_i}$
- Loss?
  - $\cancel{L(\boldsymbol{w}) = \sum_i \left(y_i - \text{sign}(f(\boldsymbol{x_i}; \boldsymbol{w}))\right)^2}$

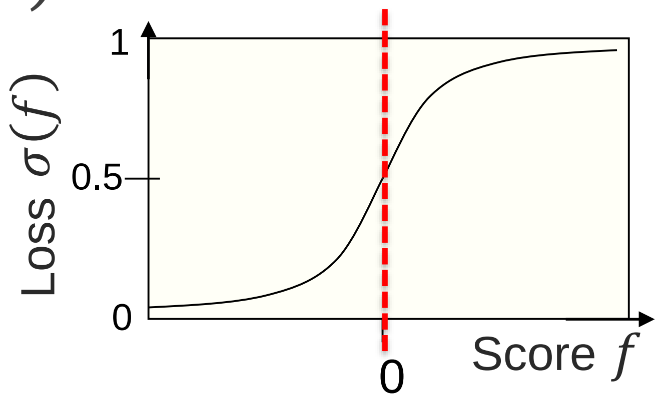
Nonlinearity!

- Optimization?



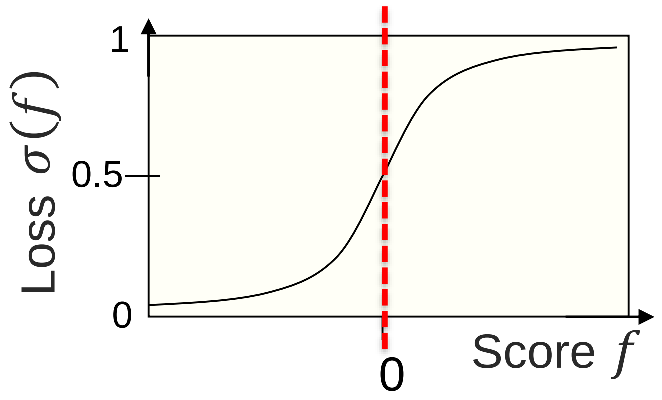
Logistic Regression – A Linear Classifier

- Score?
  - $f(\boldsymbol{x_i}; \boldsymbol{w}) = w_0 + \boldsymbol{w_1 x_i}$
- Loss?
  - *Logistic function* $\sigma(f) = \dfrac{1}{1 + e^{-f}}$
  - $L(w) = \sum_i \left(y_i - \sigma\left(f(\boldsymbol{x_i}; \boldsymbol{w})\right)\right)^2$
- Optimization?
  - Gradient descent
  - ➢ Very similar formulation for perceptron and linear SVM



Logistic Regression – Probabilistic Interpretation

- Score?
  - $f(\boldsymbol{x_i}; \boldsymbol{w}) = w_0 + \boldsymbol{w_1 x_i}$
- Loss?
  - $p(y_i = 1|\boldsymbol{x_i}; \boldsymbol{w}) = \sigma(f(\boldsymbol{x_i}; \boldsymbol{w}))$
  - Negative log likelihood
- Optimization?
  - Gradient descent



Joint Probability Distribution

Random Variables

Definition: A variable whose possible values are numerical outcomes of a random phenomenon.

- **Discrete** random variable is one which may take on only a countable number of distinct values such as 0,1,2,3,4,…
- **Continuous** random variable is one which takes an infinite number of possible values.

Examples of random variables:

- Someone's age
- Someone's height
- Someone's weight

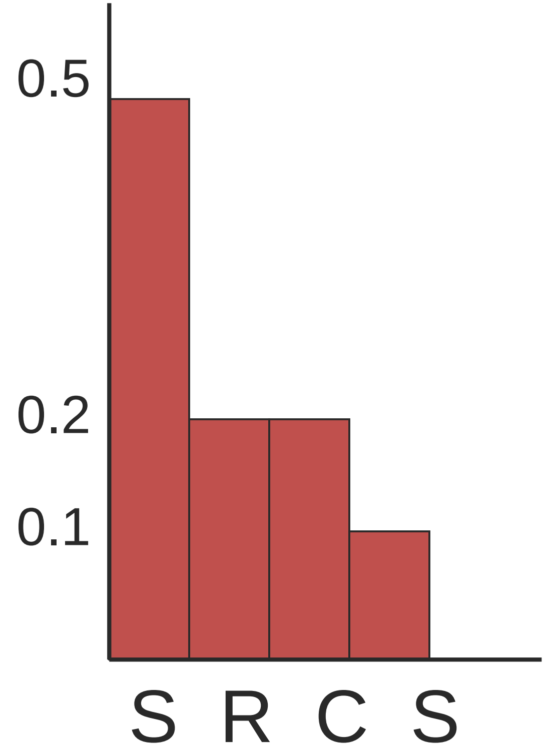
Discrete or continuous?

Correlated?

Probability Distributions

- Probability Distribution:
  - p(Weather=Sunny) = 0.5
  - p(Weather=Rain)= 0.2
  - p(Weather=Cloud)= 0.2
  - p(Weather=Snow)= 0.1

- Distribution sums to 1.



Joint Probability Distribution

- Completely specifies all beliefs in a problem domain.
- Joint probability distribution is an n-dimensional table with a probability in each cell of that state occurring.
- Written as $P(X_1, X_2, X_3 \ldots, X_n)$
- When instantiated as $P(x_1, x_2 \ldots, x_n)$

Example - Joint Probability Distribution

- Domain with 2 variables each of which can take on 2 states.

P(Toothache, Cavity)

	Toothache	¬Toothache
Cavity	0.04	0.06
¬Cavity	0.01	0.89

Background: Rules of Probability

Sum rule:
(integrating, marginalizing)

$$P(X) = \sum_Y P(X,Y)$$

Product rule:
(chain rule)

$$P(X,Y) = P(X|Y)P(Y)$$

$$P\left(\bigcap_{k=1}^{N} X_k\right) = \prod_{k=1}^{N} P\left(X_k \middle| \bigcap_{j=1}^{k-1} X_k\right)$$

Inference for Known Joint Probability Distribution

When we know the joint probability distribution :

$$P(A, B, C, D, E) \Rightarrow \left\{ \text{If A, B C, D and E are discrete variables, then P(A,B,C,D, E) will be a 5-D tensor (matrix)} \right.$$

Two main forms of inference:

1 Joint probability for a particular assignment

$$P(A = 1, B = 'car', C = 2, D = 'banana', E = 10)$$

$\Rightarrow$ A specific entry in the 5-D tensor

Inference for Known Joint Probability Distribution

2 Probability of a subset of variables (query) given known assignments of other variables (evidences)

$P(A, D|C = 3)$ → Use the product rule to *marginalize* the other variables B and E

$$P(A, D|C = 3) = \sum_{\forall b \in B, e \in E} P(A, D, b, e|C = 3)$$

→ Use the inverse of product rule $P(X|Y) = P(X, Y)/P(Y)$

$$P(A, D|C = 3) = \frac{1}{P(C)} \sum_{\forall b \in B, e \in E} P(A, D, b, e|C = 3)$$

Inference for Known Joint Probability Distribution

2. Probability of a subset of variables (query) given known assignments of other variables (evidences)

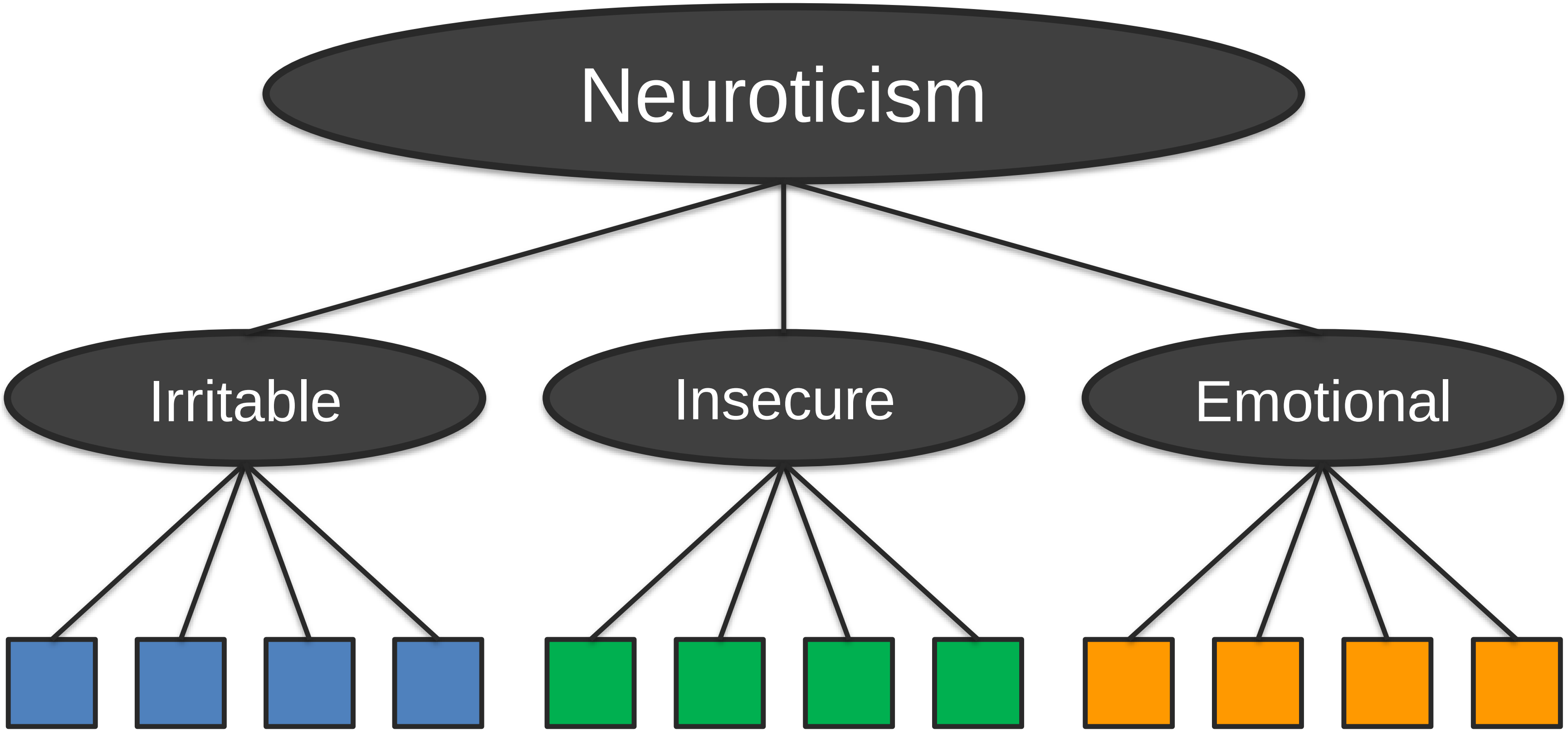
$$P(x|y) = \alpha \sum_{\forall z \in Z} P(x, y, z)$$

where x is the subset of query variables

y is the subset of evidence assignments

Z is the set of all other variables (not in x or y)

Models with Multiple Outcome and Latent Variables



How can we model the joint probability distribution of this model?

Probabilistic Graphical Models

Probabilistic Graphical Model

Definition: A probabilistic graphical model (PGM) is a graph formalism for compactly modeling joint probability distributions and dependence structures over a set of random variables.

- Random variables: $X_1,\ldots,X_n$
- P is a joint distribution over $X_1,\ldots,X_n$

Can we represent P more compactly?

- Key: Exploit independence properties

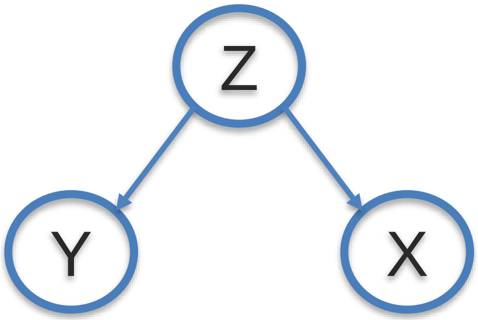
Independent Random Variables

- Two variables X and Y are independent if
  - $P(X=x|Y=y) = P(X=x)$ for all values x,y
  - Equivalently, knowing Y does not change predictions of X
- If X and Y are independent then:
  - $P(X, Y) = P(X|Y)P(Y) = P(X)P(Y)$

- If $X_1,\ldots,X_n$ are independent then:
  - $P(X_1,\ldots,X_n) = P(X_1)\ldots P(X_n)$

Conditional Independence

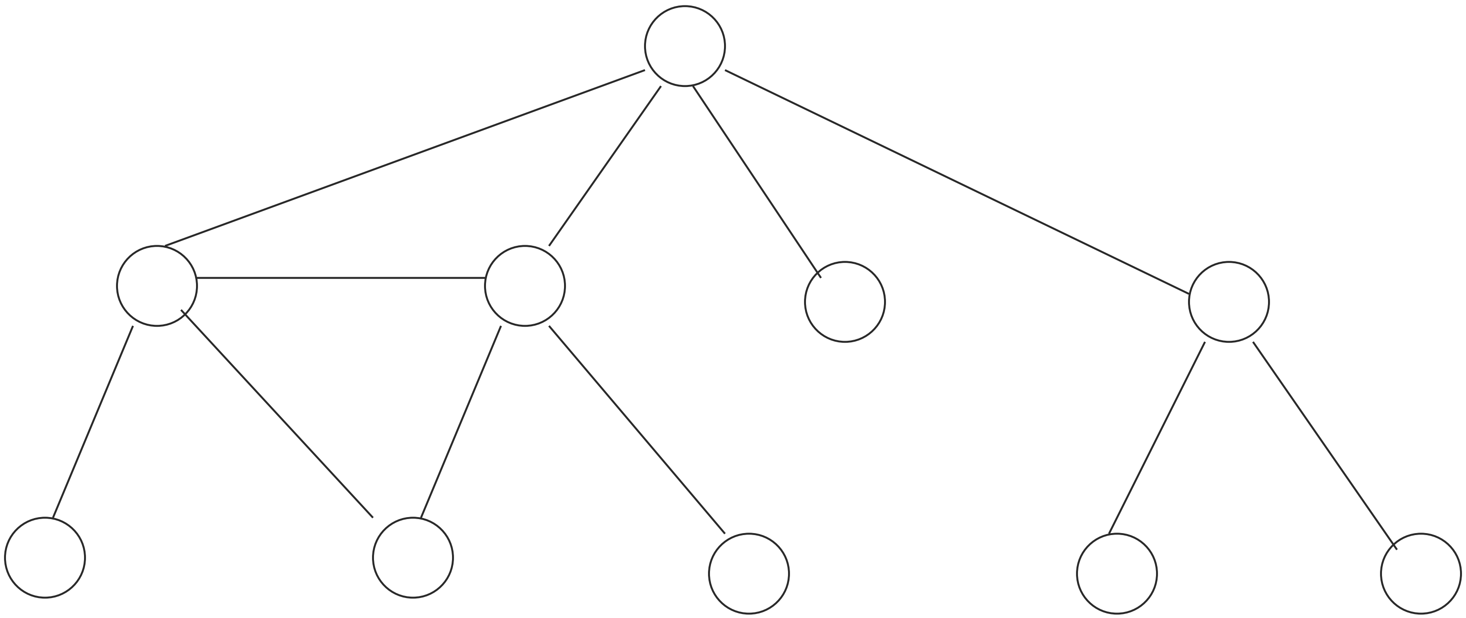
- X and Y are conditionally independent given Z if
  - $P(X=x|Y=y, Z=z) = P(X=x|Z=z)$ for all values x, y, z
  - Equivalently, if we know Z, then knowing Y does not change predictions of X



Graphical Model

- A tool that visually illustrate conditional independence among variables in a given problem.
- Consisting of nodes (Random variables or States) and edges (Connecting two nodes, directed or undirected).
- The lack of edge represents conditional independence between variables.

Graphical Model



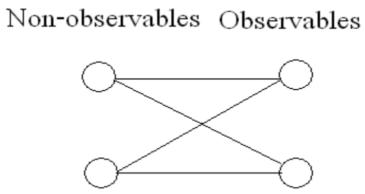
Different types of graphical models:

- Chain, Path, Cycle, Directed Acyclic Graph (DAG), Parents and Children

Uncertain Reasoning – Latent Variables

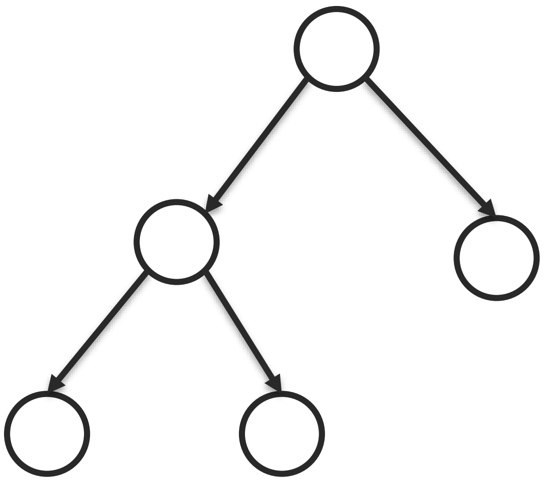
- Some aspects of the domain are often unobservable and must be estimated indirectly through other observations.

- The relationships among domain events are often uncertain, particularly the relationship between the observables and non-observables.



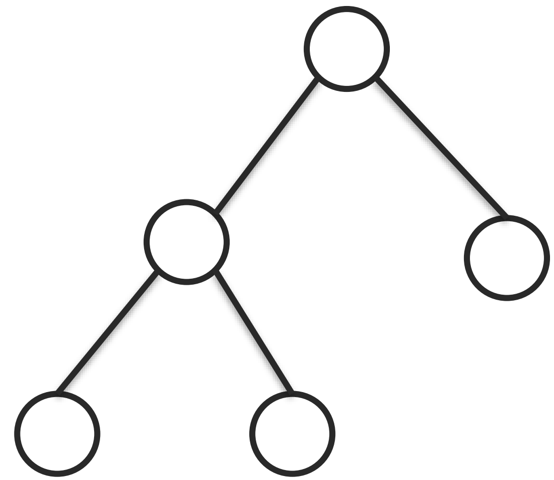
Two Main Types of Graphical Models

Bayesian networks



- Directed acyclic graph
- Conditional dependencies

Markov Models (next week)

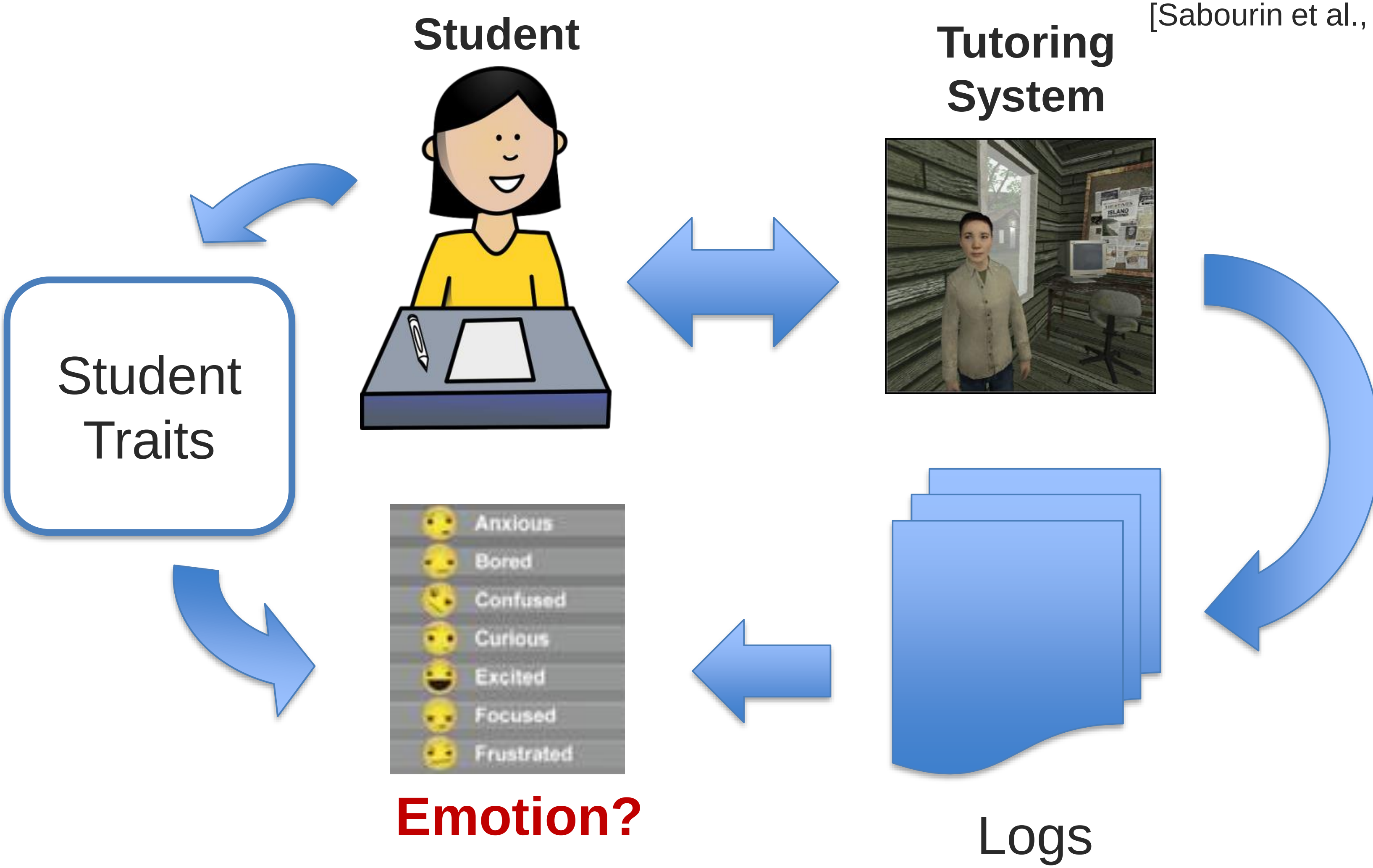


- Undirected graphical model
- Cyclic dependencies

Creating a Graphical Model

Example: Inferring Emotion from Interaction Logs

[Sabourin et al., 2011]



Example: Bayesian Network Representation

[Sabourin et al., 2011]

Outcome (non-observable)

Emotion

Evidences (observable)

book views
correct ans.
notes taken
incorrect ans.
poster views
Total goals

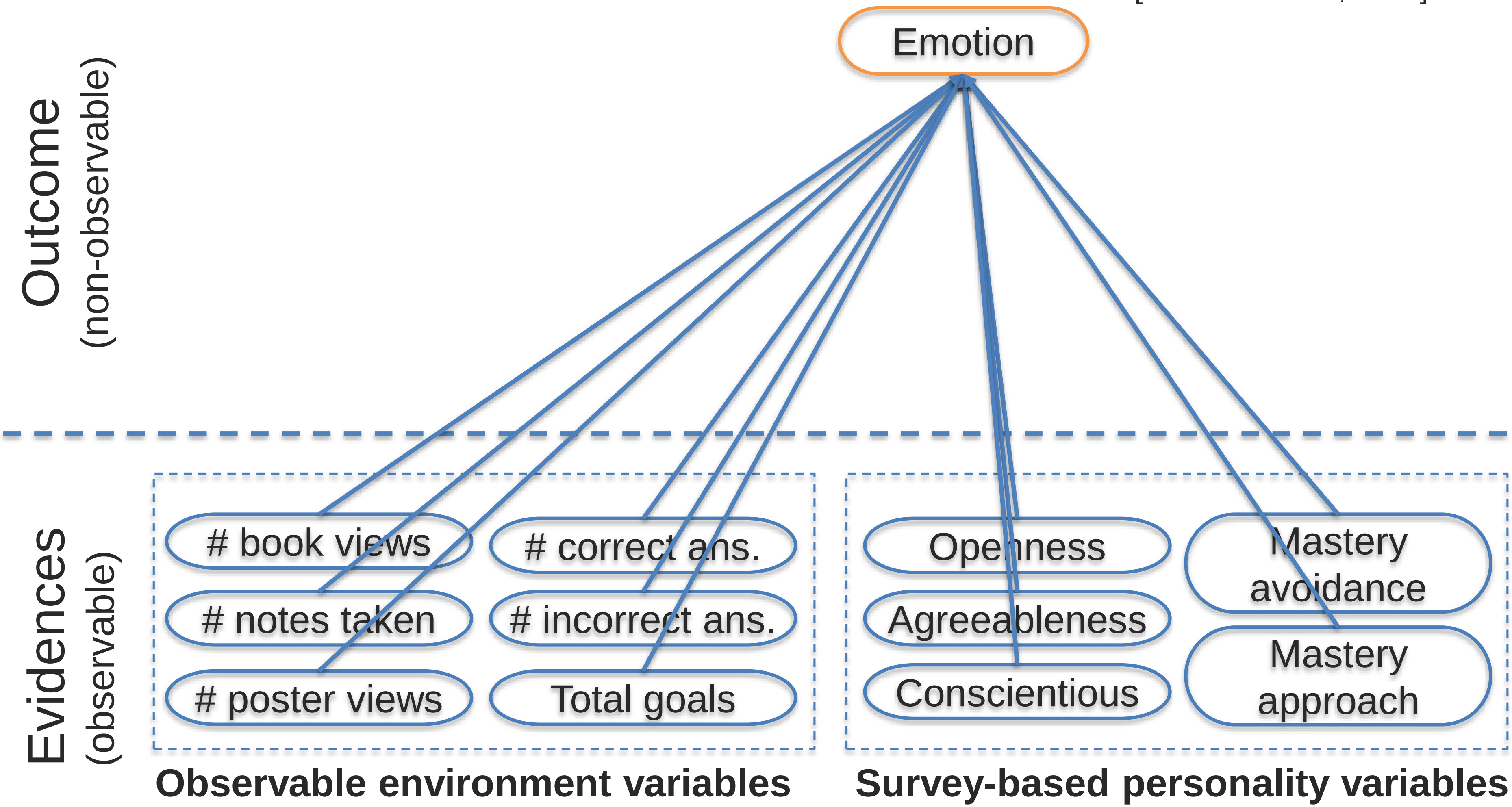
Observable environment variables

Openness
Agreeableness
Conscientious
Mastery avoidance
Mastery approach

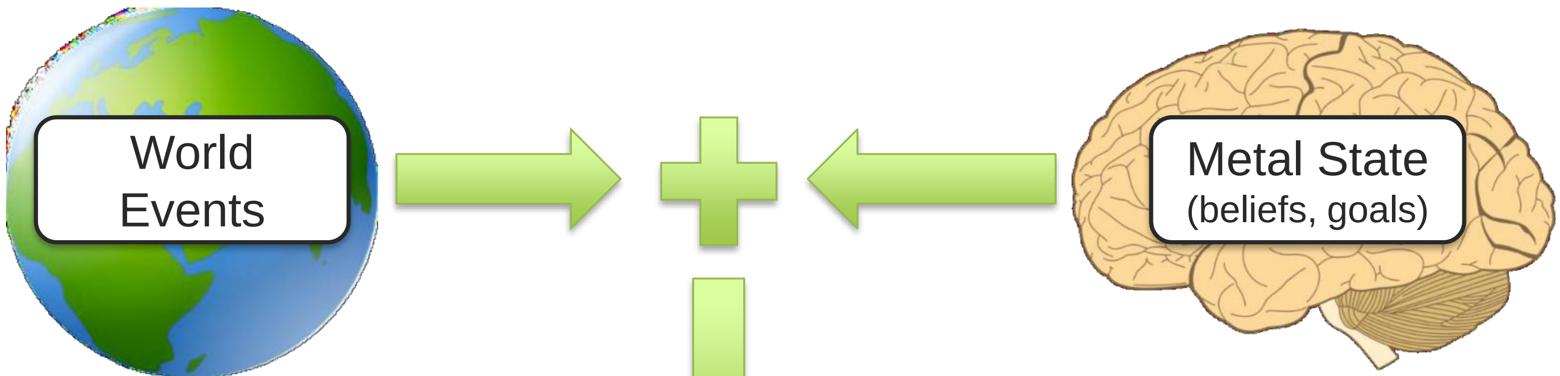
Survey-based personality variables

Example: Naïve Bayes Approach

[Sabourin et al., 2011]

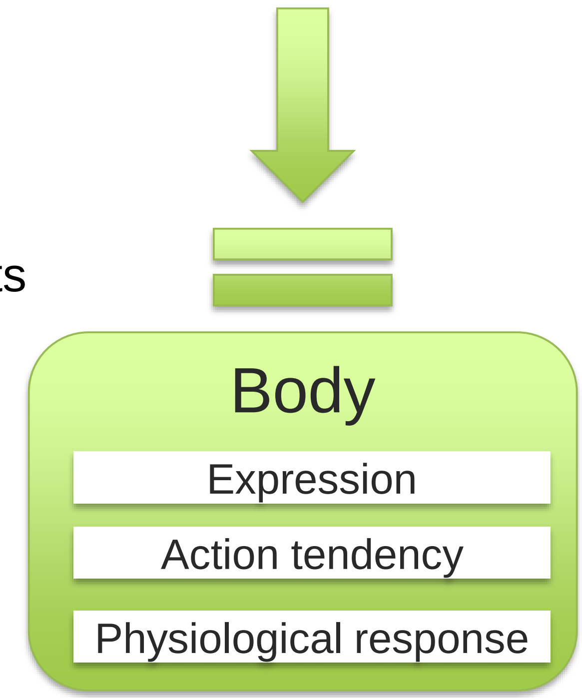


Appraisal Theory of Emotion



Argues for importance of three interrelated concepts

- World events
- Mental state
- Emotional Response

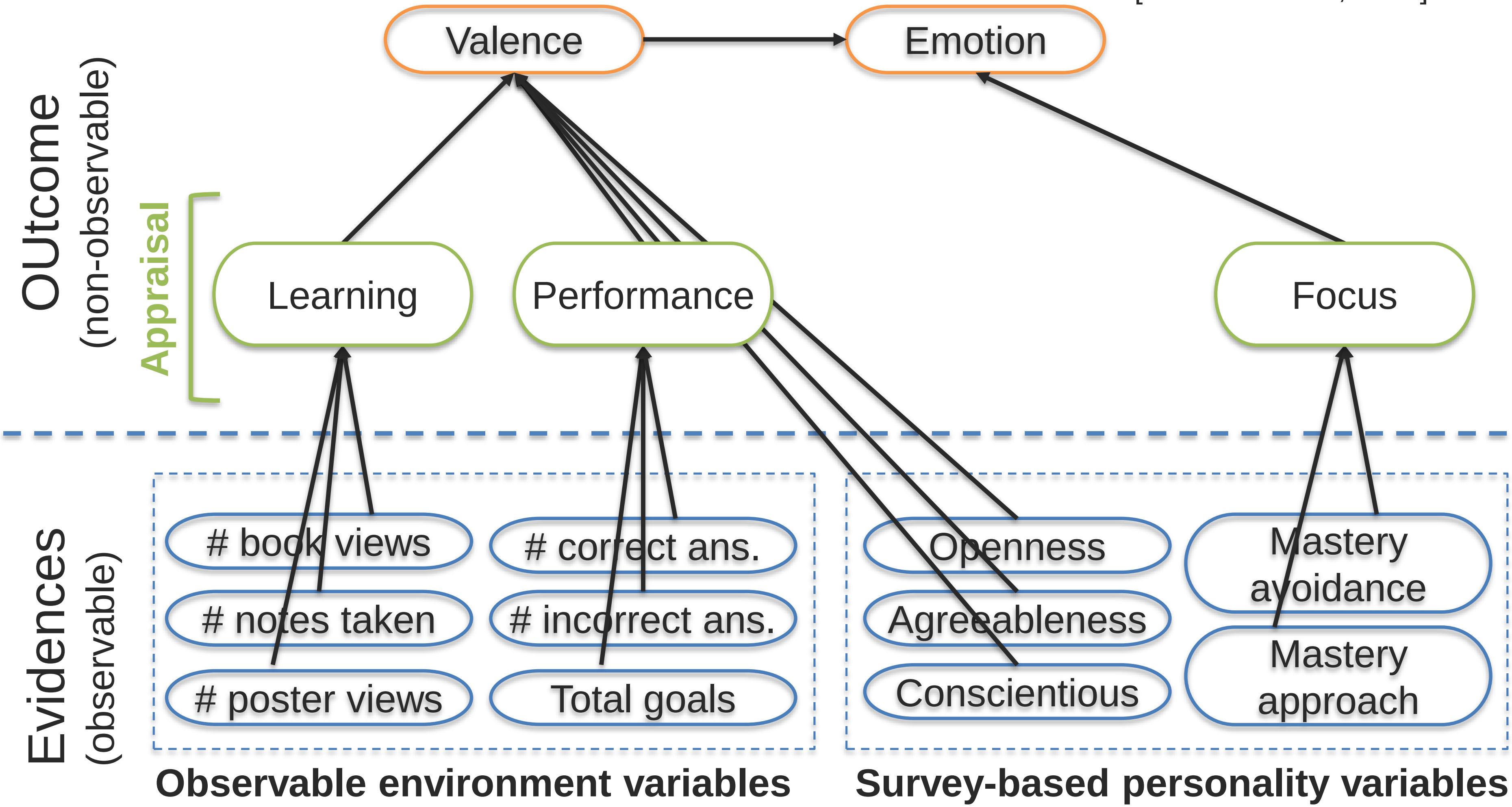


If we know two of these variables, we can make predictions about the third

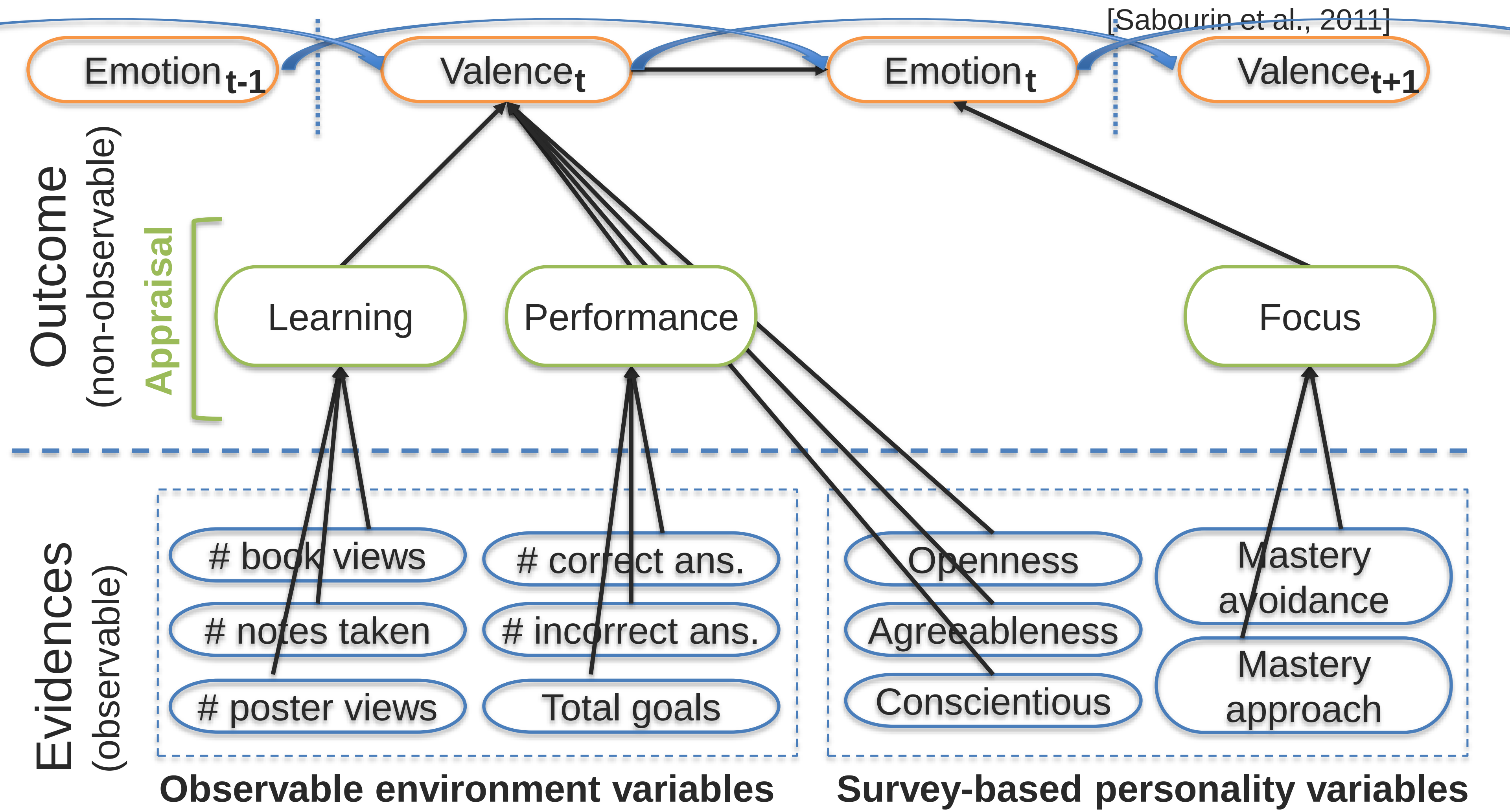
Response= f(Env., Mind)

Example: Bayesian Network Approach

[Sabourin et al., 2011]

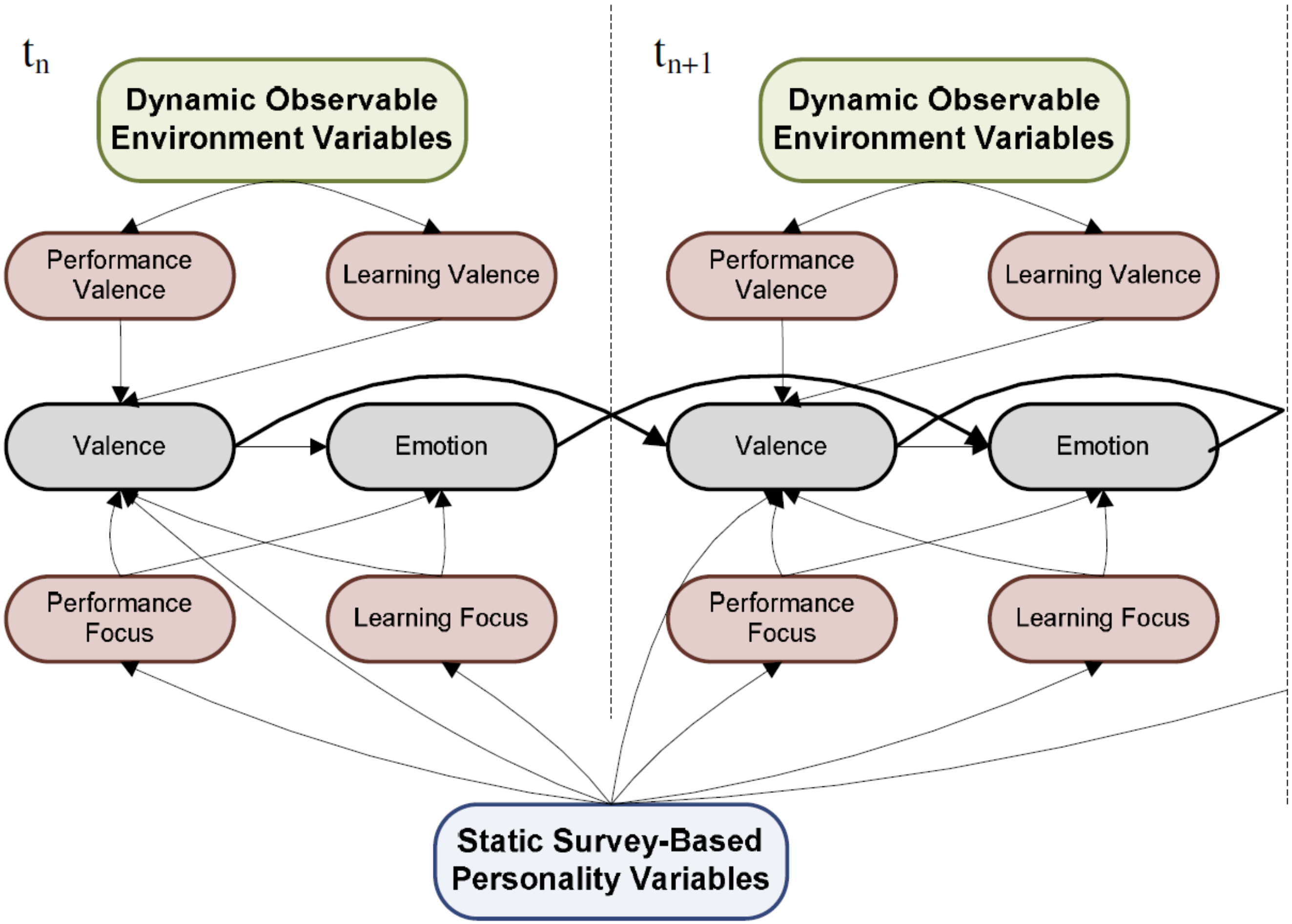


Example: Dynamic Bayesian Network Approach



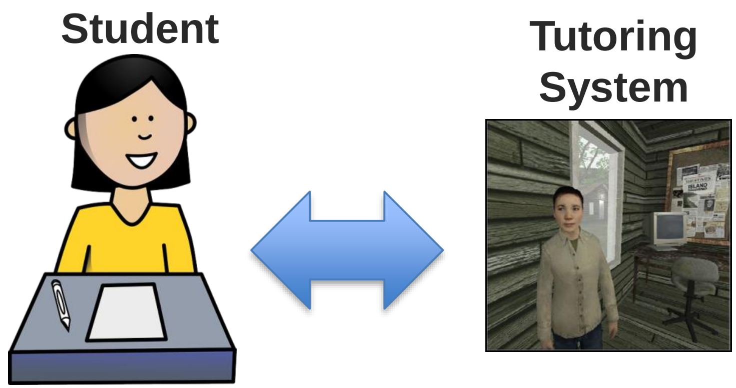
Example: Dynamic Bayesian Network Approach

[Sabourin et al., 2011]



Example: Inferring Emotion from Interaction Logs

[Sabourin et al., 2011]



	Emotion Accuracy	Valence Accuracy
Baseline	22.4%	54.5%
Naïve Bayes	18.1%	51.2%
Bayes Net	25.5%	66.8%
Dynamic BN	32.6%	72.6%

Bayesian Networks

Bayesian networks

- A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions

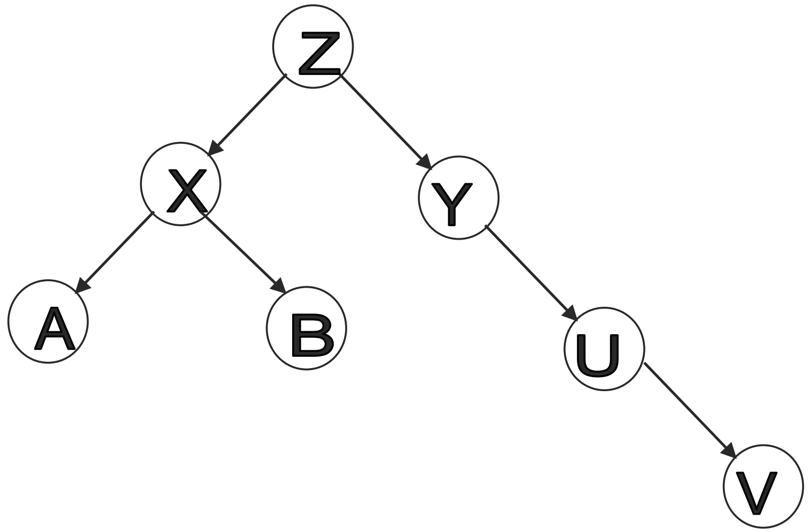
- Syntax:
  - a set of nodes, one per variable
  - a directed, acyclic graph (link ≈ "directly influences")
  - a conditional distribution for each node given its parents:

$$\mathbf{P}(X_i \mid \text{Parents}(X_i))$$

- In the simplest case, conditional distribution represented as a conditional probability distribution (CPD) giving the distribution over X_i for each combination of parent values

Bayesian Network (BN)

- A specific type of graphical model that is represented as a Directed Acyclic Graph.



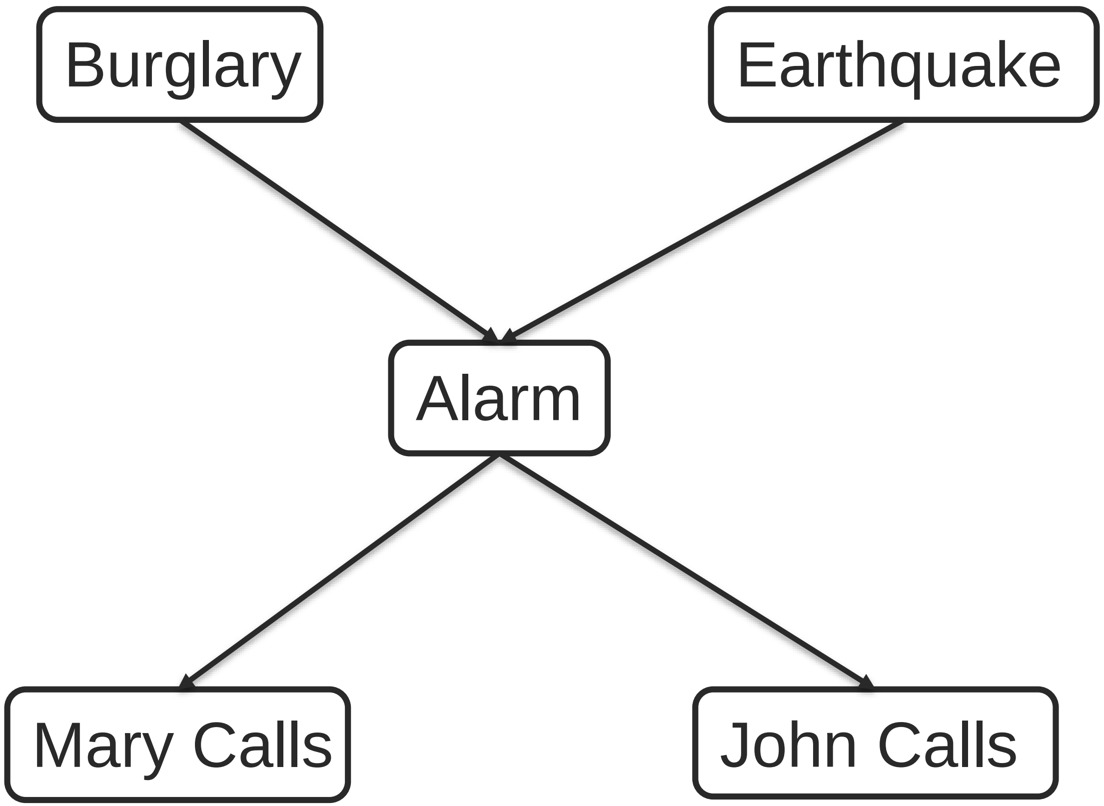
Example

“I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?”

- Variables?
  - *Burglary*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*

- “Causal" knowledge?
  - A burglar can set the alarm off
  - An earthquake can set the alarm off
  - The alarm can cause Mary to call
  - The alarm can cause John to call

Example – Network Topology



Joint Probability in Graphical Models

With chain-rule, the joint probability can be restated:

$$
\begin{aligned}
P(A,B,C,D,E) &= P(A|B,C,D,E)P(B,C,D,E) \\
&= P(A|B,C,D,E)P(B|C,D,E)P(C|D,E) \\
&= P(A|B,C,D,E)P(B|C,D,E)P(C,D,E) \\
&= P(A|B,C,D,E)P(B|C,D,E)P(C|D,E)P(D,E) \\
&= P(A|B,C,D,E)P(B|C,D,E)P(C|D,E)P(D|E)P(E)
\end{aligned}
$$

➡ The order in applying the chain-rule is arbitrary.

How can we simplify the joint probability even more, given the graphical model?

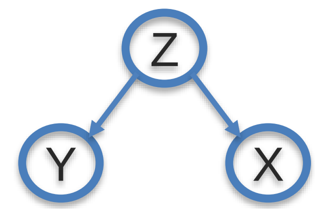
Joint Probability in Graphical Models

With chain-rule, the joint probability can be reshaped:

$$P(A,B,C,D,E) = P(A|B,C,D,E)P(B|C,D,E)P(C|D,E)P(D|E)P(E)$$

- Remember these concepts:

Independent variables



conditionally independent

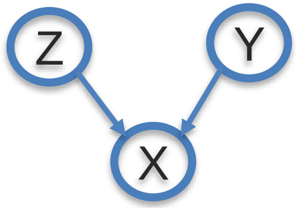
- In a Bayesian network, each conditional probability for a specific variable X only depends on its parents:

$$P(X|\ all\ variables) = \underline{P(X|parents(X))}$$

Conditional Probability Distribution (CPD)

Conditional Probability Distribution (CPD)

Given a variable X and its parents (Y and Z):



$$P(X|parents(X)) = P(X|Y,Z)$$

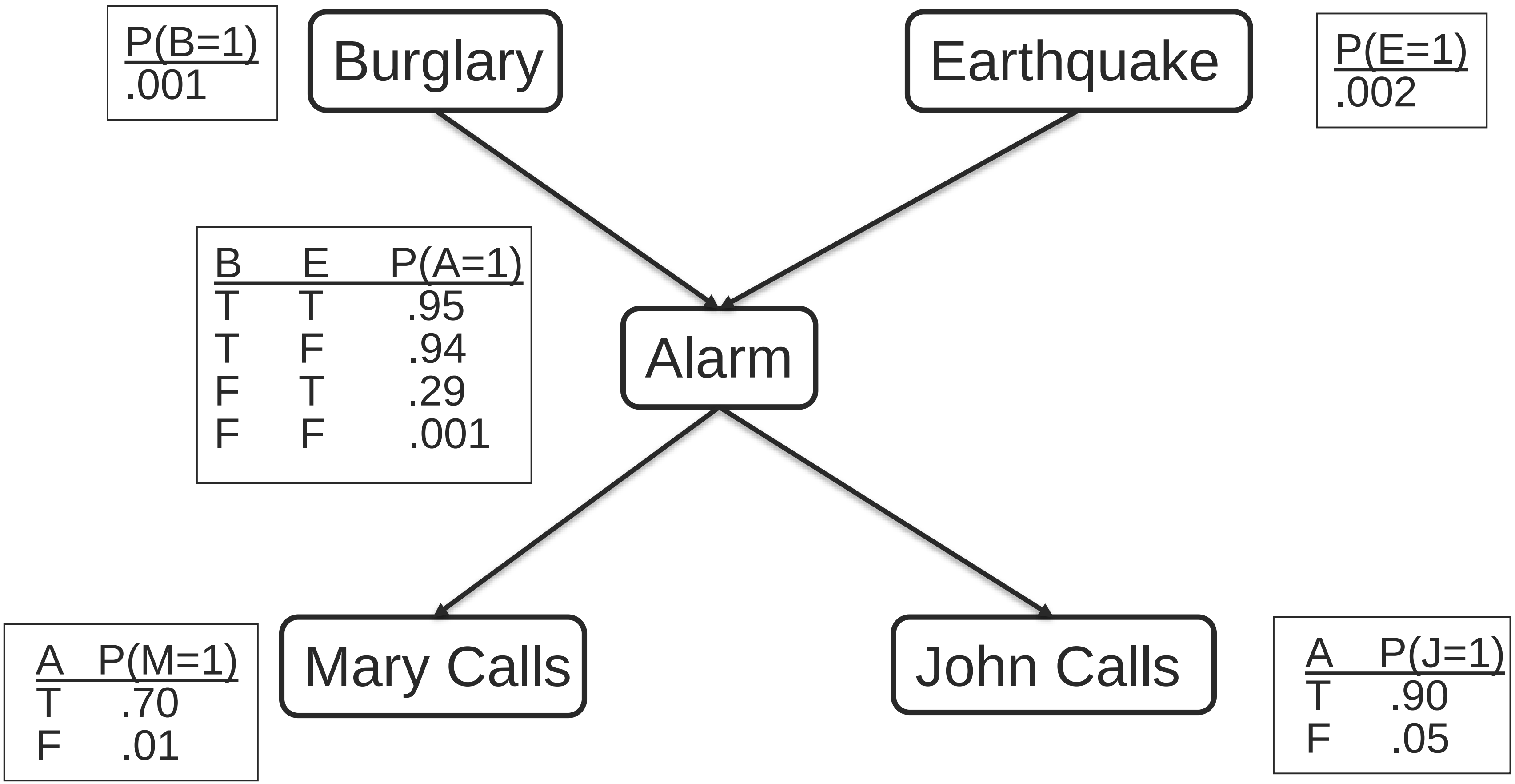
Definition: probability distribution of X when the assignment of it parents is known (Y and Z)

- ❑ For **categorical variable**: expressed as a conditional probability table

	Y=0	Y=1
P(X=0|Y)	4/6	1/3
P(X=1|Y)	2/6	2/3

- ❑ For **continuous variable**: expressed as a conditional density function
  - For example, multivariate normal density function or Gaussian linear regression (used by Bayes RegressionLinear Model)

Example – Conditional Probability Distributions



P(B=1)
.001

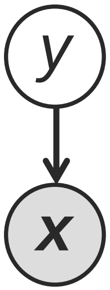
P(E=1)
.002

B	E	P(A=1)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

A	P(M=1)
T	.70
F	.01

A	P(J=1)
T	.90
F	.05

Generative Model: Naïve Bayes Classifier



Label : {0:`Dominant`, 1:`Not-dominant`}
(outcome)

Observation vector: [gaze, turn-taking,speech-energy]
(evidence)

Score function: $P(y = a|\boldsymbol{x_i})$

Likelihood

Prior

Chain rule

Bayes' theorem: $P(y|\boldsymbol{x}) = \dfrac{P(\boldsymbol{x}|y)P(y)}{P(\boldsymbol{x})} \approx P(\boldsymbol{x}|y)P(y) = P(\boldsymbol{x}, y)$

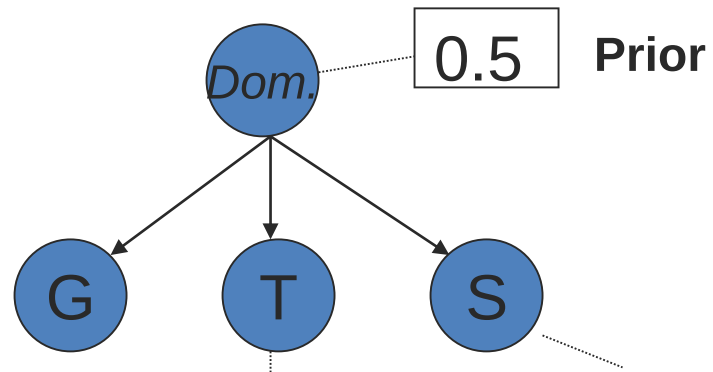
Posterior

Marginal likelihood
(partition)

$P(\boldsymbol{x}) = \sum_y P(\boldsymbol{x}|y)P(y)$

Naïve Bayes

- Strong assumption of the conditional independence of all feature variables.
- Feature variables only dependent on class variable



Conditional Probability Table (CPT):

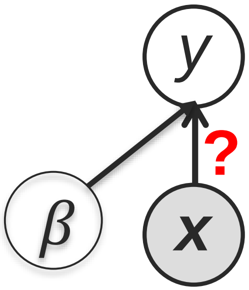
$P(G \mid dom) = 0.8$
$P(G \mid \neg dom) = 0.1$

$P(T \mid dom) = 0.7$
$P(T \mid \neg dom) = 0.4$

$P(S \mid dom) = 0.6$
$P(S \mid \neg dom) = 0.7$

$$
\begin{aligned}
P(dom, \neg G, T, \neg S) &= P(T \mid \neg G \cap \neg S \cap dom) \times P(\neg G \cap \neg S \cap dom) \\
&= P(T / dom) \times P(\neg G \mid \neg S \cap dom) \times P(\neg S \cap dom) \\
&= P(T / dom) \times P(\neg G \mid dom) \times P(\neg S \mid dom) \times P(dom) \\
&= P(dom) \prod_{i=1}^{p} P(x_i \mid dom) \\
&= 0.5 \times 0.6 \times 0.2 \times 0.3 = 0.018
\end{aligned}
$$

Bayesian Linear Regression Model



Label : {0:`Dominant`, 1:`Not-dominant`}
(dependent variable, outcome variable, response variable)

Observation vector: [gaze, turn-taking,speech-energy]
(independent variable, predictors)

Frequentist view: $y = \boldsymbol{\beta}^T \boldsymbol{x} + \epsilon$

Probabilistic view: $\epsilon \sim N(0, \sigma^2 I)$ $\quad y \sim N(\boldsymbol{\beta}^T \boldsymbol{x}, \sigma^2 I)$

"Prediction" score function would be: $p(y|\boldsymbol{x}, \beta, \sigma^2)$

But instead we are interested in the posterior distribution for the model parameters β:

$$p(\beta|\boldsymbol{x}, y, \sigma^2)$$

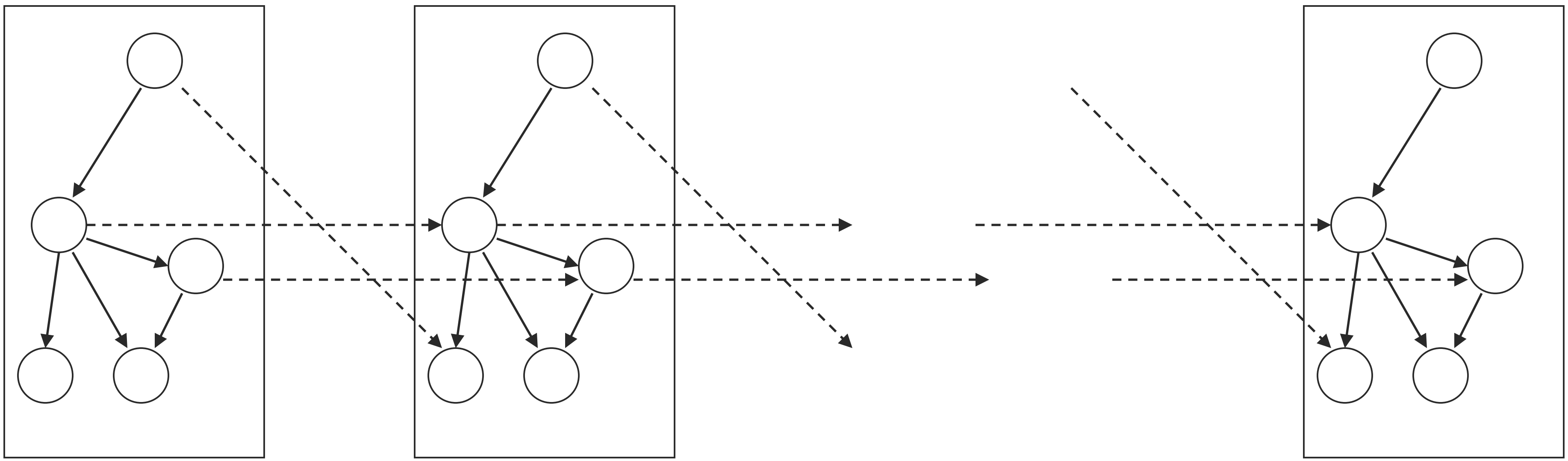
Dynamic Bayesian Network (DBN)

- Bayesian network with time-series to represent temporal dependencies.
- Dynamically changing or evolving over time.
- Directed graphical model of stochastic processes.
- Especially aiming at time series modeling.

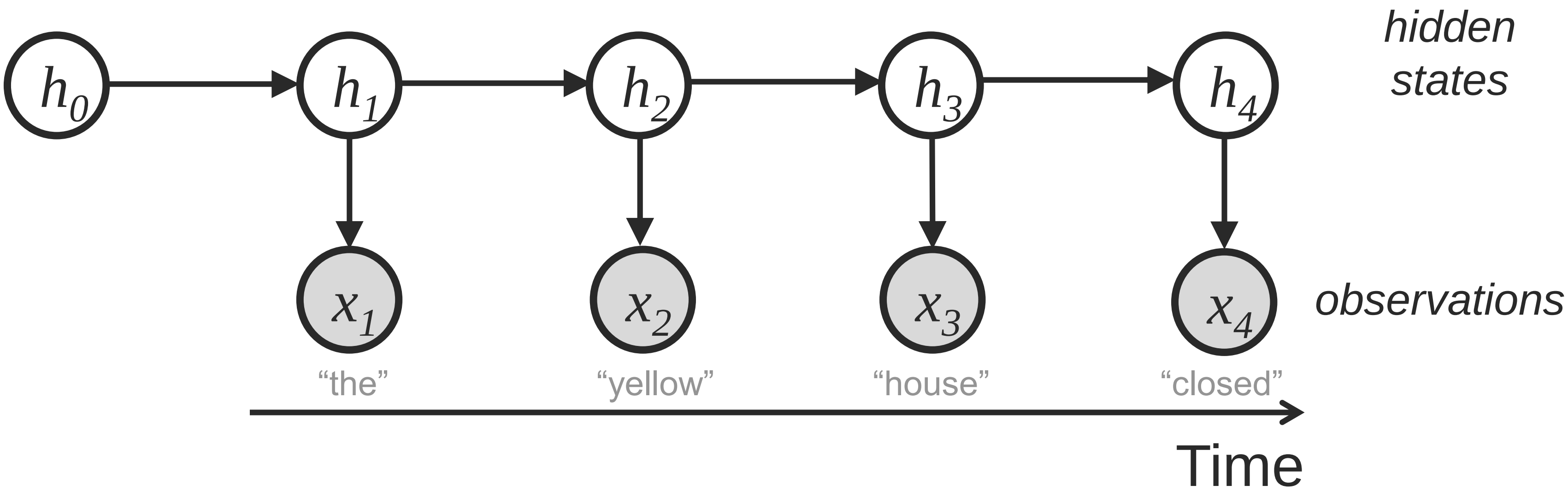
- Satisfying the Markovian condition:

  The state of a system at time t depends only on its immediate past state at time t-1.

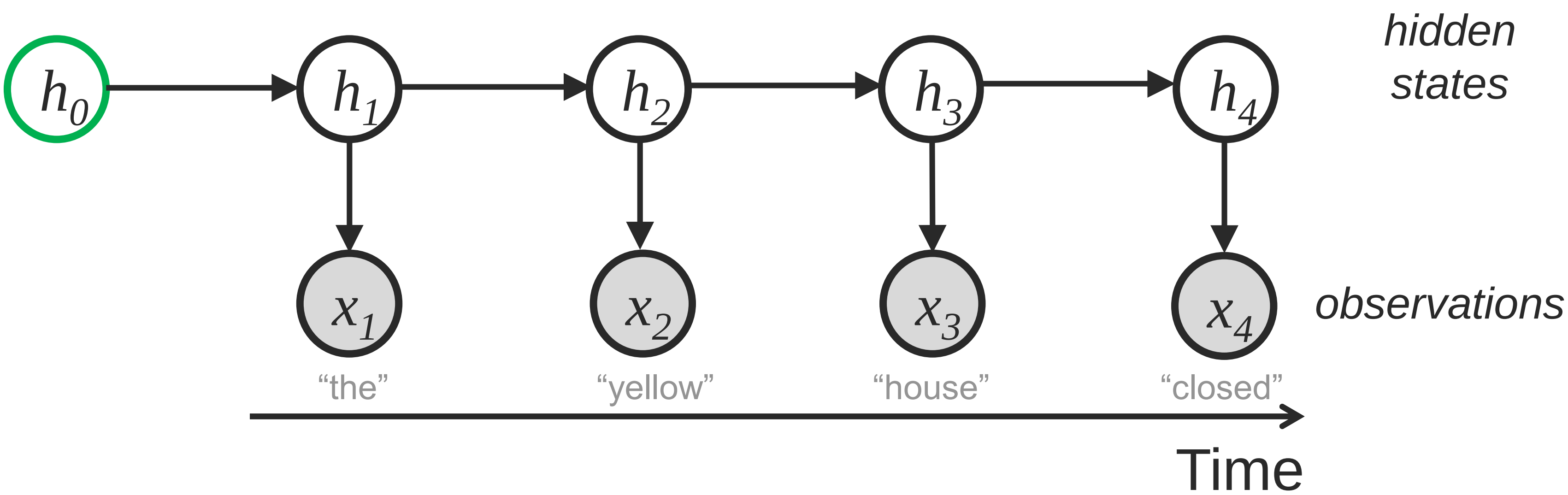
Dynamic Bayesian Network (DBN)



Hidden Markov Models

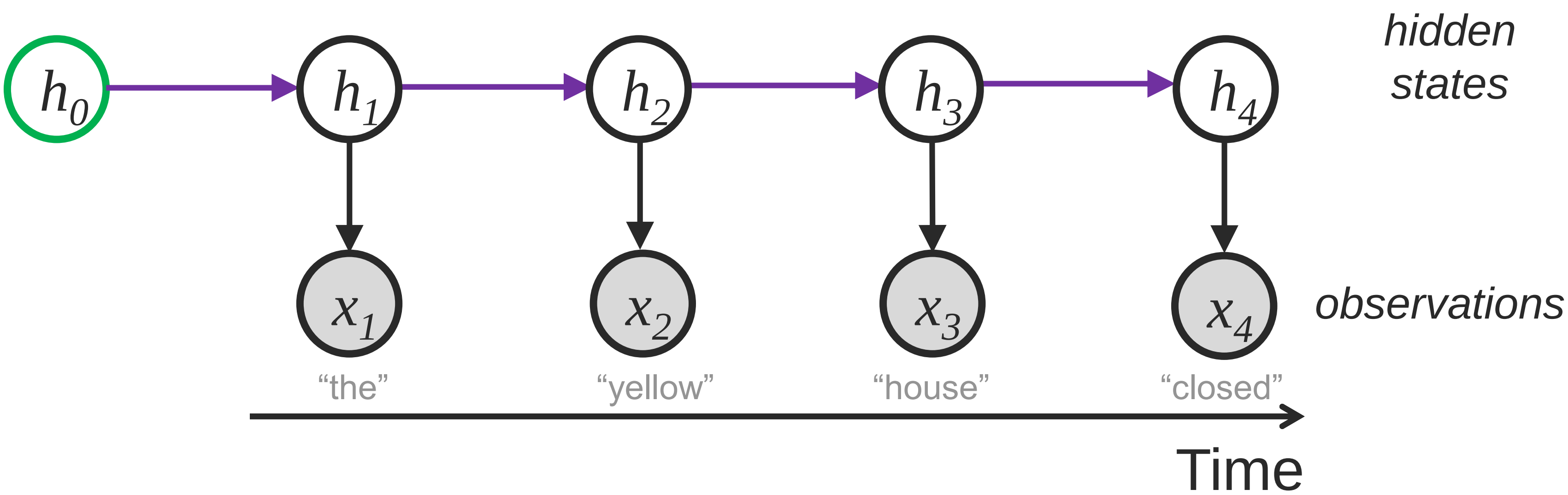


Hidden Markov Models



Initial state distribution $\boldsymbol{\pi}$ $\pi(i) = P(h_0 = i)$

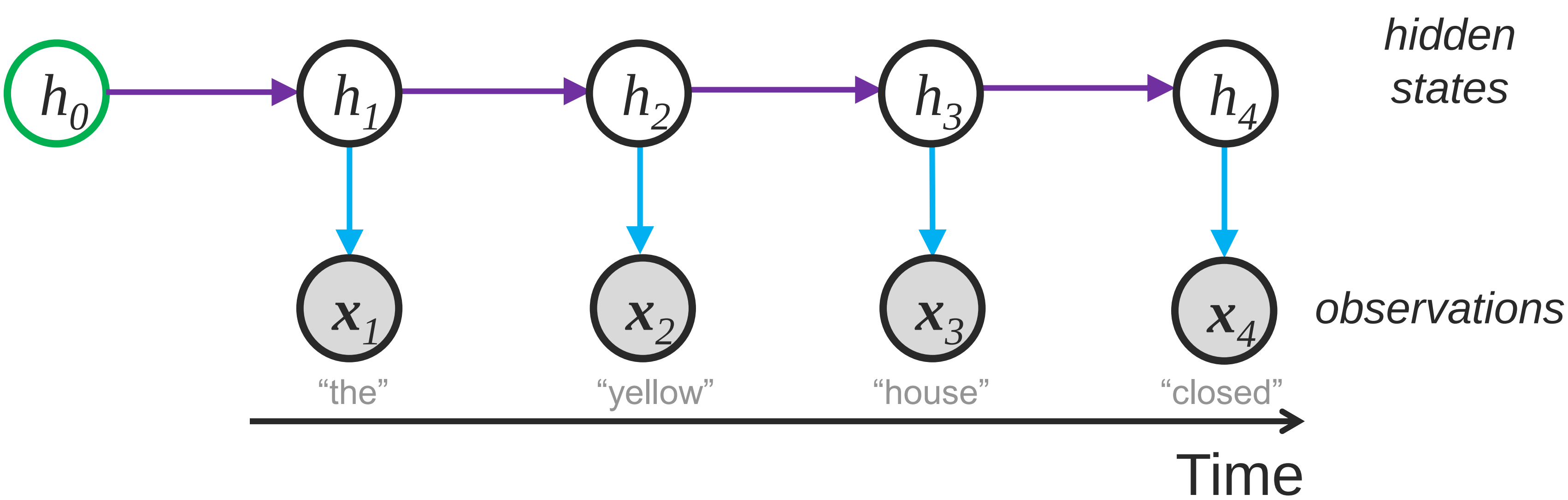
Hidden Markov Models



Initial state distribution $\boldsymbol{\pi}$ $\quad \pi(i) = P(h_0 = i)$

Transition probabilities $\boldsymbol{A}$ $\quad \mathrm{a}(i,j) = P(h_t = i | h_{t-1} = j)$

Hidden Markov Models

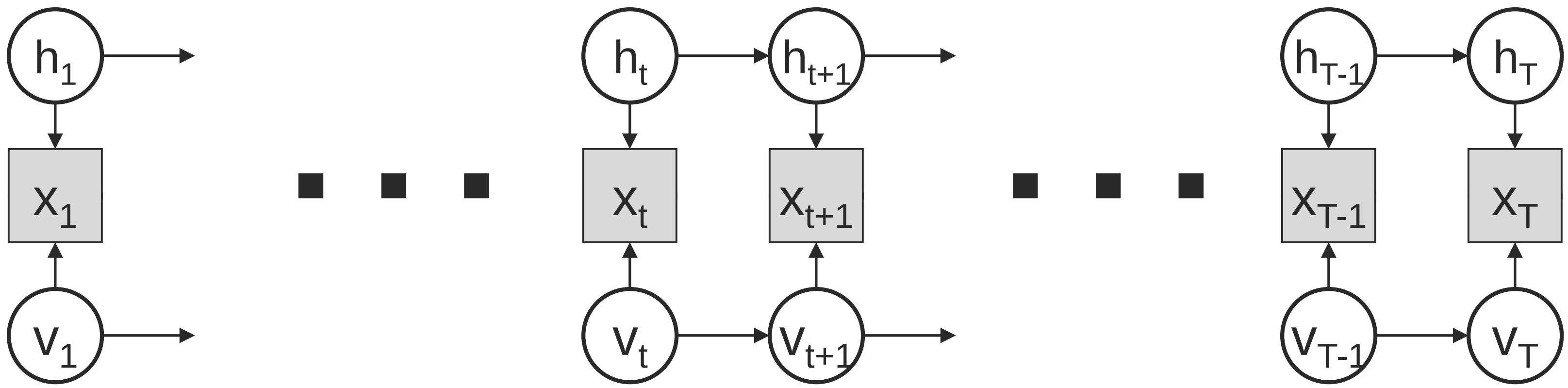


Initial state distribution $\boldsymbol{\pi}$ $\quad \pi(i) = P(h_0 = i)$

Transition probabilities $\boldsymbol{A}$ $\quad \mathrm{a}(i,j) = P(h_t = i | h_{t-1} = j)$

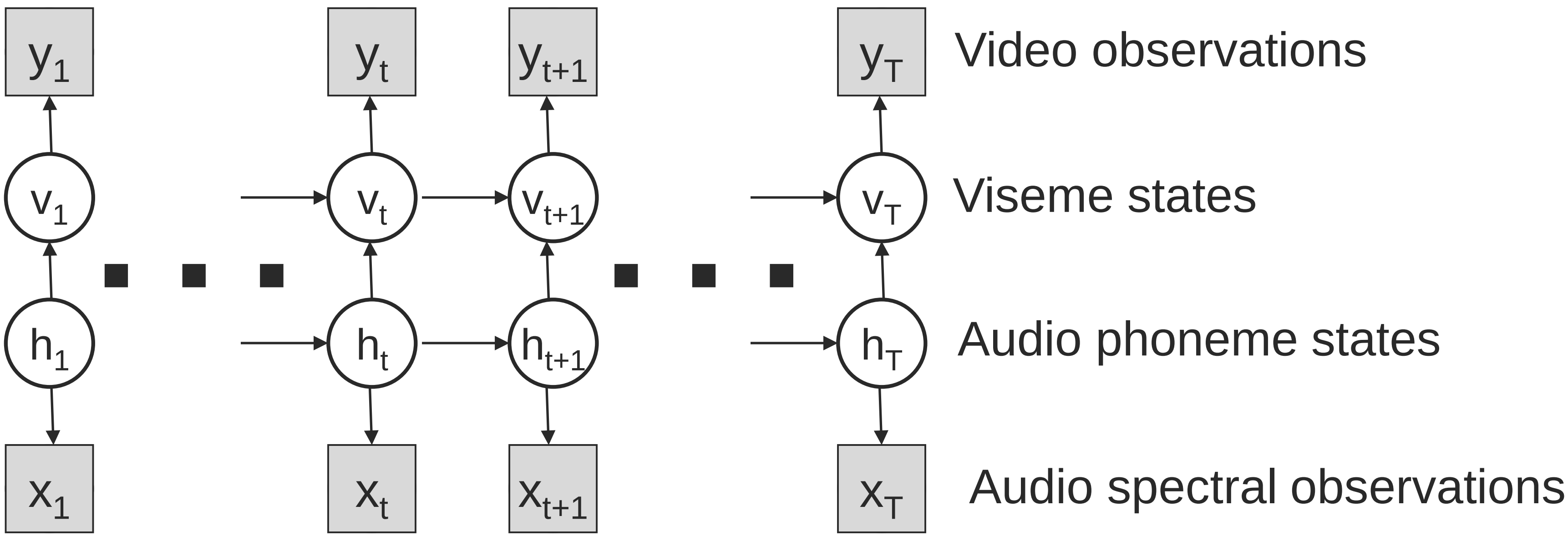
Emission Probabilities $\boldsymbol{B}$ $\quad b_t(i) = P(\boldsymbol{x}_t | h_t = i)$

Factorial HMM



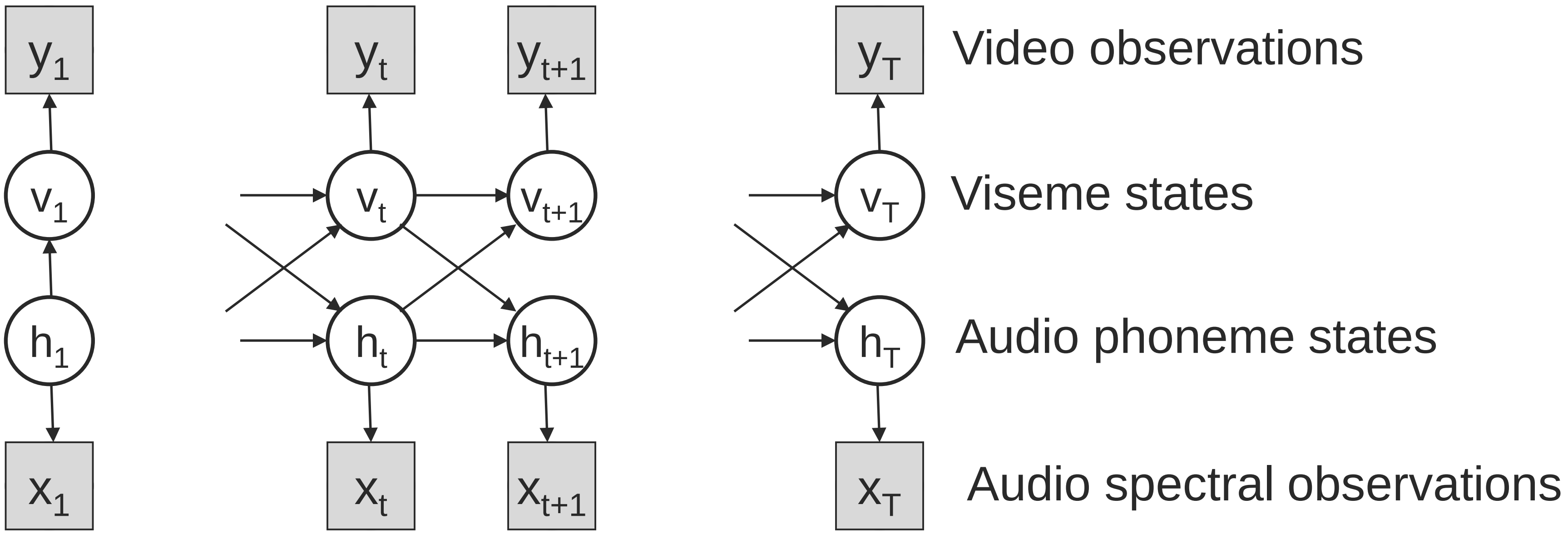
- Factorial HMM:
  - h_t and v_t represent two different types of background information, each with its own history
  - Observations x_t depend on both hidden processes
- Model parameters:
  - $p(v_{t+1}|v_t)$, $p(h_{t+1}|h_t)$, $p(x_t|h_t,v_t)$

The Boltzmann Zipper



- Both streams have a "memory" (h_t and v_t)
- Model parameters:
  - $p(h_{t+1}|h_t)$, $p(x_t|h_t)$
  - $p(v_{t+1}|v_t,h_{t+1})$, $p(y_t|h_t)$

The Coupled HMM



- Advantage over Boltzmann Zipper: More flexible, because neither vision nor sound is “privileged” over the other.
  - $p(h_{t+1}|v_t,h_t)$, $p(x_t|h_t)$
  - $p(v_{t+1}|v_t,h_t)$, $p(y_t|h_t)$

Learning (Dynamic) Bayesian Networks

- Multiple techniques exist to learn the model parameters based on data
  - Maximum likelihood estimator
  - Bayesian estimator, which allows to include prior information
- Python libraries:
  - http://pgmpy.org/
  - http://www.bayespy.org
  - https://pomegranate.readthedocs.io/en/latest/

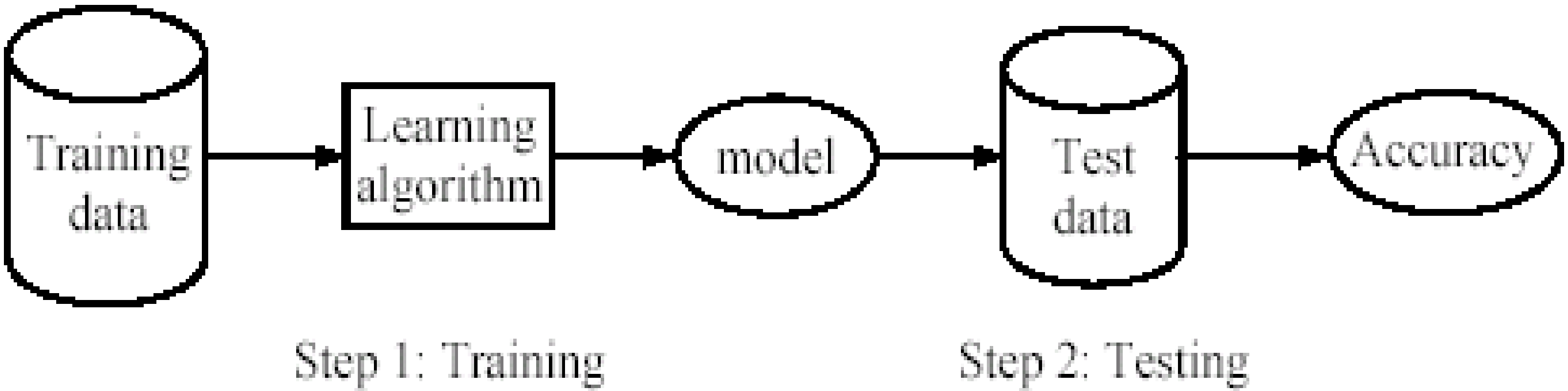
Machine Learning: Evaluation Methods

Supervised learning process: two steps

Learning (training): Learn a model using the training data

Testing: Test the model using unseen test data to assess the model accuracy

$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$



Evaluation methods

- Holdout set: The available data set *D* is divided into two disjoint subsets,
  - the *training set* D_{train} (for learning a model)
  - the *test set* D_{test} (for testing the model)
- Important: training set should not be used in testing and the test set should not be used in learning.
  - Unseen test set provides a unbiased estimate of accuracy.
- The test set is also called the holdout set. (the examples in the original data set *D* are all labeled with classes.)
- This method is mainly used when the data set *D* is large.
- **Unless building person specific models the training and test sets should not contain the same person**

Evaluation methods (cont…)

- n-fold cross-validation: The available data is partitioned into *n* equal-size disjoint subsets.
- Use each subset as the test set and combine the rest *n*-1 subsets as the training set to learn a classifier.
- The procedure is run *n* times, which give *n* accuracies.
- The final estimated accuracy of learning is the average of the *n* accuracies.
- 10-fold and 5-fold cross-validations are commonly used.
- This method is used when the available data is not large.

Evaluation methods (cont…)

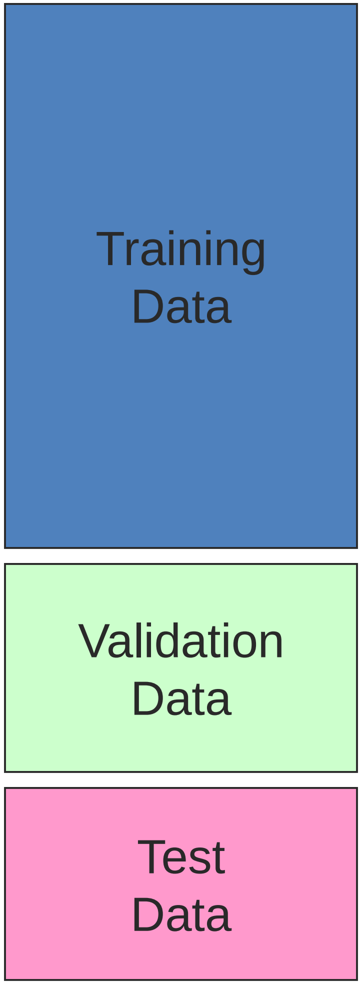
- Leave-one-out cross-validation: This method is used when the data set is very small.
- It is a special case of cross-validation
- Each fold of the cross validation has only a single test example and all the rest of the data is used in training.
- If the original data has *m* examples, this is *m*-fold cross-validation

Hyperparameters

- How do we determine C or γ for SVM training?
- Parameters that we do not learn through optimization are called hyper-parameters
- Need a way to find optimal values for our task
  - For some approaches rules of thumb exist
- Need an analytical way to do it
- Common ways
  - Grid search
  - Random search (not as bad as it sounds)

Training and Validation

- Data: labeled instances, e.g. emails marked spam/ham
  - Training set
  - Validation set
  - Test set
- Training
  - Estimate parameters on training set
  - Tune hyperparameters on validation/development set
  - Report results on test set
  - Anything short of this yields over-optimistic claims
- Evaluation
  - Many different metrics
  - Ideally, the criteria used to train the classifier should be closely related to those used to evaluate the classifier
- Statistical issues
  - Want a classifier which does well on *test* data
  - Overfitting: fitting the training data very closely, but not generalizing well
  - Error bars: want realistic (conservative) estimates of accuracy



Take home

- 1. Never touch test data during training/validation
- 2. Never touch test data during training/validation
- 3. Never touch test data during training/validation

Machine Learning: Measuring Error

Measuring Error

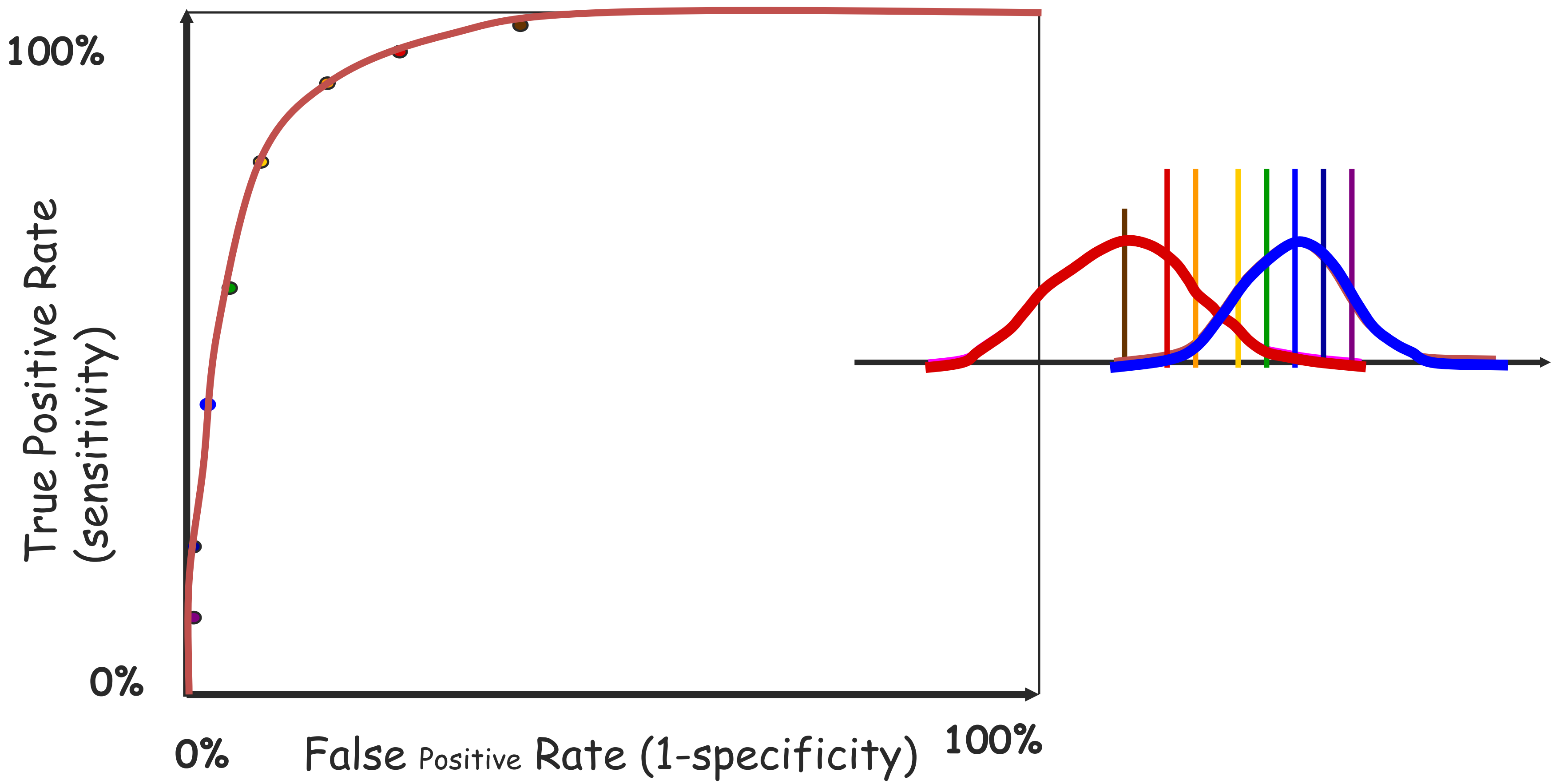
	Predicted class	
True Class	Yes	No
Yes	TP: True Positive	FN: False Negative
No	FP: False Positive	TN: True Negative

- Error rate = # of errors / # of instances = (FN+FP) / N
- Recall = # of found positives / # of positives
  = TP / (TP+FN) = sensitivity = hit rate
- Precision = # of found positives / # of found
  = TP / (TP+FP)
- Specificity = TN / (TN+FP)
- False alarm rate = FP / (FP+TN) = 1 - Specificity

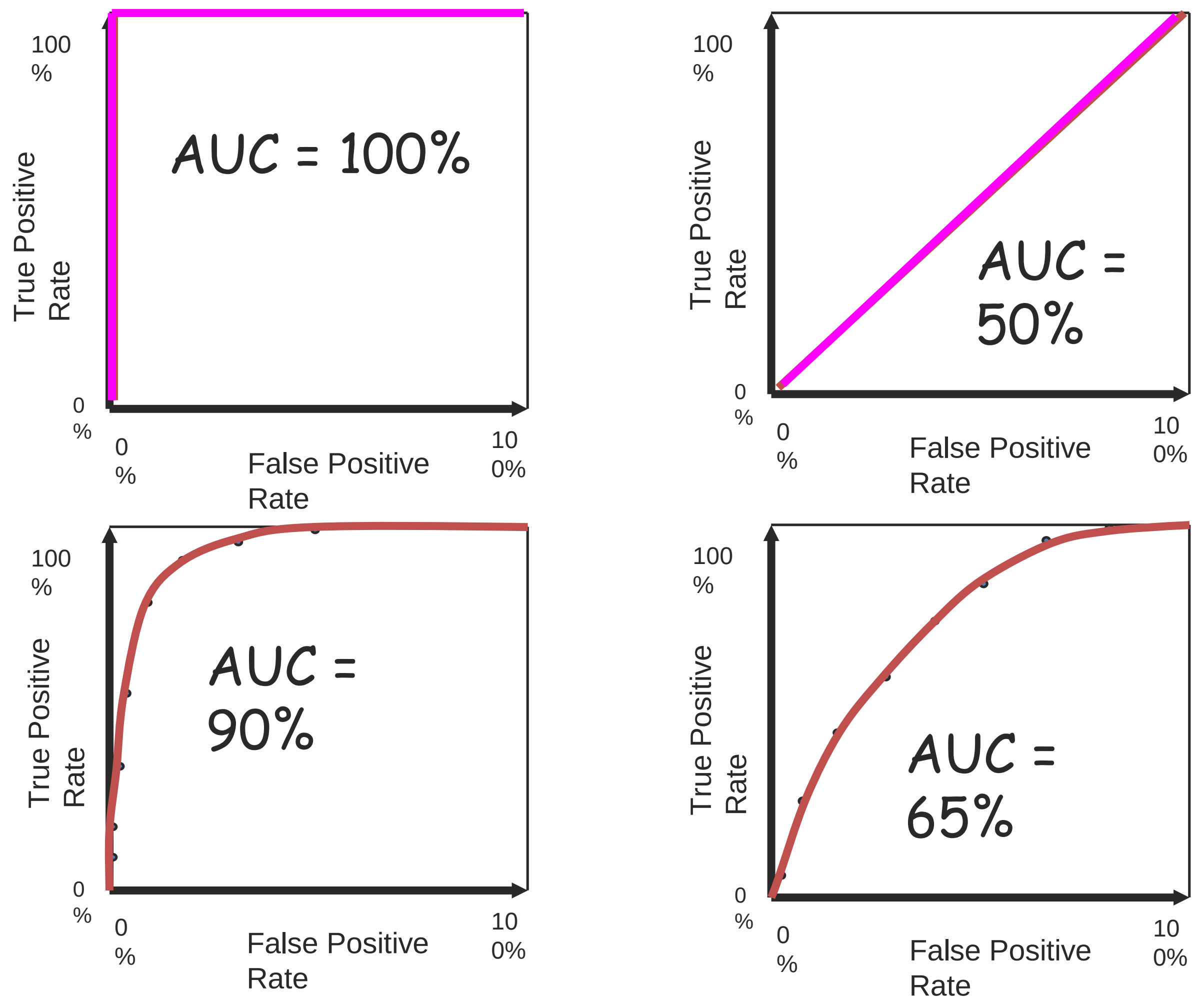
F_1-value (also called F_1-score)

- It is hard to compare two classifiers using two measures. F_1 score combines precision and recall into one measure
  - $F_1 = \frac{2 \cdot p \cdot r}{p+r}$
  - F_1 - score is the harmonic mean of precision and recall
  - $F_1 = \frac{2}{\frac{1}{p}+\frac{1}{r}}$
- The harmonic mean of two numbers tends to be closer to the smaller of the two
- Preferred over accuracy when data is unbalanced
  - Why?

Receiver Operating Characteristic (ROC) Curve



AUC for ROC curves



Evaluation of regression

- Root Mean Square Error
  - $\sqrt{\sum_i (y_i - x_i)^2}$
  - Not easily interpretable

- Correlation – trend prediction in a way
  - Nice interpretation: 0 – no relationship, 1 – perfect relationship
  - $\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{(n-1)\sigma_x \sigma_y}$

- Concordance Correlation Coefficient (CCC)
  - A method to combine both
  - $\rho_c = \frac{2\rho\sigma_x\sigma_y}{\sigma_x^2 + \sigma_y^2 + (\mu_x - \mu_y)^2}$, ρ – correlation coefficient
  - Has nice interpretability as well

Take home

- Error measure selection is not straightforward
  - Pick the right one for your problem
  - F1, AUC, Accuracy, RMSE, CCC
- Make sure the same measure is used for validation and testing
  - Otherwise you might be learning suboptimal models
- Wrong error measure can hide both bad and good results