ISE 540
Text Analytics

Mayank Kejriwal

Research Assistant Professor/Research Lead

Department of Industrial and Systems Engineering

Information Sciences Institute

USC Viterbi School of Engineering

kejriwal@isi.edu

‘Bayesian Networks’ (BNs)

Each random variable is a node.

Each node depends only on its parent.

Each node is conditionally independent of its siblings

Each node specifies a conditional probability table (CPT)

- I will not expect you to do *inference* or to determine values in CPTs in BNs but questions about *modeling* BNs (‘drawing’ a diagram given a *problem statement*) as well as verifying that conditional probability distributions are *valid* are all fair game
  - *We’ll give a nice exercise on this in the next quiz*
- Naïve Bayes is one ‘category’ of BNs (a very simple category)

FYI

$$p(C_k \mid x_1, \ldots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$

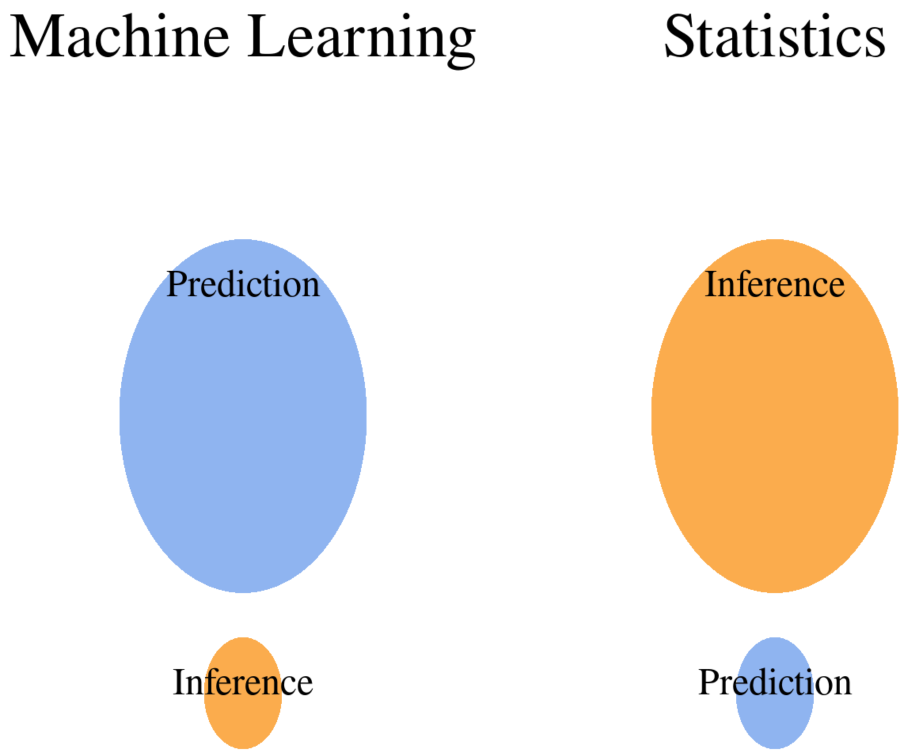
- Uses Bayes' Rule:

For events A and B, provided that $P(B) \neq 0$,

$$P(A \mid B) = \frac{P(B \mid A) P(A)}{P(B)}.$$

- For Naïve Bayes and directed graphical models P(B) can be hard to 'exactly' compute (e.g., P(coffee=True)
- Much easier to compute whether $P(A|B_1) > P(A|B_2)$

Inference vs. Prediction



- **Inference:** Use the model to learn about the data generation process.
- **Prediction:** Use the model to predict the outcomes for new data points.

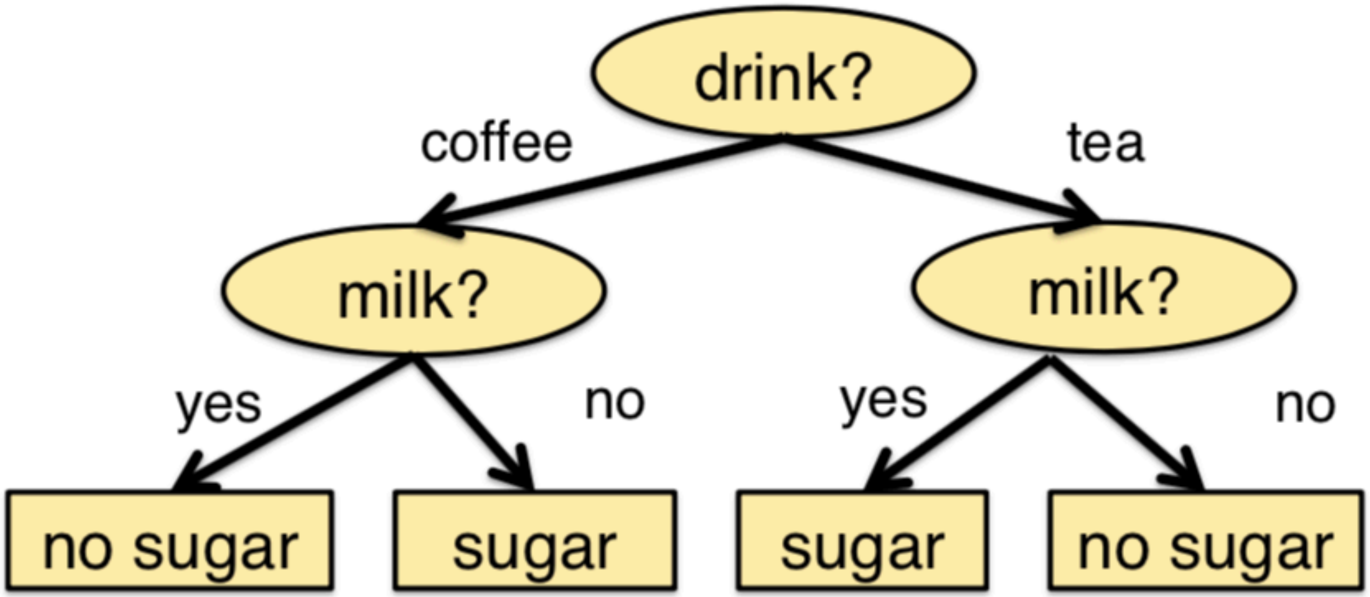
Are they really independent?

- No! But the end goal is usually one or the other…
  - We **can** use the Spam Naïve Bayes to ’infer’ P(‘million’=yes|spam=yes), P(‘billion’=yes,’million’=yes| spam=yes)
  - But the **main reason** we modeled the Naïve Bayes is to predict whether an email is more likely to be spam or not given its content (words)
- What about the student network?
  - Not clear, modeler may have intended for the model to predict the likelihood that the student would get a recommendation letter
  - But may also have intended to infer the intelligence of a student given other factors…

Continued

- In decision trees and other models (including neural networks), the goal is always to predict, not to infer
  - Same for linear models such as linear regression etc.
- Model parameters are always inferred from existing data (in this sense, all models are on an even footing: they must all derive their 'parameters' from the same set of observations)
- You should know when your task is an inference task vs. a prediction task
  - Like so much else in applied analytics, a good argument for your case is more important than formalism

Decision trees



In this example, the attributes (drink; milk?) are not conditionally independent given the class ('sugar')

Will I play tennis?

Features:

- Outlook: Sun, Overcast, Rain
- Temperature: Hot, Mild, Cool
- Humidity: High, Normal, Low
- Wind: Strong, Weak
- Label: +, -

Features are evaluated in the morning

Tennis is played in the afternoon

Training data

1.	S H H W	-
2.	S H H S	-
3.	O H H W	+
4.	R M H W	+
5.	R C N W	+
6.	R C N S	-
7.	O C N S	+
8.	S M H W	-
9.	S C N W	+
10.	R M N W	+
11.	S M N S	+
12.	O M H S	+
13.	O H N W	+
14.	R M H S	-

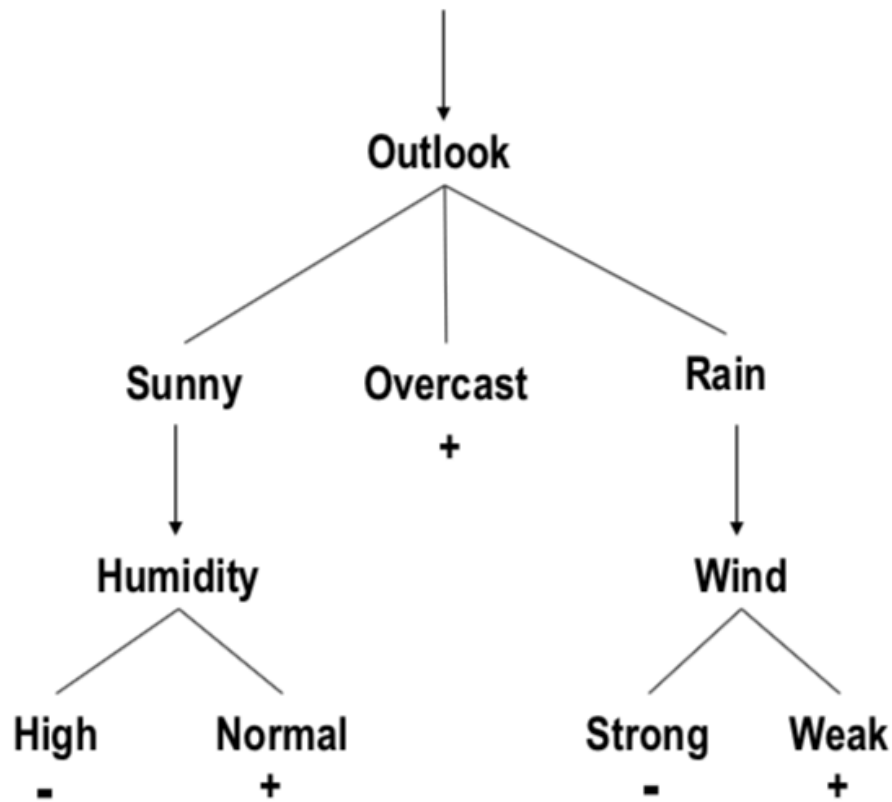
Outlook: S, O, R

Temp: H, M, C

Humidity: H, N, L

Wind: S, W

Decision tree (how did we *learn* this?)

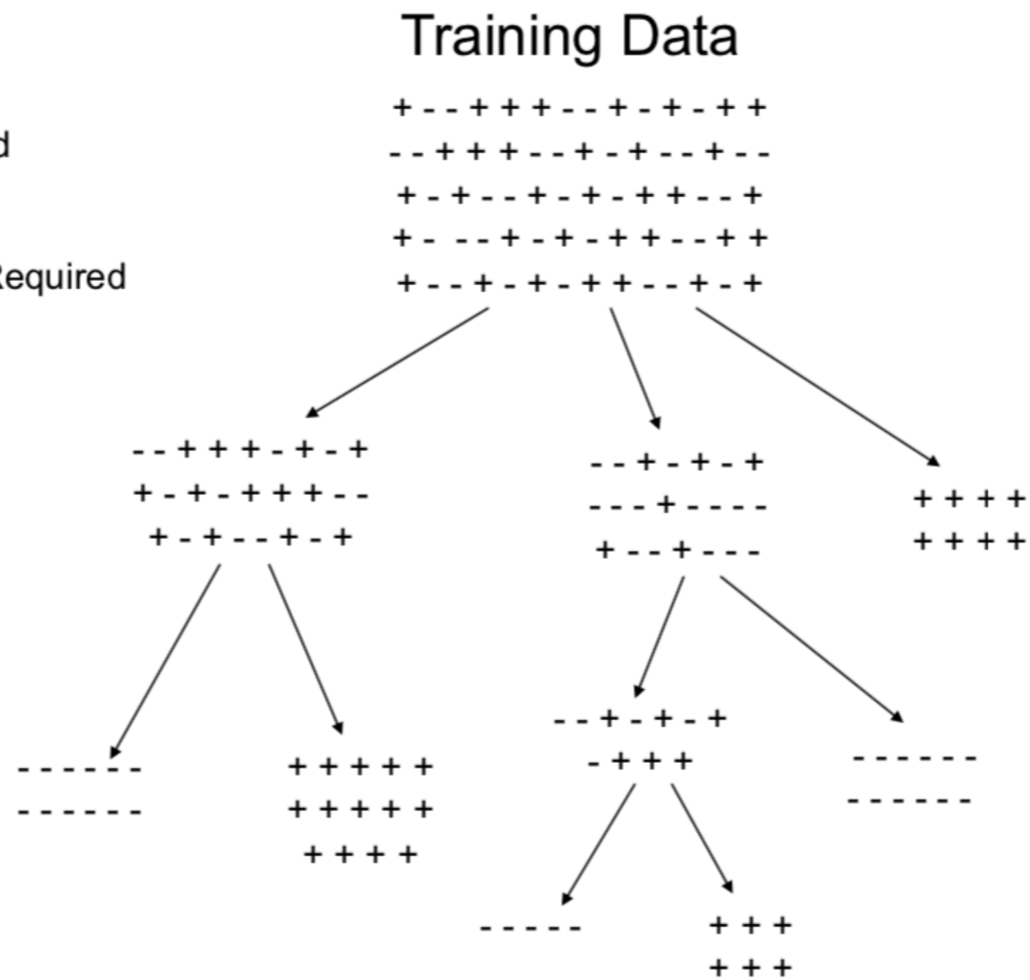


Intuition

Highly Disorganized

High Entropy

Much Information Required



Highly Organized

Low Entropy

Little Information Required

Some details on how to split

- We split the tree at each **node S** (why not each ‘level’?); the decision to be made is, which attribute to use for the split?
- We try out all attributes and choose the one with the maximum information gain (IG)
- IG can be defined in several ways but most common choice is based on entropy (H): https://en.wikipedia.org/wiki/Entropy_(information_theory)
- The IG of attribute A if used for the split at node S (parent) is given by the formula below, assuming V(A) are the possible values for A e.g., V(Outlook)={Sunny, Overcast, Rain}

$$Gain(S_{parent}, A) = H(S_{parent}) - \sum_{i \in V(A)}^{N} H(S_{child_i}) \frac{|S_{child_i}|}{|S_{parent}|}$$

Aside: how many different decision trees are there?

With n Boolean attributes, there are 2^n possible kinds of examples.

One decision tree = assign *true* to one subset of these 2^n kinds of examples.

There are 2^{2^n} possible decision trees!
(10 attributes: $2^{1024} \approx 10^{308}$ trees;
20 attributes $\approx 10^{300,000}$ trees)

What makes a learning problem ‘hard’?

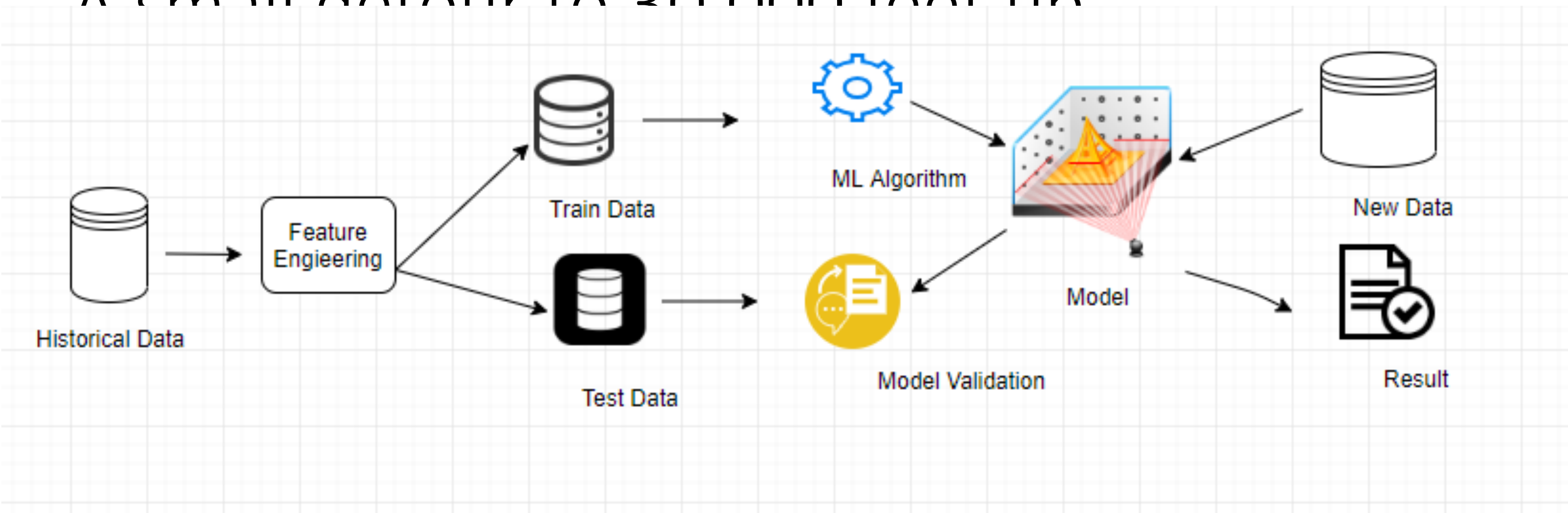
How do we measure “hard”?

- Computation time?
- Space complexity?
- Number of training examples required?

Hard learning problems require more training examples

The hardest learning problems require the entire example space to be labeled

A small detour to 30,000 feet up



- Issues that are always up for debate and require a combination of art and science:
  - How to select the model and validate it?
  - How to do feature engineering?
  - How to avoid model and/or dataset bias?