Approaches to Classification

1. Apply Bayes Decision Rule via modeling of the likelihood
   - Linear Discriminant Analysis/ QDA
   - **Bayesian Belief Networks (Naïve Bayes is a special case)**

2. Apply Bayes Decision Rule via modeling the posterior directly
   - **Logistic Regression**
   - **Feedforward neural networks, including deep learning**
   - K-Nearest Neighbor

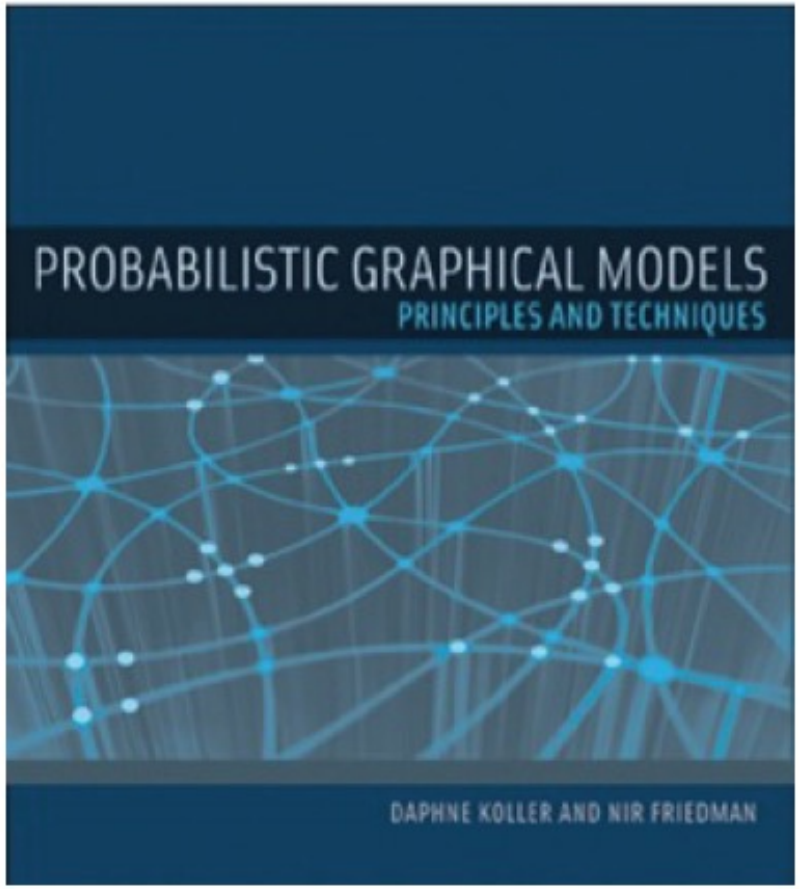
3. Focus just on Class Boundaries ("discriminative approaches"):
   - Decision trees
   - **Support Vector Machines (time permitting)**

Graphical Models

- Graphical models are (directed or undirected) graphs in which nodes represent random variables, and the lack of arcs represent (**assumed**) conditional independences.

- Allows making inferences on subsets of variables given any kind of information about some other variables (MUCH MORE general than Classification).
- Key concept in determining CAUSALITY.

(Way Beyond Classification)



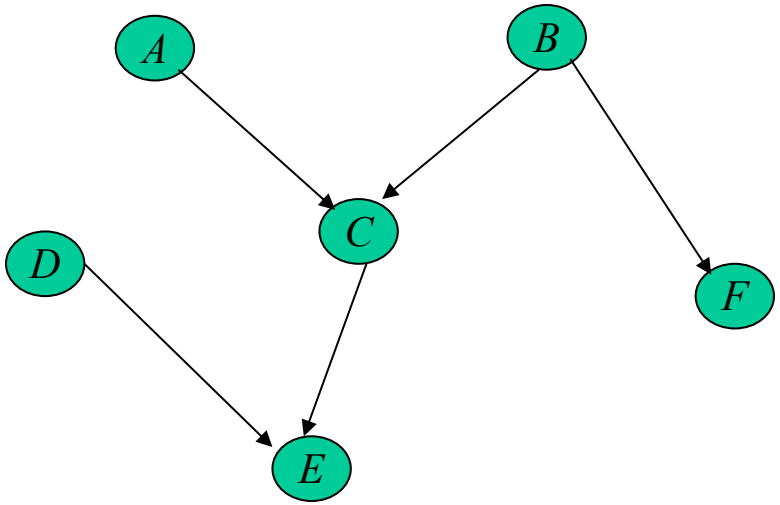
Daphne Koller
Computer Science Dept.
Stanford University

Nir Friedman
School of Computer Science & Engineering
Hebrew University

MIT Press, 2009, 1231 pages

Bayesian (Belief) Networks

- Directed Acyclic Graph on **all** variables.
- Allows combining prior knowledge about (in)dependences among variables with observed training data



1. Any variable is conditionally independent of all non-descendent variables given its parents.
2. Graph also imposes partial ordering, e.g. A,B,C,D,E,F
   From (1) and (2), get P(A,B,C,D,E,F) = P(A)P(B)P(C|A,B)P(D)P(E|C,D)P(F|B)

$= \Pi_i$ P(node i | parents of node i)

- *Naïve Bayes Classifier is a simple directed graphical model*

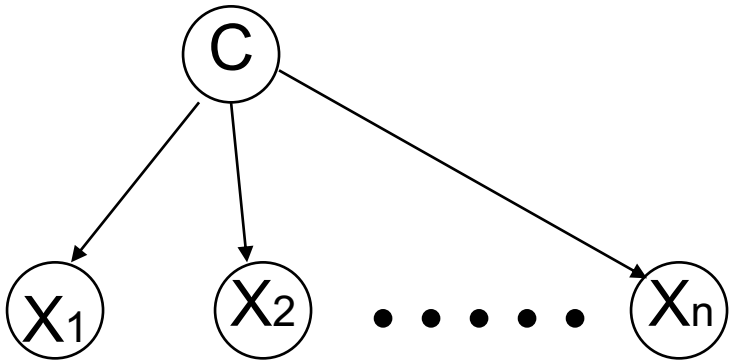
Naïve Bayes Classifier (KM Ch 9.3): Makes Conditional Independence assumption to simplify computation of likelihoods for each class.

1. *Assume independence among attributes x_i when class is given: ("independence of attributes conditioned on class variable").*

$P(x_1, x_2, \ldots, x_d | C_j) = P(x_1 | C_j)\, P(x_2 | C_j) \ldots P(x_d | C_j) = \Pi_i P(x_i | C_j)$

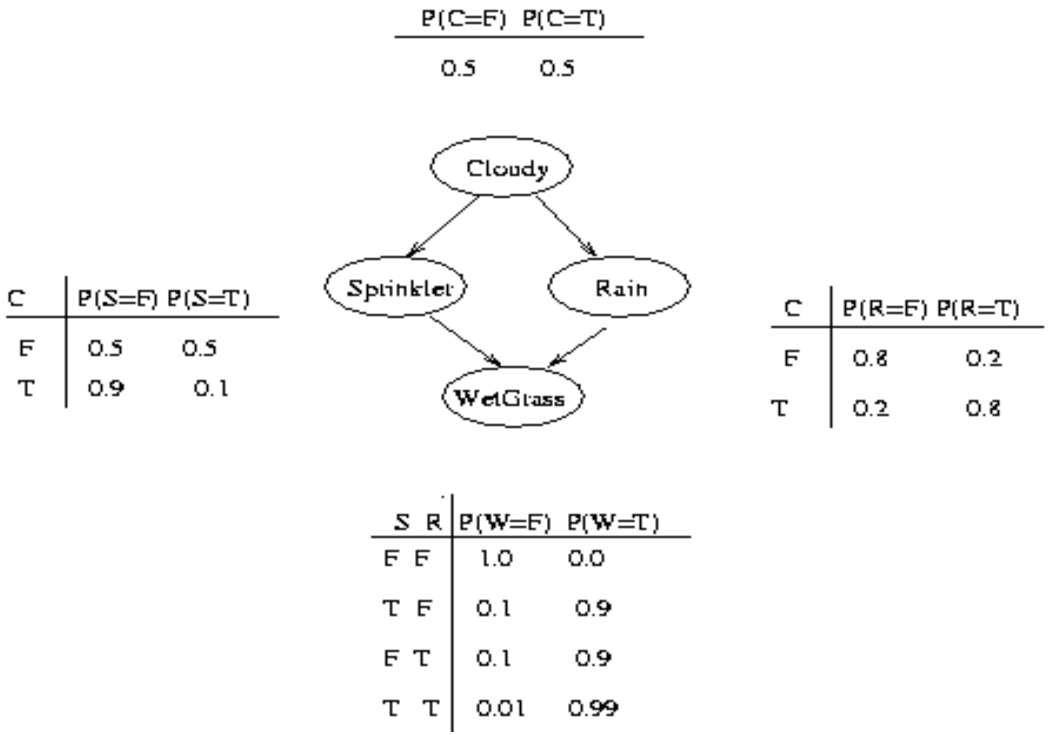
Note: Conditional independence not equal to attribute independence

2. *Estimate probabilities directly from data*



Example of BBN – Wet Grass (Murphy 01*)

1. Use training data to obtain
   Conditional Probability Distribution/Table (CPD or CPT) for each variable.
2. Use CPT to estimate probability of any "atomic event" (specific values of each of the variables), and hence any union of events
3. Use Bayes rule as needed.



*Can also see KM 3.6,

But Murphy 01 is more readable.

Video

Inference: Effect to Cause (Bottom Up)

We observe the grass is wet. Is this more likely due to sprinkler or because of rain?

Let T=1, F=0

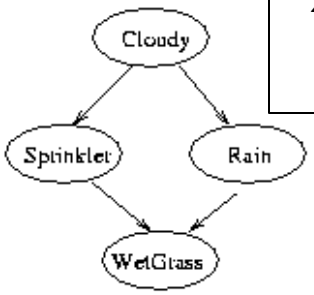
$P(S=1|W=1) =$

$\Sigma_{c,r} P(C=c,S=1, R=r, W=1) / \mathbf{P(W=1) = 0.2781/ .6471 = .43}$

$P(R=1|W=1) =$

$\Sigma c,s\ P(C=c,S=s, R=1, W=1) / P(W=1) = \mathbf{0.4581/ .6471 = .708}$

So more likely it is because of rain!



P(C=F)	P(C=T)
0.5	0.5

C	P(S=F)	P(S=T)
F	0.5	0.5
T	0.9	0.1

C	P(R=F)	P(R=T)
F	0.8	0.2
T	0.2	0.8

S	R	P(W=F)	P(W=T)
F	F	1.0	0.0
T	F	0.1	0.9
F	T	0.1	0.9
T	T	0.01	0.99

Special Types of inference

Diagnostic - B is evidence of A (bottom-up)

previous example

Predictive - A can cause B (top-down)

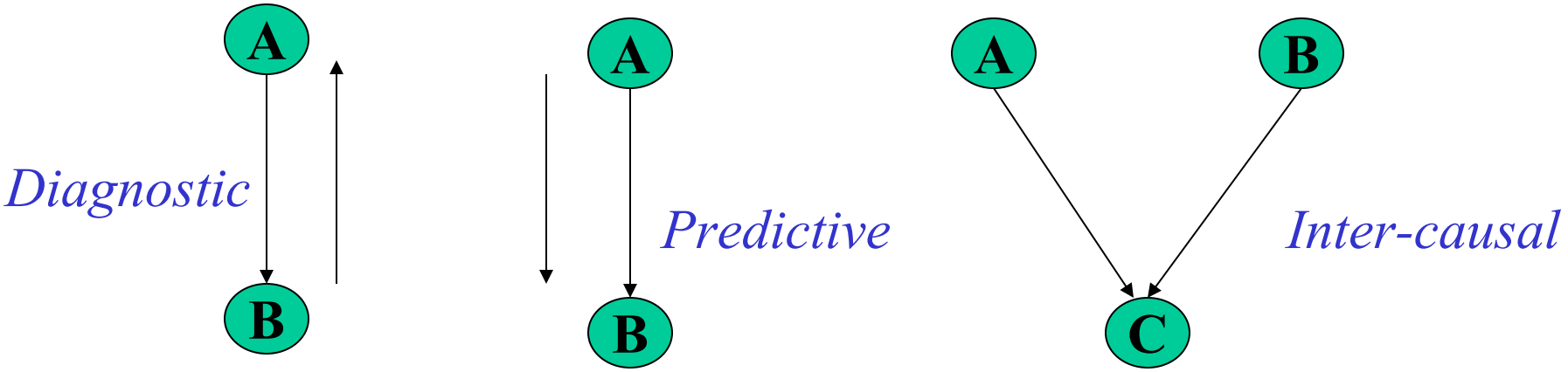
- e.g. P(grass wet | cloudy)

Inter-causal - suppose both A and B can cause C

if A "explains" C, it is evidence against B

e.g. P(S=1 | W=1, R=1) = 0.1945, i.e. lower chance of sprinkler being ON if one also knows that it rained!

("explaining away","Berkson's paradox", or "selection bias")



Summary: Network Properties and Usage

- Network structure is a modeling assumption
  - Exploit domain knowledge
  - Few edges means more independence among variables , so smaller CPTs

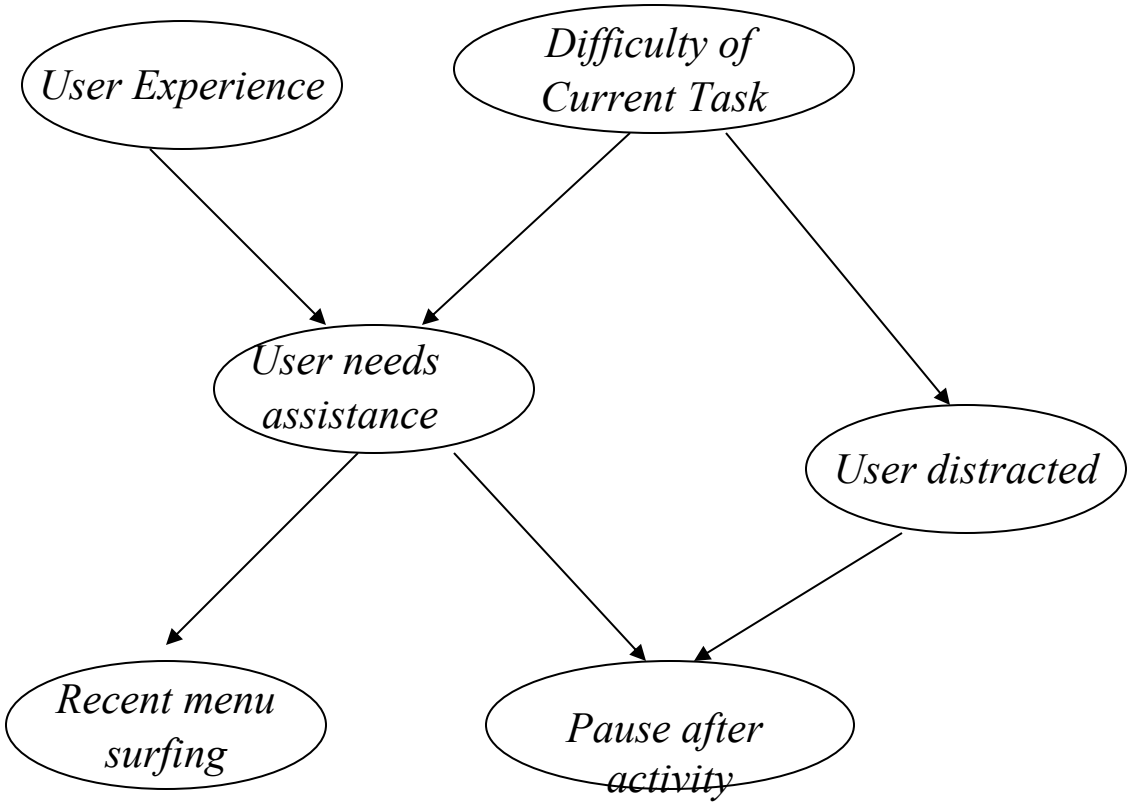
- If causality is known, make network from root causes to intermediate variables to end effects.
  - Helps to suggest "causal" explanations

- Usage: (very flexible) Inferencing
  - Infer the (probabilities of) values of one or more variables given observed values of some others.

Software: OpenBUGS http://mathstat.helsinki.fi/openbugs/
Python package
Infer.net http://research.microsoft.com/en-us/um/cambridge/projects/infernet/default.aspx
R packages such as bnlearn

Microsoft Office Assistant

- Only part of Bayesian network shown (Horvitz et al, Lumiere Project)



More Classification Methods

Directly getting to the Posterior: Logistic Regression, Neural Networks

Misc: SVMs

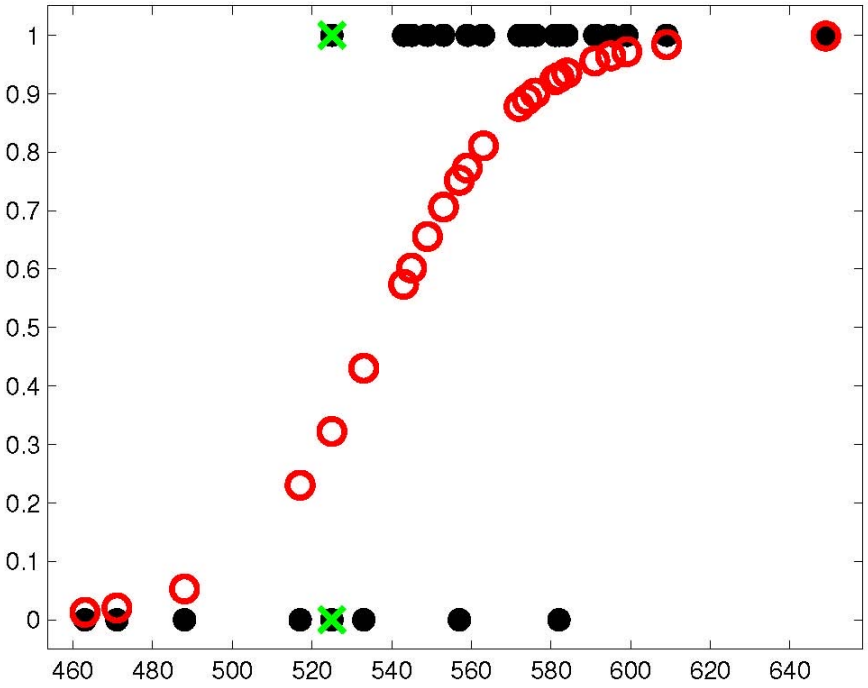
Logistic Regression

- Studied extensively and have well-developed theory (variable selection methods, model diagnostic checks, extensions for dealing with correlated data, etc).
- Retains many "simplicity" properties of linear regression
  - As well as its restrictions!

- (reading: JW 4.3; introductory video)

» Agenda:
  – Formulation
    - Binary classification
    - Multi-class
  – Learning: batch/online
  – Interpretation

Formulation



SAT data
From KM pg 259.

Formulation - II

Let y (**x**) = 0/1 Target variable for binary classification (C_0 vs. C_1)

- **Let** □ = E(y | **x**) = P(C_1|**x**)

Model: ln [□□□□□–□□□□□□□= $\beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k = \beta$□**x**□

i.e. model "log-odds ratio" (aka **logit**) as a linear function of predictors

– Equivalently: model P(C_1|**x**) = □□□□□□□□□□exp□–□β□**x**)) = σ□(β□**x**)
  - σ is called **the logistic function**, which is the inverse of logit

– Linear Class boundary (β□vector will be perpendicular to this boundary.)

Rewrite model as □ = exp(b. x)/(exp(b. x) + 1) and note that 1 = exp(0) to get a hint of how to generalize for more than 2 classes.

Multiclass Logistic Regression

- Extension to K classes: select a base class (say class K), compare each of the other K-1 with this base class
  - you need K-1 two-class logistic regression models
  - Set all coefficients for class K to 0 (to make the system **identifiable;** where K is the baseline class (choice of K is arbitrary).

  - Hard decision: select class j with highest $\boldsymbol{\beta_j}.\mathbf{x}$
  - Soft decision: get posteriors.
    - $P(C_i|\mathbf{x}) = \exp(\beta_{i0} + \beta_{i1}x_1 ..) / (1 + \Sigma_j \exp(\beta_{j0} + \beta_{j1}x_1 ..)$
    - $P(C_K|\mathbf{x}) = 1/ (1 + \Sigma_j \exp(\beta_{j0} + \beta_{j1}x_1 ..)$
      - Denominator is normalization term, ensuring probabilities sum to 1.

Training*

Minimize Negative Log-Likelihood (NLL) of a suitable probability model

Implied Stat Model: y's are i.i.d. Bernoulli

- $p(y|\mathbf{x}, \beta) = \text{Bernoulli}(y \mid \sigma(\beta . \mathbf{x}))$
- So $NLL(\beta) = -\ \Sigma_i [\ y_i \log \mu_i + (1-y_i) \log(1- \mu_i)\ ]$

(cross-entropy error function)

If we use $\tilde{y}_i \in \{-1,1\}$ *then*

$NLL(\beta) = \Sigma_i \log(1+ \exp(- \tilde{y}_i \beta . \mathbf{x}_i))$

Takeaway: a non-linear max-likelihood problem needs that to be solved iteratively.

The unknown parameters (β) are estimated by maximum likelihood. (gradient descent or iterative solutions, e.g. Newton-Raphson, or iterative reweighted least squares see KM 8.3)

SGD for Logistic Regression (0/1 Target Encoding)

(* KM Ch 10-10.2.4, somewhat advanced)

- Loss for *i*th data point (when target is encoded as 0/1)

$$= -[y_i \log \sigma(\boldsymbol{\beta} \cdot x_i + b) + (1 - y_i) \log(1 - \sigma(\boldsymbol{\beta} \cdot x_i + b))]$$

- So gradient is: $\frac{\partial L(\boldsymbol{\beta}, b)}{\partial \boldsymbol{\beta}_j} = [\sigma(\boldsymbol{\beta} \cdot x_i + b) - y] x_{ij}$

  –Where x_{ij} refers to the *j*th feature of x_i.

- The overall gradient: $\nabla L(f(x; \boldsymbol{\beta}), y)) = \begin{bmatrix} \frac{\partial}{\partial \boldsymbol{\beta}_1} L(f(x; \boldsymbol{\beta}), y) \\ \frac{\partial}{\partial \boldsymbol{\beta}_2} L(f(x; \boldsymbol{\beta}), y) \\ \vdots \\ \frac{\partial}{\partial \boldsymbol{\beta}_n} L(f(x; \boldsymbol{\beta}), y) \end{bmatrix}$

- SGD Update: $\boldsymbol{\beta}_{new} = \boldsymbol{\beta}_{old} - \eta \nabla L\left(f\left(x; \boldsymbol{\beta}_{old}\right), y\right)$

SGD for Logistic Regression (-1/1 Target Encoding)

- Loss for *i*th data point (when target is encoded as -1/1)

$$= \log\left(1 + e^{-y_i \boldsymbol{\beta}\cdot\mathbf{x}_i}\right)$$

- So gradient is: $\dfrac{\partial L(\boldsymbol{\beta}, b)}{\partial \boldsymbol{\beta}_j} = \left(\dfrac{-y_i \mathbf{x}_{ij}}{1 + e^{y_i \boldsymbol{\beta}\cdot\mathbf{x}_i}}\right)$

  –Where x_{ij} refers to the *j*th feature of x_i.

- The overall gradient: $\nabla L(f(x;\boldsymbol{\beta}), y)) = \begin{bmatrix} \frac{\partial}{\partial \boldsymbol{\beta}_1} L(f(x;\boldsymbol{\beta}), y) \\ \frac{\partial}{\partial \boldsymbol{\beta}_2} L(f(x;\boldsymbol{\beta}), y) \\ \vdots \\ \frac{\partial}{\partial \boldsymbol{\beta}_n} L(f(x;\boldsymbol{\beta}), y) \end{bmatrix}$

- SGD Update: $\boldsymbol{\beta}_{new} = \boldsymbol{\beta}_{old} - \eta \nabla L\left(f\left(x;\boldsymbol{\beta}_{old}\right), y\right)$

Interpretation

Logistic Regression is considered interpretable!

Model: ln [□□□□–□□□□□= $\beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k = \beta$□$\mathbf{x}$

- the effect of a unit change in x_i is to increase the odds of a response multiplicatively by the factor $\exp(\beta_i)$

– Model seems restrictive but quite robust for many applications, so long as effect of each feature is monotonic.
– Usually well calibrated

Visit to SAS

Multilayered Feedforward Neural Networks for Classification

- choose sufficiently large network (no. of hidden units)
- trained by "1 of M" desired output values
  - (“one-hot coding”)
- use validation set to determine when to stop training
- try several initializations, training regimens
- + powerful, nonlinear, flexible
- - interpretation? (Needs extra effort); slow ?

MLPs as Approximate Bayes Classifiers

- Output of "universal" Feedforward neural nets (MLP, RBF, deep nets) trained by "1 of M" desired output values, estimate Bayesian *a posteriori* probabilities if the cost function is

  mean square error OR cross-entropy

- Estimates become exact at the limit of infinite data, if one attains the global minimum.

Nowadays some packages use softmax at the output layer to force the outcomes to be interpreted as posterior class probabilities.

Softmax

» Monotonic transformation of a set of numbers: {x} → {t(x)}
  – All t(x) are non-negative
  – Sum to one
  – Potential for interpretation as discrete probabilities.

Exponentiate and normalize.

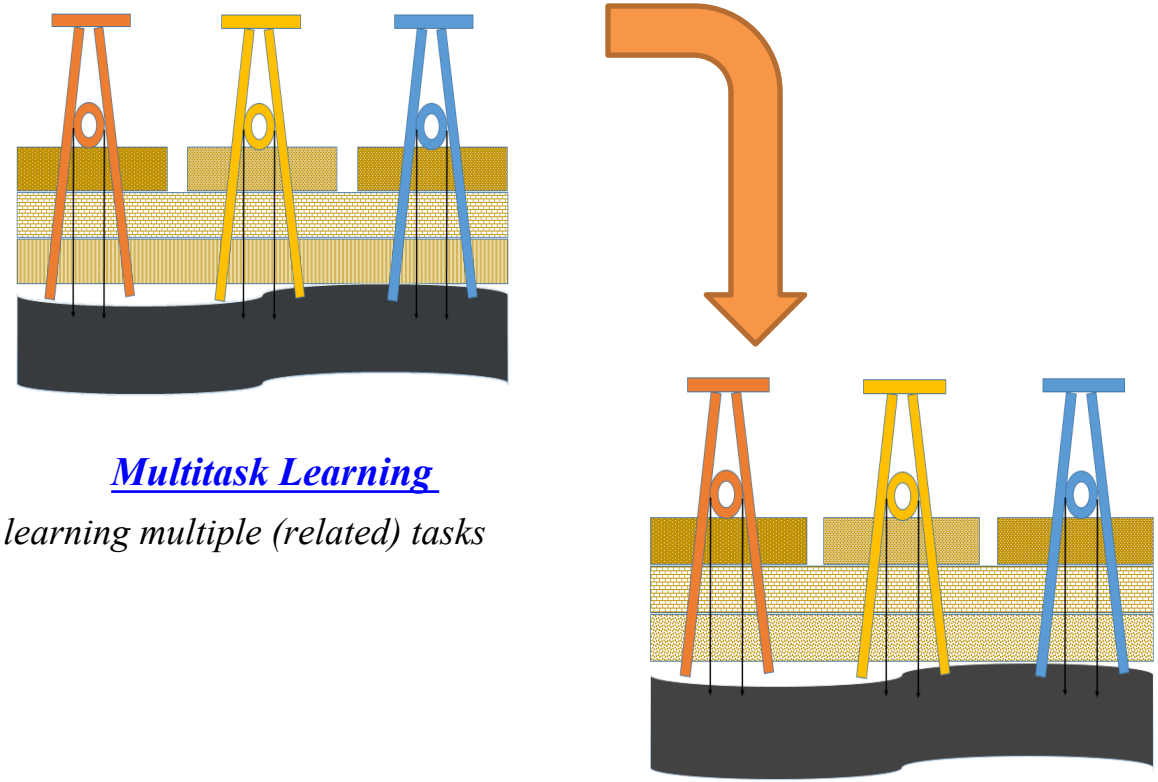
https://deepai.org/machine-learning-glossary-and-terms/softmax-layer#:~:text=The%20softmax%20function%20is%20a,can%20be%20interpreted%20as%20probabilities.

https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax

*Multi-Task & Transfer Learning**

Transfer Learning

make use of the knowledge gained while solving one problem and applying it to a different but related problem.



Multitask Learning

Simultaneously learning multiple (related) tasks

Code leaders

Active Learning: *incrementally recruiting labelled points based on analysis of model on existing training set. Usually evaluated using a "banana" plot.*

DL oriented blog on TL

- https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a
- **Motivation for Transfer Learning**
- **Understanding Transfer Learning**
- **Transfer Learning Strategies**
- **Transfer Learning for Deep Learning**
- **Deep Transfer Learning Strategies**
- **Types of Deep Transfer Learning**
- **Applications of Transfer Learning**
- **Case Study 1: Image Classification with a Data Availability Constraint**
- **Case Study 2: Multi-Class Fine-grained Image Classification with Large Number of Classes and Less Data Availability**
- **Transfer Learning Advantages**
- **Transfer Learning Challenges**
- **Conclusion & Future Scope**

Extras

Quiz (Bayes Rule)

- Suppose 0.01% of Austin's population have cancer. A new test for cancer shows positive 90% of the time when a person actually has cancer, and correctly indicates "negative" 95% of the time when run on someone who do not have cancer.

This test is conducted on an Austinite and the results come out positive.

- What is the probability that this person actually has cancer?

Covid example from KM Ch 2.3.1

2.3.1 Example: Testing for COVID-19

Suppose you think you may have contracted **COVID-19**, which is an infectious disease caused by the **SARS-CoV-2** virus. You decide to take a diagnostic test, and you want to use its result to determine if you are infected or not.

Let $H = 1$ be the event that you are infected, and $H = 0$ be the event you are not infected. Let $Y = 1$ if the test is positive, and $Y = 0$ if the test is negative. We want to compute $p(H = h|Y = y)$, for $h \in \{0, 1\}$, where y is the observed test outcome. (We will write the distribution of values, $[p(H = 0|Y = y), p(H = 1|Y = y)]$ as $p(H|y)$, for brevity.) We can think of this as a form of **binary classification**, where H is the unknown class label, and y is the feature vector.

First we must specify the likelihood. This quantity obviously depends on how reliable the test is. There are two key parameters. The **sensitivity** (aka **true positive rate**) is defined as $p(Y = 1|H = 1)$, i.e., the probability of a positive test given that the truth is positive. The **false negative rate** is defined as one minus the sensitivity. The **specificity** (aka **true negative rate**) is defined as $p(Y = 0|H = 0)$, i.e., the probability of a negative test given that the truth is negative. The **false positive rate** is defined as one minus the specificity. We summarize all these quantities in Table 2.1. (See Section 5.1.3.1 for more details.) Following https://nyti.ms/31MTZgV, we set the sensitivity to 87.5% and the specificity to 97.5%.

Next we must specify the prior. The quantity $p(H = 1)$ represents the **prevalence** of the disease in the area in which you live. We set this to $p(H = 1) = 0.1$ (i.e., 10%), which was the prevalence in New York City in Spring 2020. (This example was chosen to match the numbers in https://nyti.ms/31MTZgV.)

Now suppose you test positive. We have

$$p(H=1|Y=1) = \frac{p(Y=1|H=1)p(H=1)}{p(Y=1|H=1)p(H=1)+p(Y=1|H=0)p(H=0)} \tag{2.55}$$

$$= \frac{\text{TPR} \times \text{prior}}{\text{TPR} \times \text{prior} + \text{FPR} \times (1-\text{prior})} \tag{2.56}$$

$$= \frac{0.875 \times 0.1}{0.875 \times 0.1 + 0.025 \times 0.9} = 0.795 \tag{2.57}$$

So there is a 79.5% chance you are infected.

Now suppose you test negative. The probability you are infected is given by

$$p(H=1|Y=0) = \frac{p(Y=0|H=1)p(H=1)}{p(Y=0|H=1)p(H=1)+p(Y=0|H=0)p(H=0)} \tag{2.58}$$

$$= \frac{\text{FNR} \times \text{prior}}{\text{FNR} \times \text{prior} + \text{TNR} \times (1-\text{prior})} \tag{2.59}$$

$$= \frac{0.125 \times 0.1}{0.125 \times 0.1 + 0.975 \times 0.9} = 0.014 \tag{2.60}$$

So there is just a 1.4% chance you are infected.

Nowadays COVID-19 prevalence is much lower. Suppose we repeat these calculations using a base rate of 1%; now the posteriors reduce to 26% and 0.13% respectively.

The fact that you only have a 26% chance of being infected with COVID-19, even after a positive test, is very counter-intuitive. The reason is that a single positive test is more likely to be a false positive than due to the disease, since the disease is rare. To see this, suppose we have a population of 100,000 people, of whom 1000 are infected. Of those who are infected, $875 = 0.875 \times 1000$ test positive, and of those who are uninfected, $2475 = 0.025 \times 99,000$ test positive. Thus the total number of positives is $3350 = 875 + 2475$, so the posterior probability of being infected given a positive test is $875/3350 = 0.26$.

Of course, the above calculations assume we know the sensitivity and specificity of the test. See [GC20] for how to apply Bayes rule for diagnostic testing when there is uncertainty about these parameters.

Revisiting Bayes Decision Rule

- Let input **x** have d features or attributes $(x_1, x_2,\ldots,x_d)$;
  C is a random variable over class labels.

- **Bayes Decision rule:** Choose value of C that
  maximizes $P(x_1, x_2, \ldots, x_d|C)\ P(C)$

- Problem: how to estimate $P(x_1, x_2,\ldots,x_d\,|C)$ for each class?
  - Especially in high dimensions, interacting variables?

Naïve Bayes (KM Ch 9.3)

Naïve Bayes Classifier (KM Ch 9.3): Makes Conditional Independence assumption to simplify computation of likelihoods for each class.

1. Assume independence among attributes x_i when class is given: ("independence of attributes conditioned on class variable").
   - $P(x_1, x_2, \ldots, x_d | C_j) = P(x_1 | C_j)\, P(x_2 | C_j) \ldots P(x_d | C_j) = \Pi_i P(x_i | C_j)$
   - Note: Conditional independence not equal to attribute independence
2. Estimate probabilities directly from data

Conditional Independence:

- X is conditionally independent of Y given Z if $P(X|Y,Z) = P(X|Z)$
  - X, Y, Z could be sets of variables too

Estimating Probabilities from Data (Discrete Attributes)

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

(Example from TSK)

- Class: $P(C) = N_c/N$
  - e.g., P(No) = 7/10,
    P(Yes) = 3/10
- For discrete attributes:

  $P(x_i = v \mid C_k)$ = fraction of examples of class *k* for which attribute x_i takes value *v*.
  - Examples:

P(Status=Married|No) = 4/7
P(Refund=Yes|Yes)=0

Naïve Bayes Example

Step 1: Estimating Univariate Probabilities for each variable-class combination

Discrete Probabilities: (Binary/Categorical)

See tax evasion example in main slides.

Continuous variables:

- Discretize the range into bins
  - one ordinal attribute per bin
  - violates independence assumption
- Binarize: (may lose substantial info)
- Probability density estimation:
  - (usually assuming Normal distribution)

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Normal distribution:

$$P(x_i \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(x_i-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - One for each (x_i, c_i) pair

- For (Income, Class=No):
  - If Class=No
    - sample mean = 110
    - sample variance = 2975

$$P(Income = 120 \mid No) = \frac{1}{\sqrt{2\pi}\,(54.54)} e^{-\frac{(120-110)^2}{2(2975)}} = 0.0072$$

Example of Naïve Bayes Classifier

Given a Test Record:

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=No|Class=No)
  × P(Married| Class=No)
  × P(Income=120K| Class=No)
  = 4/7 × 4/7 × 0.0072 = 0.0024

- P(X|Class=Yes) = P(Refund=No| Class=Yes)
  × P(Married| Class=Yes)
  × P(Income=120K| Class=Yes)
  = 1 × 0 × 1.2 × 10^{-9} = 0

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)

=> Class = No

Smoothing Naïve Bayes

- Avoids zero probablity due to one attribute-value/class combo being absent in training data.
  - Zeroes entire product term
- Probability estimation:

$$\text{Original}: P(x_i \mid C) \approx \frac{N_{ic}}{N_c}$$

$$\text{m-estimate}: P(x_i \mid C) \approx \frac{N_{ic} + mp_i}{N_c + m}$$

c: number of classes

p: prior probability

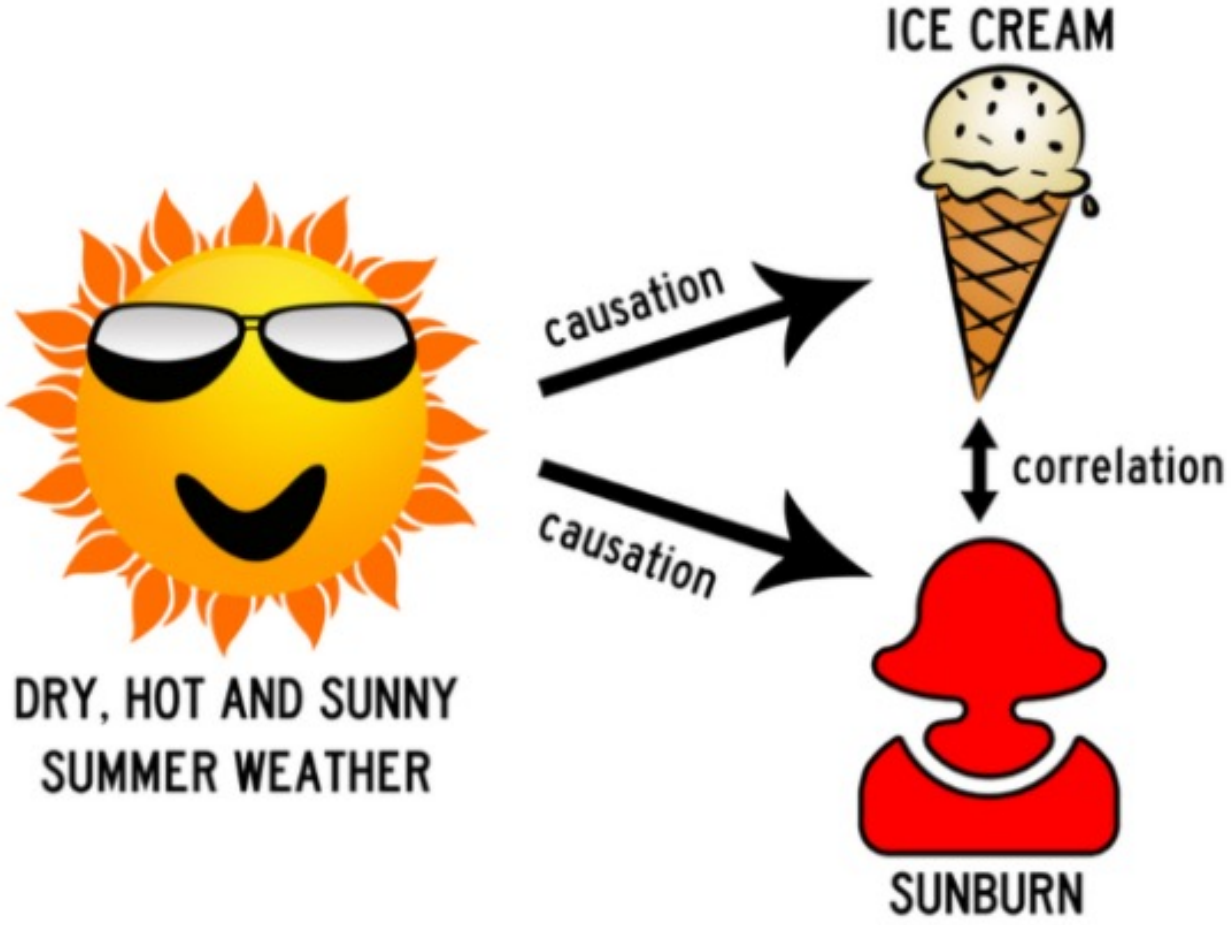
m: weight of prior (i.e. # of virtual samples)

e.g. in text analysis, add a "virtual document that has one instance of every word in the vocabulary (Laplace smoothing): $(N_{ic} + 1) / (Nc + |vocab|)$

Naïve Bayes (Comments)

- Conditional Independence assumption often does not hold
  - Poor calibration: estimate of P(C|x), often unrealistically close to 0 or 1
    - but still may pick the most likely class correctly.
- Somewhat robust to isolated noise points, and irrelevant attributes
- Tries to finesse "curse of dimensionality"
- Most popular with binary or small cardinality categorical attributes
- **Requires only single scan of data; also streaming version is trivial.**

- Notable "Success": Text (bag-of-words representation + multinomial model per class) before advent of data-intensive embedding/transformer/… techniques


ICE CREAM
causation
correlation
causation
DRY, HOT AND SUNNY
SUMMER WEATHER
SUNBURN

GLMs and Interpretation

- Generalized linear model (GLM): $\mathbf{l}(y) = \beta \square \mathbf{x}$
  - $\mathbf{l}(y)$ is the canonical link function being used.

- Hence Logistic regression is a GLM with logit(.) as the canonical link function

Logistic Regression is considered interpretable!

- the effect of a unit change in x_i is to increase the odds of a response multiplicatively by the factor $\exp(\beta_i)$

– Model seems restrictive but quite robust for many applications, so long as effect of each feature is monotonic.
– Usually well calibrated