Lecture 16: Parametric ML (perceptron, logistic regression, naive bayes)

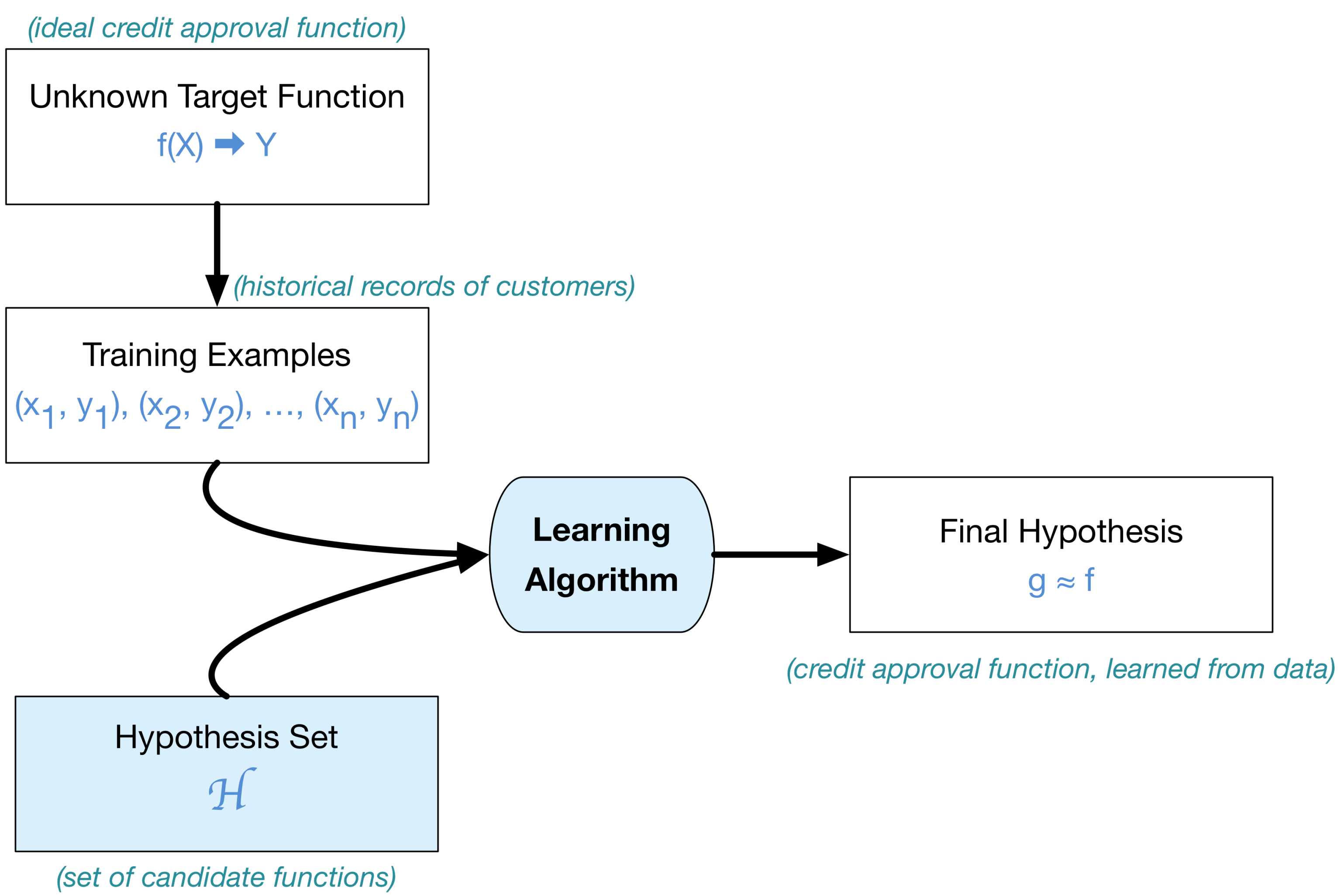
COSC 480 Data Science, Spring 2017

Michael Hay

Recall from last time

A machine learning solution has two components

- Hypothesis set
- Learning algorithm



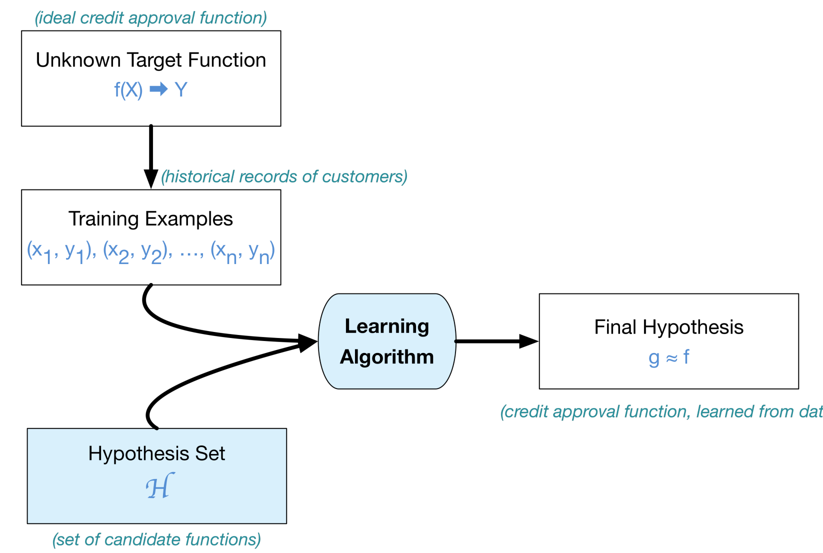
Credit: Yaser Abu Mostafa https://work.caltech.edu/

Recall from last time

- Perceptron: outputs a *decision:* +1 (spam) or -1 (ham) based on h(x)

$$h(x) = \text{sign}\left(\sum_{i=0}^{d} w_i x_i\right)$$

- Components of perceptron solution
  - $\mathcal{H}$ = all possible settings of weights $w_0, w_1, \ldots w_d$.
  - Learning algorithm: PerceptronLearner (described last time)



Today

- Perceptron: outputs a *decision:* +1 (spam) or -1 (ham)

- Today: two techniques that output *probability*
  — i.e., a number in [0,1] reflecting confidence
  (0.95 likely to be spam)

- Why output a *probability*?

  - Reflect inherent uncertainty. E.g., predict heart attack within year given cholesterol, BP, age, etc.

  - Tune your solution to application specific costs (e.g., spam false positives vs. false negatives)

Today

- We look at two techniques:
  - Naive Bayes
  - Logistic Regression

$$h(x) = \text{sign}\left(\sum_{i=0}^{d} w_i x_i\right)$$

- Spoiler alert:
  - They are linear models (like perceptron, linear regression)
  - **Difference #1**: outputs a probability rather than classification decision
  - **Similarity with perceptron**: when viewed as a classifier (if p > 0.5, output +1 else -1), they have the same form as perceptron — i.e., can be written like h(x)
  - **Difference #2**: the *learning algorithm* to find $w_1, \ldots, w_d$

Bayesian classification

- Recall Bayes' rule $$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$
- Let's design a classifier around Bayes' rule:

$$P(Y=+1|X) = P(X|Y=+1)P(Y=+1)/P(X)$$

$$P(Y=-1|X) = P(X|Y=-1)P(Y=-1)/P(X)$$

- Same **denominator**, so assign to whichever Y value has largest **numerator**
  - Notation: $$P(Y=y|X) \propto P(X|Y=y)P(Y=y)$$

"proportional to"

Credit: slide adapted with permission from Ashwin Machanavajjhalla and Jun Yang, Duke University

Example: classify 

- Y is +1 (spam) or -1 (ham)
- X is a vector of features

here there be choices!
(feature engineering)

- Let's assume X is the following:
  - Fix a vocabulary of *d* words (e.g. every "word" we see in our training data).
  - Let V define the vocab: V = [$word_1$, $word_2$, …, $word_d$]
  - X = (X_1, X_2, …, X_d) where X_i = 1 if $word_i$ occurs in email and 0 otherwise
- So, informally, X = "words in email"

Bayes' classifier for 

- Bayes' classifier

$$P(Y = y|X) \propto P(X|Y = y)P(Y = y)$$

- Bayes' classifier for spam/ham

$$P(Y = \text{spam}|\text{words in email}) \propto P(\text{words in email}|Y = \text{spam})P(Y = \text{spam})$$

- **Aside**: if we *knew* P(X=x|Y=y) and P(Y=y) this is the *optimal* classifier (i.e., highest accuracy)

- But these are unknown. Can we estimate from data?
  - Too many combos of possible values!
    If *d* words, then 2^d possible assignments of X
  - Most won't even appear in training data

Naive Bayes' assumption

- Given class, features are **independent**.

$$P(X = x|Y = y) = \prod_{i=1}^{d} P(X_i = x_i|Y = y)$$

- Applied to spam

$$P(\text{words in email}|Y = \text{spam}) = \prod_{i=1}^{d} P(\text{word}_i \text{ occurs in email}|Y = \text{spam})$$

- Why naive?

- What does assumption get us? We can now *estimate* P(X=x|Y=y) using *training data* (details in a bit).

Naive Bayes' for 

- Naive Bayes':

$$P(Y=y|X=x) \propto \prod_{i=1}^{d} P(X_i = x_i | Y = y) P(Y=y)$$

- Applied to Spam
  - Compute this $\prod_{i=1}^{d} P(\text{word}_i \text{ occurs in email} | Y = \text{spam}) P(Y = \text{spam})$
  - and this $\prod_{i=1}^{d} P(\text{word}_i \text{ occurs in email} | Y = \text{ham}) P(Y = \text{ham})$
  - and classify based on whichever is larger

Parameters of Naive Bayes

	$P(X_i=1 \mid \text{spam})$	$P(X_i=0 \mid \text{spam})$
X_1=viagra occurs	0.3	0.7
X_2=you occurs	0.6	0.4
X_3=meeting occurs	0.05	0.95

Y	**P(Y = y)**
spam	0.33
ham	0.66

	$P(X_i=1 \mid \text{ham})$	$P(X_i=0 \mid \text{ham})$
X_1=viagra occurs	0.01	0.99
X_2=you occurs	0.65	0.35
X_3=meeting occurs	0.25	0.75

Example on board:
How classify this email?
"John, I hope you are not late for the meeting!"

Exercise

Instructions: ~1 minute to think/ answer on your own; then discuss with neighbors; then I will call on one of you

	$P(X_i=1 \mid \text{spam})$	$P(X_i=0 \mid \text{spam})$
X_1=viagra occurs	0.3	0.7
X_2=you occurs	0.6	0.4
X_3=meeting occurs	0.05	0.95

Y	**P(Y = y)**
spam	0.33
ham	0.66

	$P(X_i=1 \mid \text{ham})$	$P(X_i=0 \mid \text{ham})$
X_1=viagra occurs	0.01	0.99
X_2=you occurs	0.65	0.35
X_3=meeting occurs	0.25	0.75

Write out expressions to compute the probability of spam for this email: *"Cheap Viagra… special deal available only for you!"*

Estimating parameters

	P(X_i=1|ham)	**P(X_i=0|ham)**
X_1=viagra occurs	0.01	0.99
X_2=you occurs	0.65	0.35
X_3=meeting occurs	0.25	0.75

no. ham emails with viagra in them / no. of ham emails

Disclaimer: there are many versions of Naïve Bayes with different assumptions and calculations. This is the Bernoulli NB model. Multinomial NB is often used for text data. http://scikit-learn.org/stable/modules/naive_bayes.html

Credit: slide adapted with permission from Ashwin Machanavajjhalla and Jun Yang, Duke University

Exercise

Instructions: ~1 minute to think/ answer on your own; then discuss with neighbors; then I will call on one of you

	$P(X_i=1\|spam)$	$P(X_i=0\|spam)$
X_1=viagra occurs	0.3	0.7
X_2=you occurs	0.6	0.4
X_3=meeting occurs	**0.00**	**1.00**

	$P(X_i=1\|ham)$	$P(X_i=0\|ham)$
X_1=viagra occurs	0.01	0.99
X_2=you occurs	0.65	0.35
X_3=meeting occurs	0.25	0.75

Y	**P(Y = y)**
spam	0.33
ham	0.66

Suppose estimates were same as before **except for the change in highlighted in green**. Now how would this email be classified? *"Cheap Viagra… special deal available only for you! Come to exclusive meeting to find out more!"*

Estimating parameters

	P(X_i=1|ham)	**P(X_i=0|ham)**
X_1=viagra occurs	0.01	0.99
X_2=you occurs	0.65	0.35
X_3=meeting occurs	0.25	0.75

no. ham emails with viagra in them / no. of ham emails

- Problem: what if a word *i* never occurs in a ham email?
  - Then $P(X_i = 1 \mid \text{ham}) = 0$… and $P(\text{ham} \mid X) = 0$!
  - Example of ***overfitting***. Zero occurrences in training data ≠ event is impossible!

Credit: slide adapted with permission from Ashwin Machanavajjhalla and Jun Yang, Duke University

Smoothing

- Again, a Bayesian idea
- We always have some prior expectation
  - E.g., coin flips are fair
- Given little evidence, we lean toward prior
  - E.g., 8 heads in 10 flips; still fair?
- Given lots of evidence, we lean toward data
  - E.g., 8,000 heads in 10,000 flips; still fair?

Credit: slide adapted with permission from Ashwin Machanavajjhalla and Jun Yang, Duke University

Laplace smoothing

- Pseudo training examples: pretend we saw *k* additional training examples for each combination of feature value (X_i=1,X_i=0) and class label (Y=+1, Y=-1)

- Spam example:
  - For each word (Viagra), we saw 2*k* additional hams, *k* had Viagra and *k* didn't
  - (Same goes for spam)

(no. ham emails with v1 in them + k) / (no. of ham + 2k)

Before smoothing

	P(X_i=1| ham)	P(X_i=0| ham)
…	…	…
X_i=viagra occurs	0/1000	1000/1000
…	…	…

After smoothing (k=1)

	P(X_i=1| ham)	P(X_i=0| ham)
…	…	…
X_i=viagra occurs	1/1002	1001/1002
…	…	…

Recap

- Train: given training dataset of (x,y) pairs
  - Estimate $P(X_i = x_i \mid Y = y)$ by counting + smoothing
  - Estimate P(Y=y) by simple counting (+ smoothing)
- Predict: given new x,
  - Compute

$$p = \prod_{i=1}^{d} P(X_i = x_i | Y = +1) P(Y = +1)$$

and

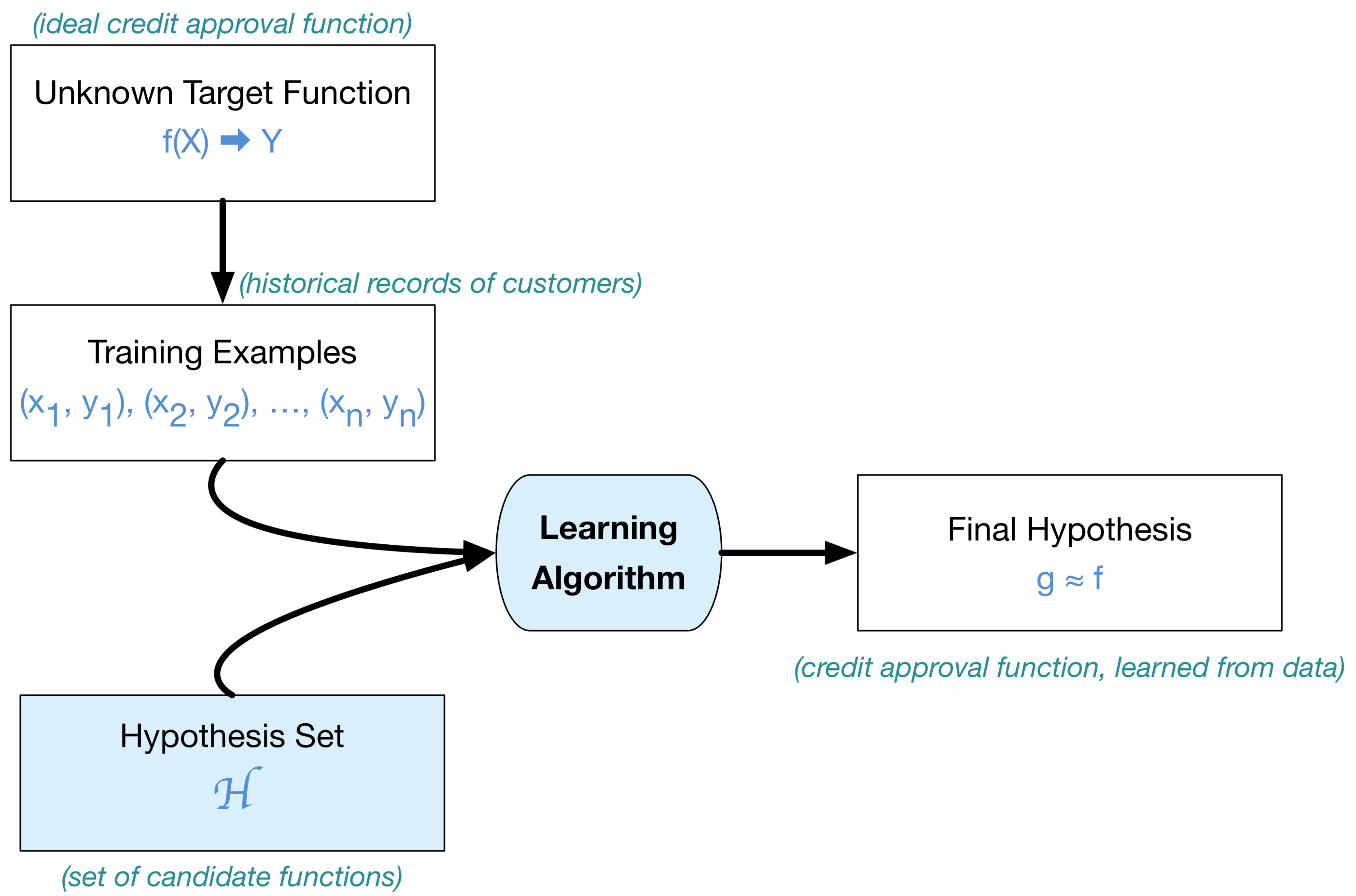
$$q = \prod_{i=1}^{d} P(X_i = x_i | Y = -1) P(Y = -1)$$

  - To get probability
    P(Y=+1 | X), *normalize — p / (p + q)*
  - To get prediction, choose whichever is *larger*

Recall from last time

The solution has two components

- Hypothesis set
- Learning algorithm



- **Naive Bayes hypothesis set**: what is it?

Credit: Yaser Abu Mostafa https://work.caltech.edu/

Naive Bayes vs. Perceptron

- For *binary* features*, it turns out, the NB classifier can be written like this: $$h(x) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + w_0\right)$$
- Where w_i for i=1..d, depends on the "log odds" terms $\log\left(\frac{P(X_i=0|Y=+1)}{P(X_i=0|Y=-1)}\right)$ and $\log\left(\frac{P(X_i=1|Y=+1)}{P(X_i=1|Y=-1)}\right)$
- And w_0 depends on $\log\left(\frac{P(Y=+1)}{P(Y=-1)}\right)$
- In other words, NB is an alternative ***learning algorithm*** for the perceptron *hypothesis set*

* For non-binary features, it's still a perceptron but for a different feature space (details omitted)

NB is a linear classifier: math details

$$1 < \frac{P(Y=+1|X=x)}{P(Y=-1|X=x)} \qquad \text{(we predict +1 when this holds)}$$

$$= \prod_{i=1}^{d} \frac{P(X_i = x_i|Y=+1)}{P(X_i = x_i|Y=-1)} \frac{P(Y=+1)}{P(Y=-1)} \qquad \text{(Naive Bayes classification)}$$

$$\log 1 < \log \left(\prod_{i=1}^{d} \frac{P(X_i = x_i|Y=+1)}{P(X_i = x_i|Y=-1)} \frac{P(Y=+1)}{P(Y=-1)} \right) \qquad \text{(take log of both sides)}$$

$$0 < \sum_{i=1}^{d} \log \left(\frac{P(X_i = x_i|Y=+1)}{P(X_i = x_i|Y=-1)} \right) + \log \left(\frac{P(Y=+1)}{P(Y=-1)} \right) \qquad \text{(distribute log, product becomes a sum)}$$

So, Naive Bayes makes predictions according to h(x):

$$h(x) = \operatorname{sign} \left(\sum_{i=1}^{d} \log \left(\frac{P(X_i = x_i|Y=+1)}{P(X_i = x_i|Y=-1)} \right) + \log \left(\frac{P(Y=+1)}{P(Y=-1)} \right) \right)$$

NB is a linear classifier: more math details

- Naive Bayes predicts Y according to h(x)

$$h(x) = \text{sign}\left(\sum_{i=1}^{d} \log\left(\frac{P(X_i = x_i|Y=+1)}{P(X_i = x_i|Y=-1)}\right) + \log\left(\frac{P(Y=+1)}{P(Y=-1)}\right)\right)$$

- Let's simplify *h(x)*

$$\log\left(\frac{P(X_i = x_i|Y=+1)}{P(X_i = x_i|Y=-1)}\right) = \begin{cases} \log\left(\frac{P(X_i=1|Y=+1)}{P(X_i=1|Y=-1)}\right) & \text{if } x_i = 1 \\ \log\left(\frac{P(X_i=0|Y=+1)}{P(X_i=0|Y=-1)}\right) & \text{if } x_i = 0 \end{cases}$$

call this p_i

call this q_i

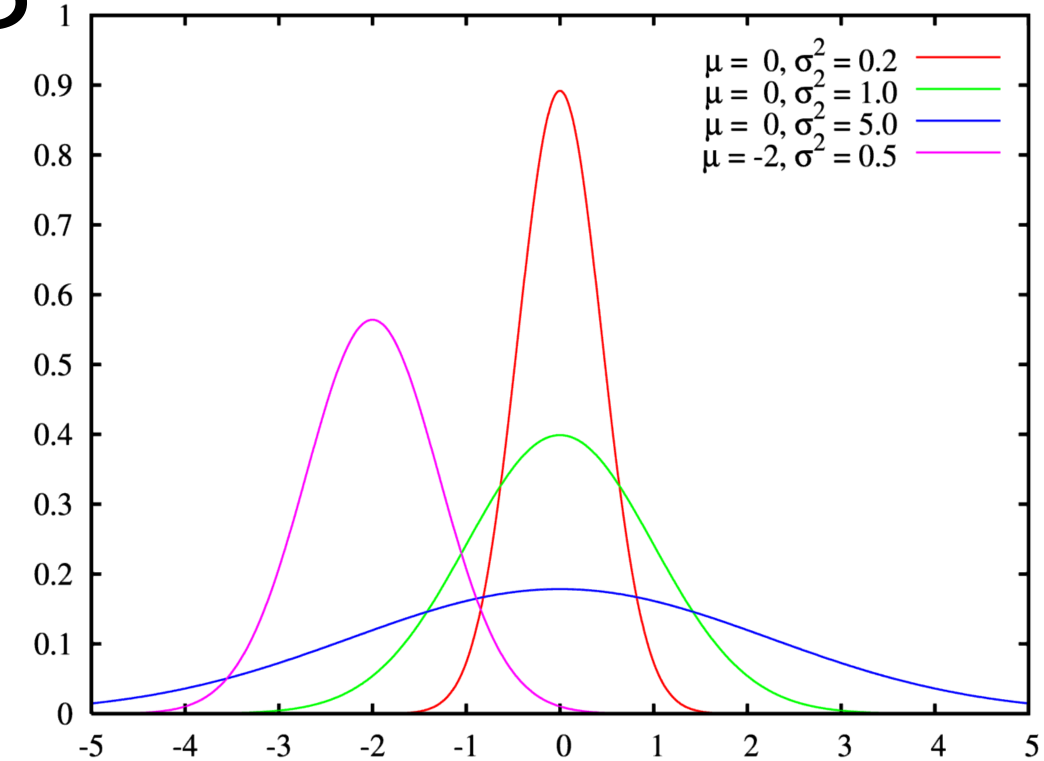
$$= p_i x_i + q_i(1 - x_i)$$

$$= (p_i - q_i)x_i + q_i$$

- We can write *h(x)* in terms of weights $w_0, \ldots, w_d$.

$$h(x) = \text{sign}\left(\sum_{i=1}^{d} \underbrace{(p_i - q_i)}_{w_i} x_i + \underbrace{\sum_{i=1}^{d} q_i + \log\left(\frac{P(Y=+1)}{P(Y=-1)}\right)}_{w_0}\right)$$

What about non-binary features?



Numerical: Assume P($X_i|Y$) is *Gaussian* (i.e., bell curve)

- For each class, think of the collection of X_i values from training examples in this class as a sample from P($X_i|Y$)
- Compute sample mean and variance as estimates for the Gaussian distribution parameters

Categorical: with *k* values, use *multinomial* distribution

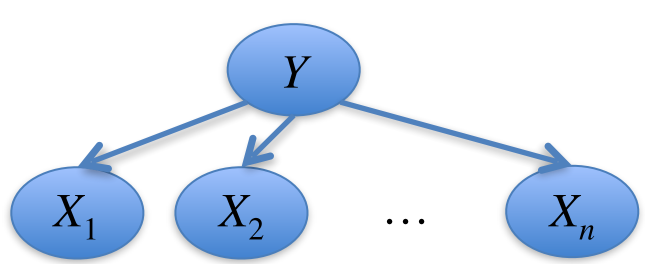
Credit: slide adapted with permission from Ashwin Machanavajjhalla and Jun Yang, Duke University
http://commons.wikimedia.org/wiki/File:Normal_distribution_pdf.png

Bigger picture

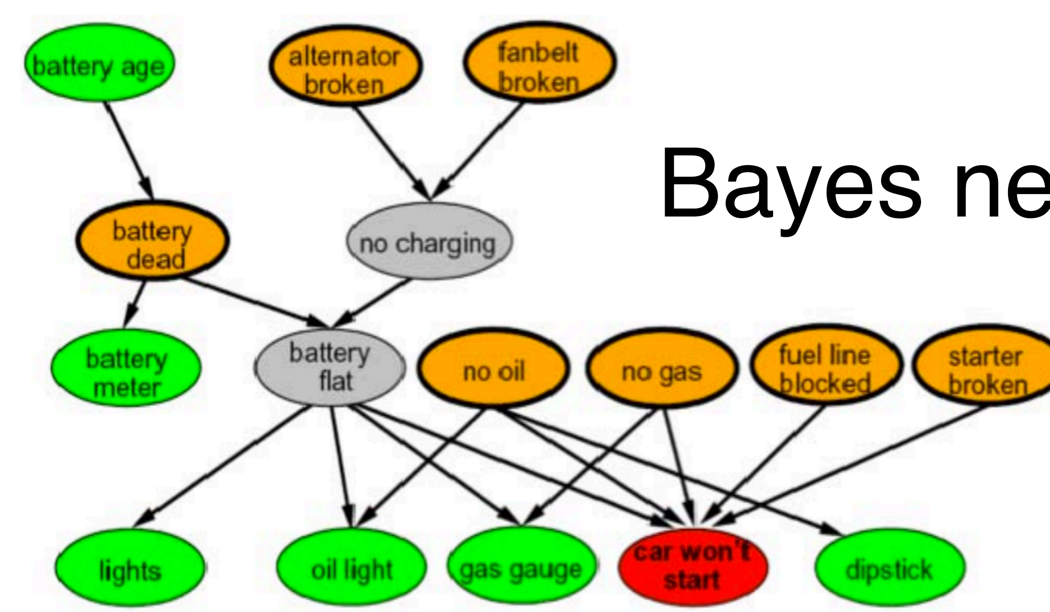
- When does Naive Bayes shine?
    - When you have *lots* of features and not much data — learning algorithm unlikely to overfit
    - When you have *tons* of data with *lots* of features — it's simple, fast, and empirically does pretty well

Predictive Modeling with Big Data: Is Bigger Really Better?
http://online.liebertpub.com/doi/pdf/10.1089/big.2013.0037

- Beyond Naive: Bayesian networks allow you to encode more complex dependencies



Naïve Bayes is just one special case



Bayes net

Exercise

Instructions: ~1 minute to think/answer on your own; then discuss with neighbors; then I will call on one of you

Suppose you used a perceptron to classify email as spam or ham. We will interpret +1 to mean that the email is spam and -1 to mean ham.

Attributes are word occurrences. For example, x_{Viagra} is 1 if the word "Viagra" appears in the email and 0 if not. We have a feature for every word in language (!).

- Q: Can you think up some words that should receive a positive weight? Negative weight?

- Q: How would an empty email (no words) be classified by the perceptron?