Prediction

Everything Data
CompSci 216 Spring 2015

DUKE
COMPUTER SCIENCE

Parts of this lecture are inspired by contents from
Dan Weld's CSE 473 at U. Washington
(http://courses.cs.washington.edu/courses/cse473/12sp/slides/27-learn-em.pdf),
and Daniel B. Neil's 95-796 at CMU
(http://www.cs.cmu.edu/~neill/courses/95796-module3.pdf)

Announcements (Wed. Feb. 18)

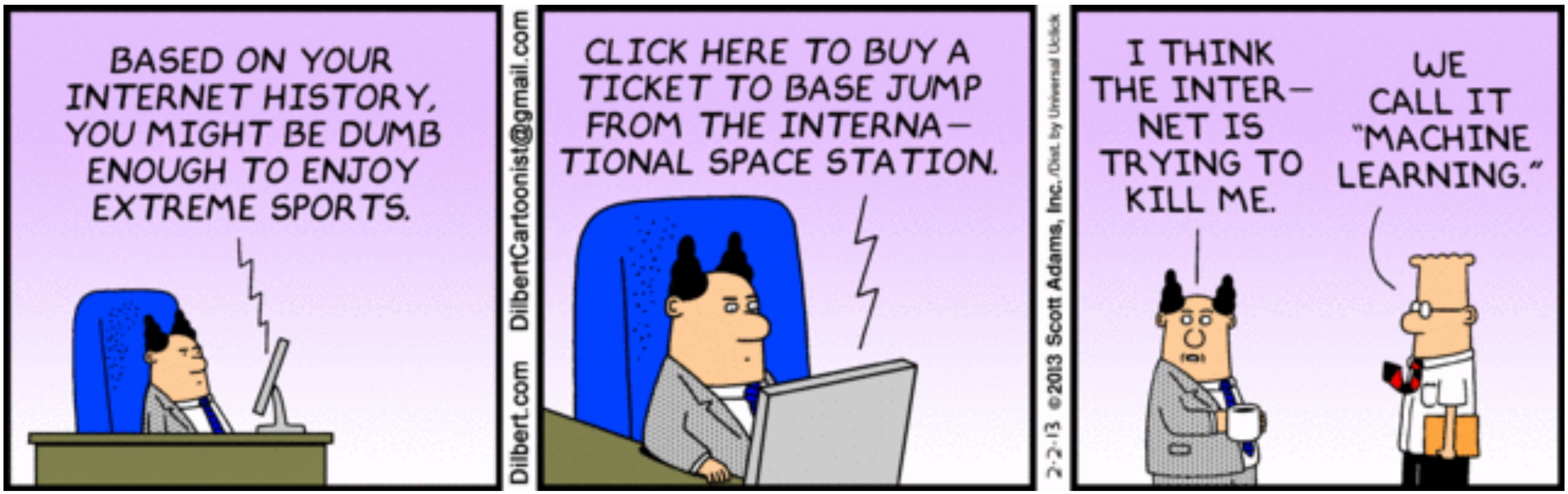
- **Homework #6** to be posted by tomorrow morning
  - Due midnight Sunday
- **Project description** posted on website
  - Team formation due next Monday

The prediction problem

X *Y*

m1	…	m1682	o1	…	o21	gender
0	…	0	1	…	0	M
1	…	1	0	…	0	F
1	…	1	0	…	0	M
..	…	…	…	…	…	…
1	…	0	1	…	0	???
…	…	…	…	…	…	???

- Data = a table of records
  - All values for columns in *X* are known
  - For some rows (*training data*), *Y* value is known
  - For others, *Y* is unknown
- ***Prediction***: guess a missing *Y* value
  - You can look at the given *X* values in the same row, as well as all training data
  - Categorical Y: ***classification***
  - Numerical Y: ***regression***


BASED ON YOUR INTERNET HISTORY, YOU MIGHT BE DUMB ENOUGH TO ENJOY EXTREME SPORTS.
Dilbert.com DilbertCartoonist@gmail.com
CLICK HERE TO BUY A TICKET TO BASE JUMP FROM THE INTERNA– TIONAL SPACE STATION.
2-2-13 ©2013 Scott Adams, Inc./Dist. by Universal Uclick
I THINK THE INTER– NET IS TRYING TO KILL ME.
WE CALL IT "MACHINE LEARNING."

Spam filtering

- Data: collection of emails, hand-labeled spam or ham

Dear Sir:

spam

First, I must solicit your confidence in this transaction, this is by virture of its nature as being utterly confidencial and top secret. …

TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

spam

99 MILLION EMAIL ADDRESSES FOR ONLY $99

Ok, Iknow this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

ham

- Problem: given a new email, tell whether it's spam or ham

Background image: http://techcrunch.com/2013/08/03/a-year-of-spam-twoo/

Bayesian classification

Recall Bayes' Rule: $\mathrm{P}(Y \mid X) = \dfrac{\mathrm{P}(X \mid Y) \cdot \mathrm{P}(Y)}{\mathrm{P}(X)}$

- Given $X = x$, what's Y?

$\mathrm{P}(Y = y_0 \mid X = x) = \mathrm{P}(X = x \mid Y = y_0) \cdot \mathrm{P}(Y = y_0) \,/\, \mathrm{P}(X = x)$

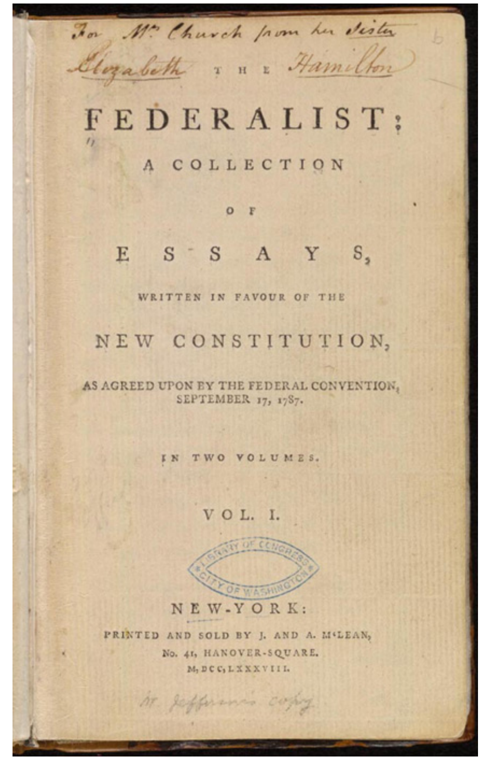
$\mathrm{P}(Y = y_1 \mid X = x) = \mathrm{P}(X = x \mid Y = y_1) \cdot \mathrm{P}(Y = y_1) \,/\, \mathrm{P}(X = x)$

$\mathrm{P}(Y = y_2 \mid X = x) = \mathrm{P}(X = x \mid Y = y_2) \cdot \mathrm{P}(Y = y_2) \,/\, \mathrm{P}(X = x)$

…

  – Same denominator, so we predict $Y = y_i$ with the largest numerator
    - So oftentimes you see: $\mathrm{P}(Y \mid X) \propto \mathrm{P}(X \mid Y) \cdot \mathrm{P}(Y)$

The Federalist Papers



- Anonymous essays written by *Alexander Hamilton*, *James Madison*, and *John Jay*
  - Authorships of 73 essays were certain; 12 were in dispute
- Mosteller and Wallace solved the mystery using Bayesian methods
  - *Inference and Disputed Authorship: The Federalist*. Addison-Wesley, 1964.

http://en.wikipedia.org/wiki/Federalist_Papers

Modeling—what's our X?

- A document = a set of words
- Feature = set contains a particular word

$$P(\text{spam} \mid \textit{cheap, viagra, discount, delivery}, \ldots)$$
$$\propto P(\textit{cheap, viagra, discount, delivery}, \ldots \mid \text{spam})\, P(\text{spam})$$

Same goes for ham

- Learn $P(X \mid Y)$ directly?
  - Too many features; too many combinations of possible values
  - Most won't even appear in training data

Naïve Bayes assumption

Given class, features are independent:

$$P(X_1 X_2 \cdots X_n \mid Y) = \prod_{i=1}^{n} P(X_i \mid Y)$$

– A pragmatic and often effective way to simplify a massive joint distribution

So for spam filtering:

P(spam | *cheap, viagra, discount, delivery, …*)
∝ P(*cheap* | spam) P(*viagra* | spam) P(*discount* | spam) P(*delivery* | spam)…
P(spam)

Training Naïve Bayes

Compute from training data:

P(ham) vs. P(spam)

ham : 0.66
spam: 0.33

P(*word* | ham)

...
to : 0.0153
...
nation : 0.0002
morally : 0.0001
...

P(*word* | spam)

...
to : 0.0133
...
viagra : 0.0002
delivery : 0.0002
...

Applying Naïve Bayes

P(spam | *viagra, delivery, to, …*) ∝
P(*viagra* | spam) P(*delivery* | spam) P(*to* | spam)…P(spam)
P(ham | *viagra, delivery, to, …*) ∝
P(*viagra* | ham) P(*delivery* | ham) P(*to* | ham)…P(ham)

- P(*to* | spam) ≈ P(*to* | ham), so this feature isn't very useful
- P(*viagra* | ham) is extremely low (say 0), so spam wins

P(spam | *flower, delivery, …*) ∝ P(*flower* | spam) P(*delivery* | spam)…P(spam)
P(ham | *flower, delivery, …*) ∝ P(*flower* | ham) P(*delivery* | ham)…P(ham)

- Say *delivery* wasn't seen in any ham during training, so P(*delivery* | ham) = 0; spam again!

What went wrong?

- An example of ***overfitting***

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?

Laplace smoothing

For each word, pretend we additionally saw a ham email with that word

$$P(word \mid \text{ham}) = \frac{C_{\text{ham}}^{word} + 1}{\sum_w C_{\text{ham}}^{w} + (\text{\# words in vocabulary})}$$

- C_{ham}^{word} counts the # times we observe ham emails generating the given *word*
- $\sum_w C_{\text{ham}}^{w}$ is the total # of times we observe ham emails generating any word

Same goes for spam

Smoothed probabilities

Example:

- 5,000 words
- 1,000 spams (two contain *delivery*)
- 1,000 hams (none contains *delivery*)
- Each email contains 10 distinct words

Before smoothing:

P(*delivery* | spam) = 2 / (1,000 × 10) = 0.0002

P(*delivery* | ham) = 0

After smoothing:

P(*delivery* | spam) = (2+1) / (1,000 × 10 + 5,000) = 0.0002

P(*delivery* | ham) = 1 / (1,000 × 10 + 5,000) = 0.00007

Recap of Naïve Bayes

- Train:
  - P(Y): for each class, calculate

    (# docs in class) / (total # docs)

  - P(X_i | Y): for each class & word, calculate

$$\frac{(\text{\# docs in class with this word}) + 1}{(\text{total ``length'' of docs in class}) + (\text{\# possible words})}$$

- Predict: given doc containing words x_1, x_2, …, x_n, bet on the class y with the largest

$$P(x_1 | y)\ P(x_2 | y)\ \ldots\ P(x_n | y)\ P(y)$$

Disclaimer: there are many versions of Naïve Bayes with different assumptions and calculations;
see V. Metsis, I. Androutsopoulos and G. Paliouras. Spam filtering with Naive Bayes – Which Naive Bayes? *CEAS 2006.*
This one is multinomial with boolean attributes.

One last detail

$P(x_1 | y)\, P(x_2 | y) \ldots P(x_n | y)\, P(y)$

- These are pretty small numbers!
- Arithmetic underflow may occur

Solution?

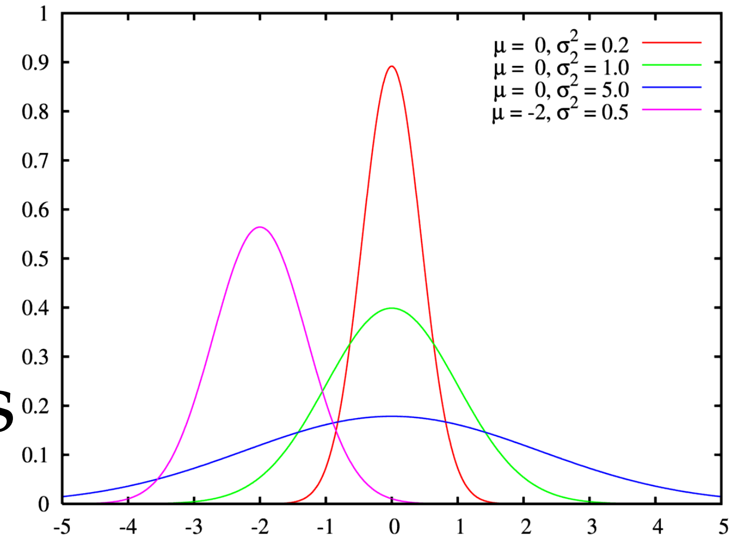
- Compute instead
  $\ln(\, P(x_1 | y)\, P(x_2 | y) \ldots P(x_n | y)\, P(y)\,)$

$= \ln P(x_1 | y) + \ldots + \ln P(x_n | y) + \ln P(y)$

- Just keep everything in the ln form!

What about numerical features?

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

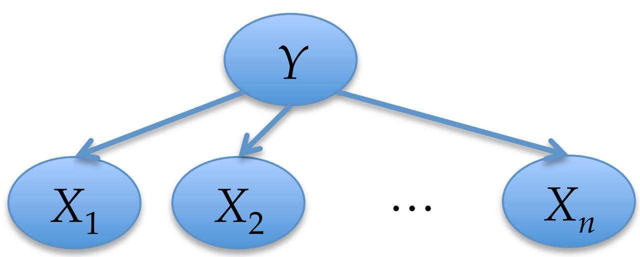


- For each class, think of the collection of X_i values from training examples in this class as a sample from $P(X_i | Y)$
  - Compute sample mean and variance as estimates for the Gaussian distribution parameters

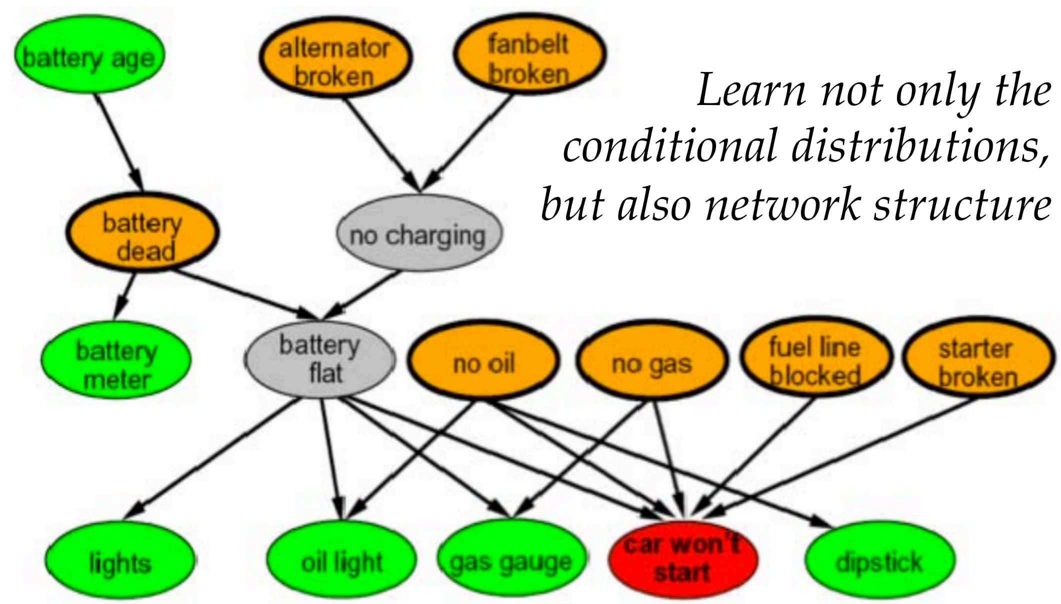
http://commons.wikimedia.org/wiki/File:Normal_distribution_pdf.png

The bigger picture

- When does Naïve Bayes shine?
  - Lots of features, not much data—it "generalizes" well
  - Too much data—it's simple and fast
- Beyond naïve: ***Bayesian networks***—to encode more general conditional independence assumptions



Naïve Bayes is just one special case



Learn not only the conditional distributions, but also network structure

Car example from http://courses.cs.washington.edu/courses/cse473/12sp/slides/27-learn-em.pdf

Outline

- Naïve Bayes classification
- ***Linear regression***
- Sample of other prediction methods
  - *k*NN (Lab #4)
  - Support vector machines
  - Decision trees

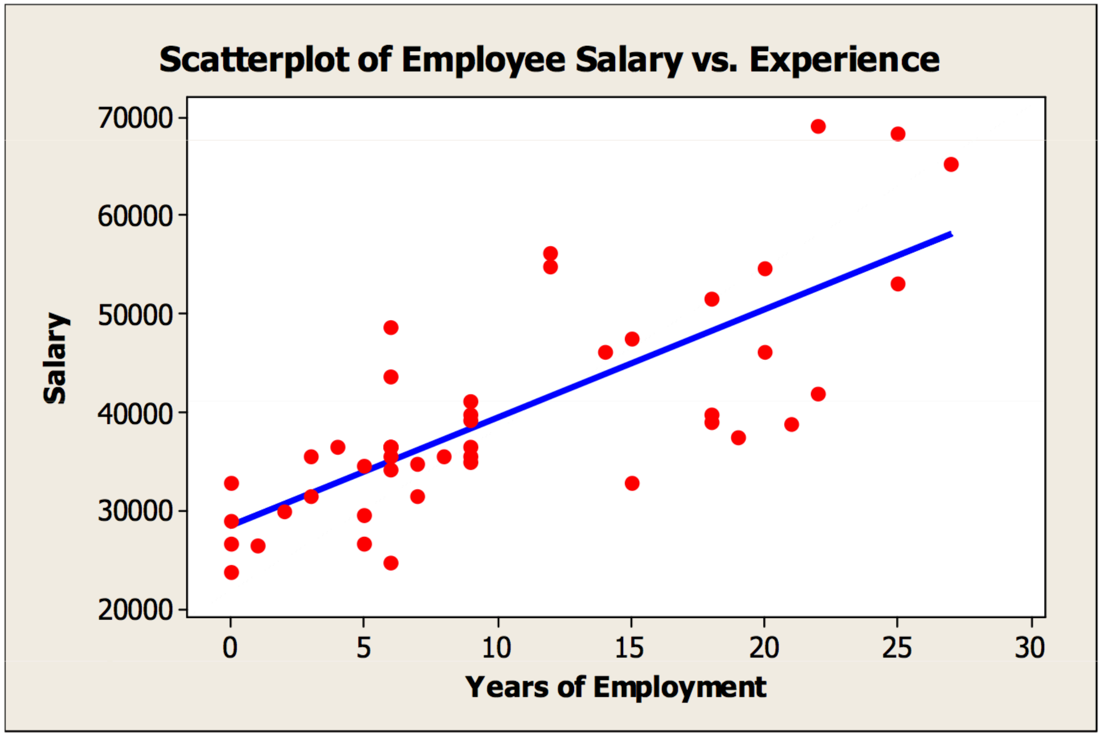
Regression

Given training data (x_1, y_1), (x_2, y_2), …, learn a function $X \rightarrow Y$, where Y is numerical

- Model how different variables relate
  - E.g., longer experience means higher salary
- Predict value of one variable given others
  - E.g., given his or her profile, how would a user rate a new fantasy-themed movie?

Linear regression

Assume linear relationships between two variables: $y = f(x) = \beta_0 + \beta_1 x$



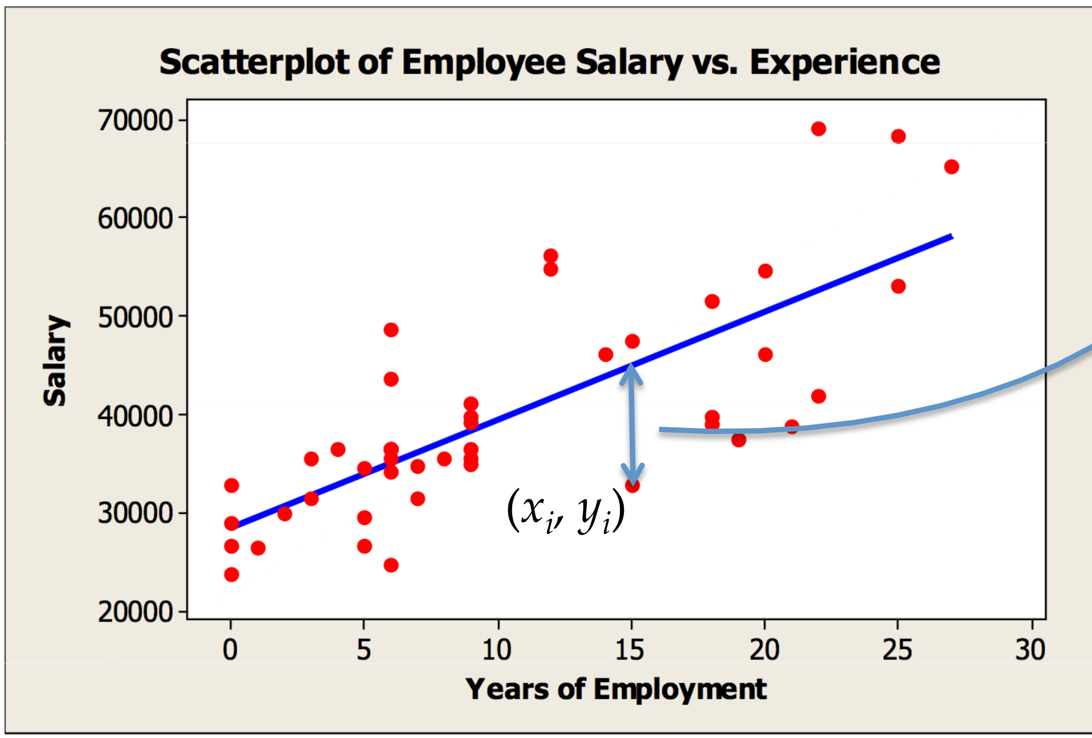
$$Salary = 28394 + 1107 \times \text{years}$$

Example from http://www.cs.cmu.edu/~neill/courses/95796-module3.pdf

How do we "fit the line"…

… given the training data $\{(x_i, y_i)\}$?

- Find the line to minimize overall error



Error for the i-th point is $f(x_i) - y_i$

What about the "overall" error?

"Least squares" regression

Minimize ***sum of squared errors***:

$$\sum_i (f(x_i) - y_i)^2$$

But why?

- Why not $\sum_i (f(x_i) - y_i)$?
- How about $\sum_i \left| f(x_i) - y_i \right|$?

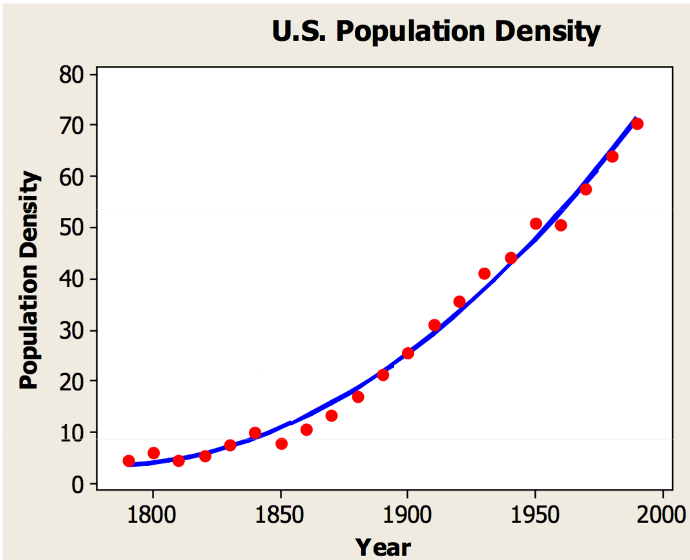
A model-based interpretation

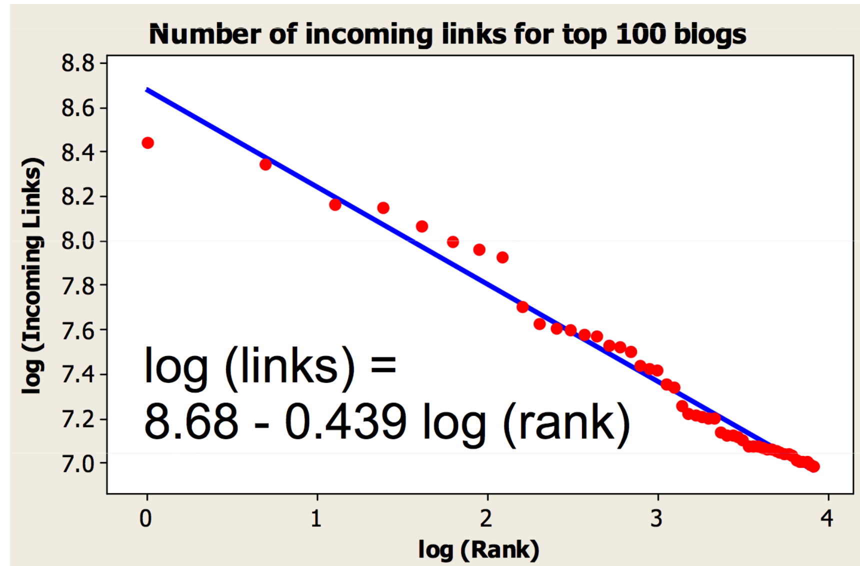
- Assume data follows $y = f(x) + \epsilon$
  - Where the noise ϵ follows Gaussian with 0 mean and σ^2 variance
- Compute the ***maximum likelihood estimate*** for (β_0, β_1)
  - Maximize the probability of seeing the set of y's for the set of x's in training data
- After you work out the math, it amounts to minimizing sum of squared errors

Least squares linear regression

- Closed-form formula are available for calculating the best fit directly
- How about more variables, e.g., Y vs. X_1 and X_2?
  - Generalization is straightforward
    - E.g., $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$; fit 3-d points to a plane
  - But beware of "multicollinearity"
    - E.g., if X_1 and X_2 are highly correlated, many planes might fit—there is no unique solution

Beyond "linear"





Add high-order terms as new variables

- E.g., *U.S. population density* = 4985 – 5.59 *year* + 0.00157 $year^2$
  - Regress on *year*, $year^2$

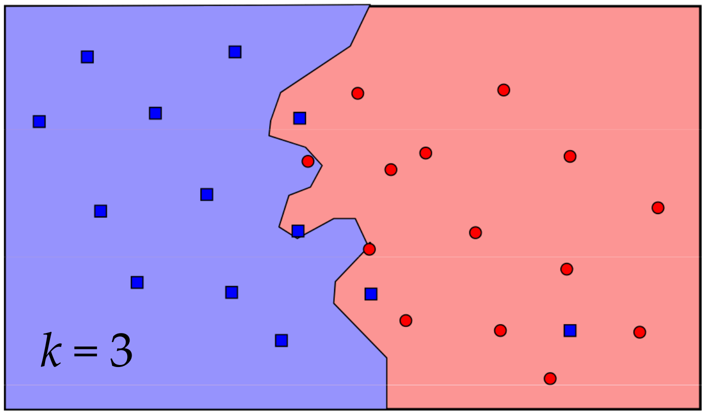
Transform variables to uncover linear relationships

- E.g., # of links to a blog decreases as a power law with popularity rank
  - *# links* = $10^{8.68}$/ $rank^{0.439}$

Examples from http://www.cs.cmu.edu/~neill/courses/95796-module3.pdf

Outline

- Naïve Bayes classification
- Linear regression
- Sample of other prediction methods
  - *k*NN (Lab #4)

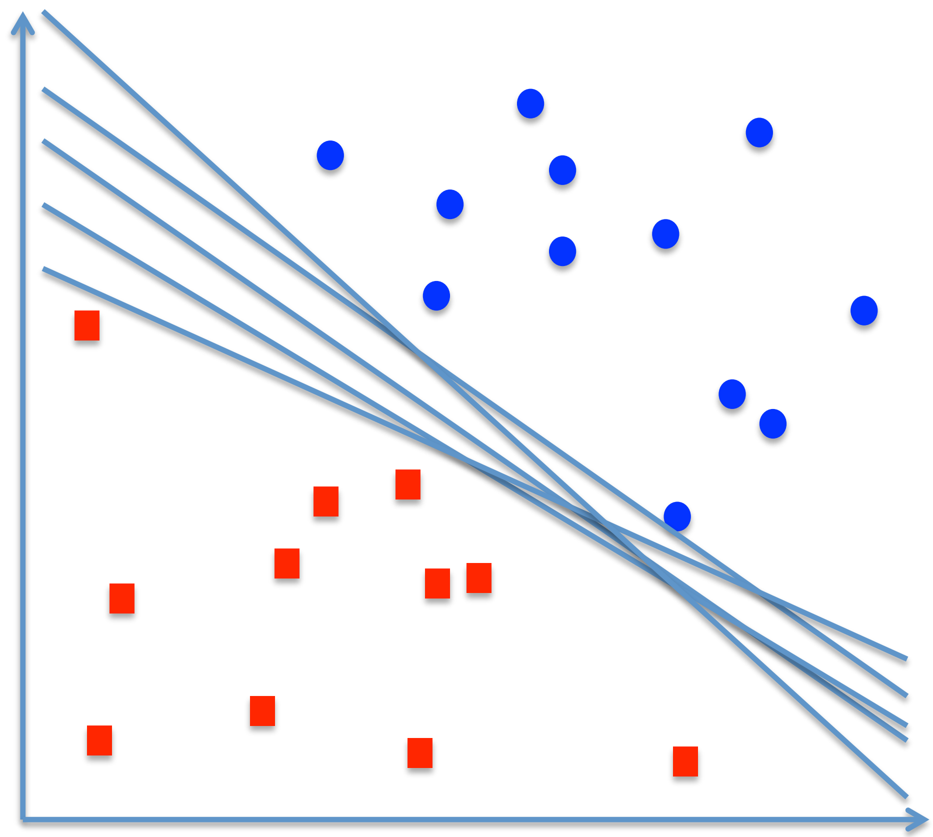


  - *Support vector machines*
  - Decision trees

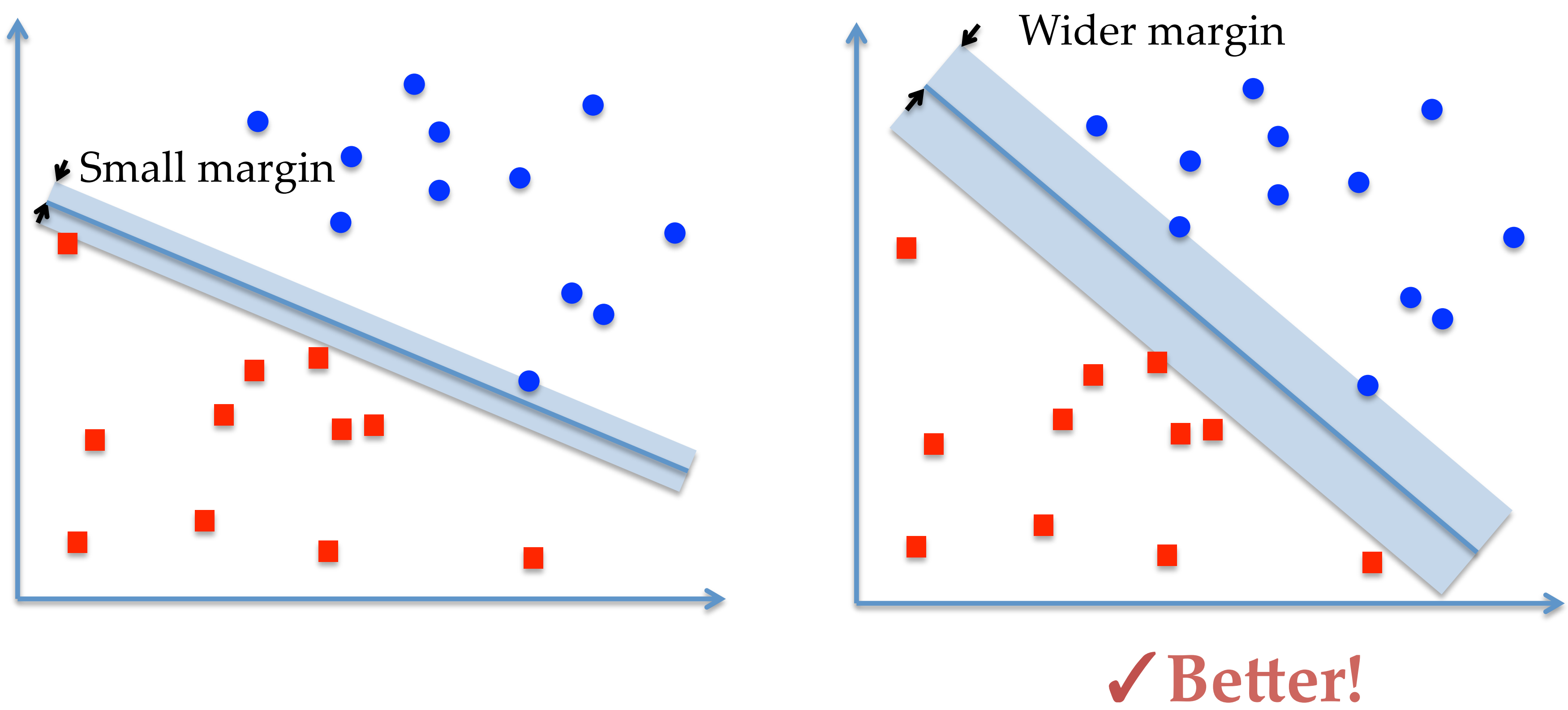
Intuition behind linear SVM

- Points labeled with two classes
- Find a hyperplane separating the two classes

But which one would you pick?



Max-margin classifier

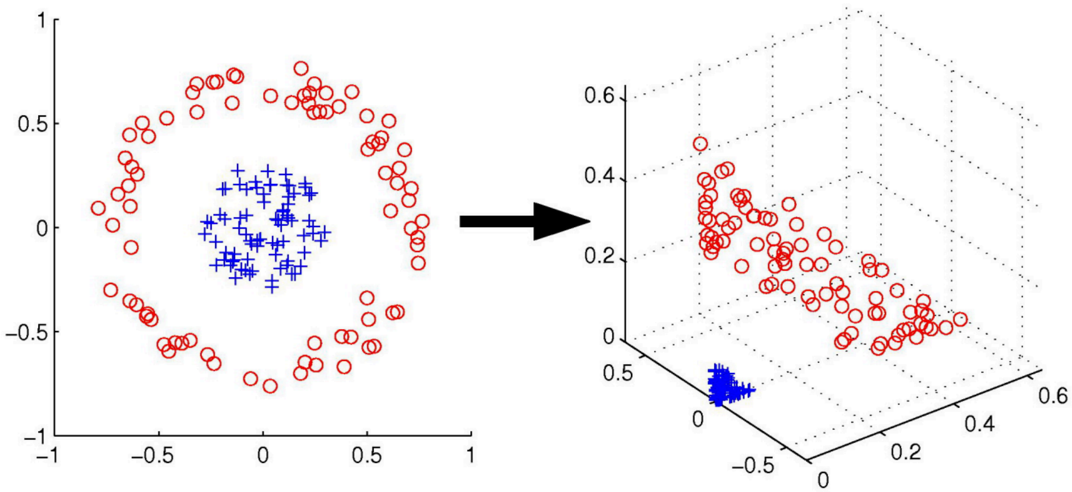


☞*Pick the hyperplane with the widest margin*

– Turns out this problem can be solved efficiently

Not linearly separable?

- Transform data to make it separable



$$(x_1, x_2) \mapsto (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

Instead of really transforming data, pick a distance metric (kernel), and the "kernel trick" will keep SVM efficient!

- In other cases, you can make the SVM "soft"
  - Allow misclassified points but pay a penalty

Image from http://www.cs.berkeley.edu/~jordan/courses/281B-spring04/lectures/lec3.pdf

Outline

- Naïve Bayes classification
- Linear regression
- Sample of other prediction methods
  - *k*NN (Lab #5)
  - Support vector machines
  - ***Decision trees***

Decision tree

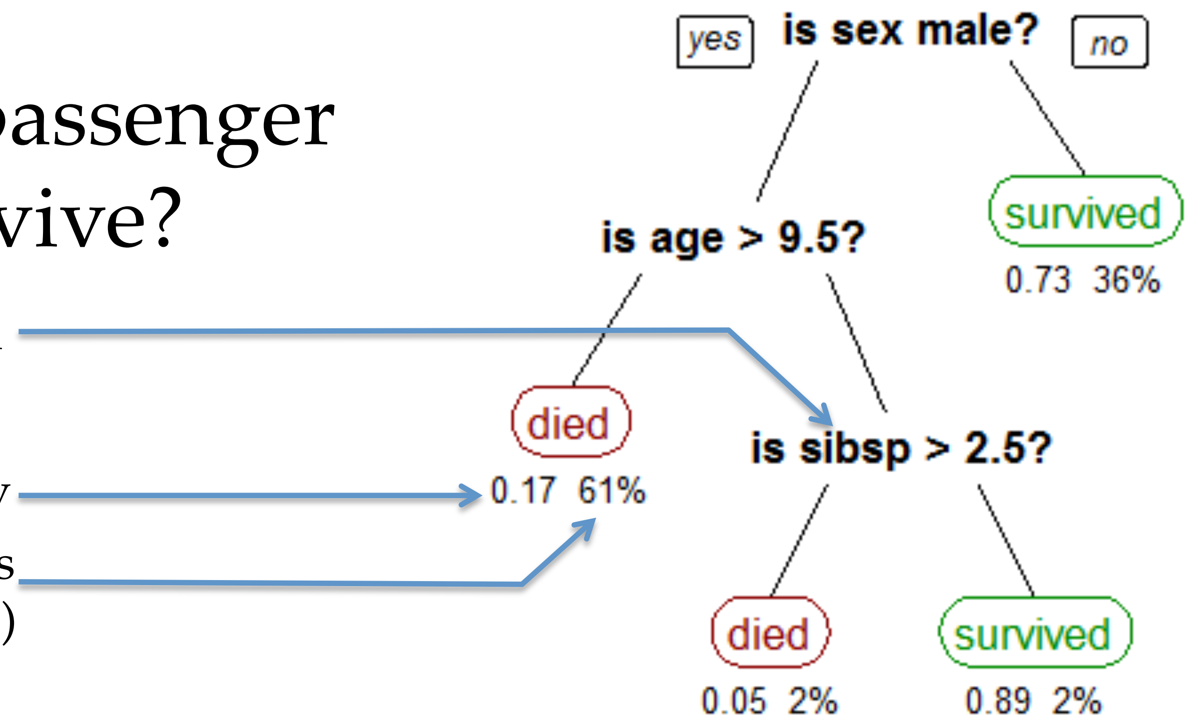
A series of questions lead you from the root to a leaf where a prediction can be made

- E.g., did the passenger on *Titanic* survive?

spouse/sibling on board

Survival probability

Percentage of observations (in training data)



http://en.wikipedia.org/wiki/File:CART_tree_titanic_survivors.png

Learning a decision tree

1. Start from with all data in a single node; predict the most common value
2. Choose the "best" question, and use it to split data in a node into two groups (in two child nodes)
   - What's the "best" split?
3. Repeat 2 above until some stopping criterion is reached
   - E.g., no more questioning/splitting will help

The "best" split

- Intuitively, ***entropy*** measures the expected # of bits needed to encode the information content of a distribution: $\sum_i -p_i \log_2 p_i$ where p_i is the probability of class i
  - E.g., entropy(unbiased coin) = 1; ← *Harder to predict*
    entropy(totally biased coin) = 0 ← *Easier to predict*
- ***Information gain*** of a split
  = (entropy of parent)
  – (weighted average entropy of children)

☞*Pick the split with the highest gain*

More about decision trees

- Worth trying if you have good features and want human-interpretable classifiers
- Small decision trees (if you manage to find them) are fast
  - "Best" splits are heuristic; no optimality guarantee
  - Trees can be large for some patterns of data
- Prone to overfitting; "pruning" nodes based on test data helps
- Can apply other tricks like *bagging* (let many trees vote) or *boosting* (reweight mistakes to train "better" trees)

Summary

- Model-based learning
  - Naïve Bayes
  - Linear regression
- Instance-based learning: *k*NN
- Powerful, but blackbox: SVMs
- Rule-based learning: decision trees

A few takeaway points

- Train/test, cross-validation
- Overfitting and underfitting
- Power of ***model***: *All models are wrong but some are useful* – George E. P. Box
- Power of ***geometry***: data as points in high dimensions
- How do we cope with this bewildering collection of tools?