COMP9321

Data Services Engineering

Term 1, 2019

Week 6 Lecture 1

Quiz 4

1. Which HTTP method is suitable for updating resources?

- PUT
- UPDATE
- POST
- POST and OPTIONS

Quiz 4

2. POST is neither idempotent nor safe operation

- True
- False

Quiz 4

3. Having Uniform Interfaces in RESTful Services mean

- The developers do not have to implement the operations as they are standards
- If the conventions are properly followed, understanding the interface is easy
- The developers can build more secure applications
- Standard data types for HTTP operations

Quiz 4

4. Which of the following is correct of a resource in RESTful services

- A resource is not to be updated by the client application to maintain statelessness

- A resource can have many representations

- A resource is a collection of hidden data set managed by a RESTful service

Quiz 4

5. Which one of the following is both Safe and Idempotent?

- HTTP DELETE
- HTTP PATCH
- HTTP GET
- HTTP PUT

Supervised Learning

COMP9321 2019T1

Supervised Learning

We are given input samples (X) and output samples (y) of a function y = f(X).

We would like to "learn" f, and evaluate it on new data.

- **Classification:** y is discrete (class labels).
- **Regression:** y is continuous, e.g. linear regression.

Supervised Learning

Given training data $\{(x_1, y_1), \ldots, (x_N, y_N)\}$

N input/output pairs; x_i - input, y_i - output/label

x_i is a **vector** consisting of D features

Also called attributes or dimensions Features can be discrete or continuous

x_{im} denotes the m-th feature of x_i

Forms of the output:

- $y_i \in \{1, \ldots, C\}$ for classification; a discrete variable
- $y_i \in R$ for regression; a continuous (real-valued) variable

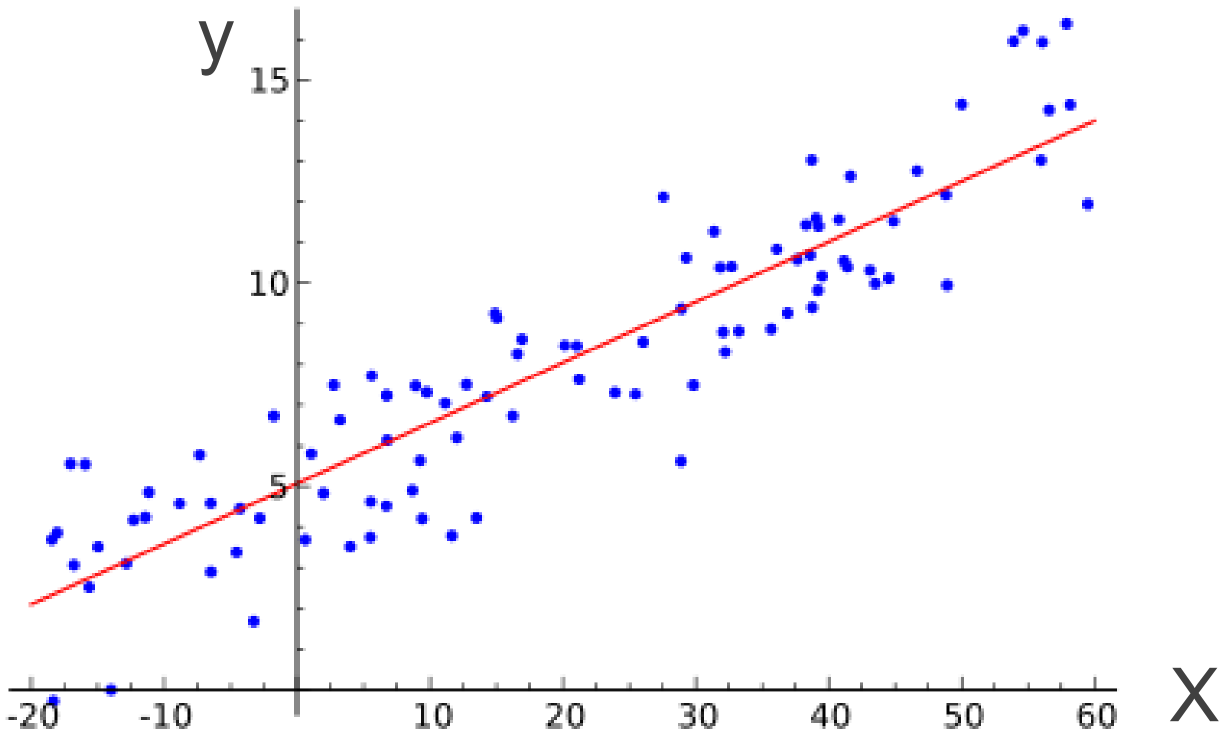
Goal: predict the output y for an unseen test example x

Linear Regression

Supervised Learning

Linear Regression

We want to find the "best" line (linear function y=f(X)) to explain the data.



Linear Regression

The predicted value of y is given by:

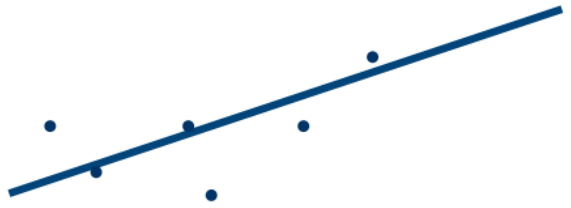
$$\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j$$

The vector of coefficients $\hat{\beta}$ is the regression model.

Linear Regression

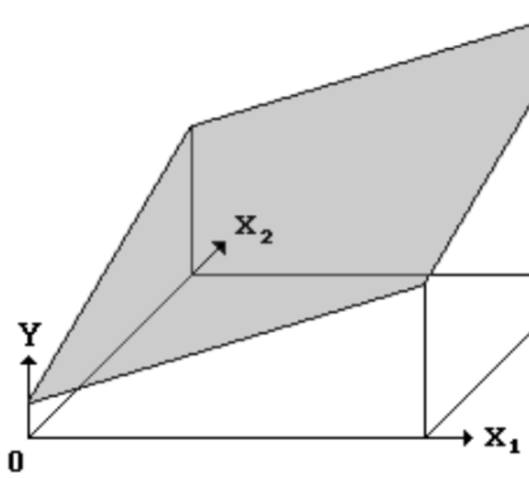
Simple linear regression

Y = β0 + β1X1 + ε



Multiple linear regression

Y = β0 + β1X1 + β2X2 + ε



Linear Regression

The regression formula $\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j + \varepsilon$

e.g., j = 1

Random error

$$\hat{y} = \hat{\beta}_0 + X_1 \hat{\beta}_1 + \varepsilon$$

predictor

Slope of the line

Intercept (where the line crosses y-axis)

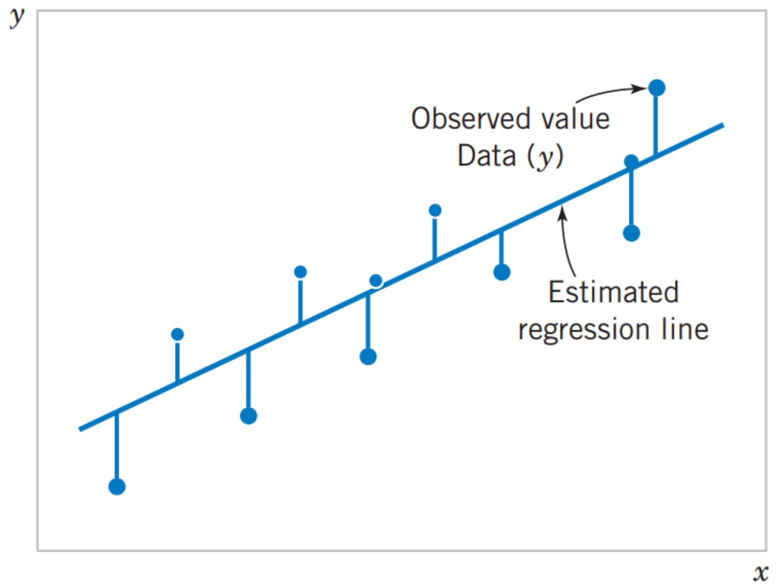
The slope and intercept of the line are called regression coefficients, model parameters

Our goal is to estimate the model parameters

$$\text{Min SS}(\beta) = \sum_{i=1}^{N} (y_i - X_i \beta)^2$$

Least Square Error Solution

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \qquad i = 1, 2, \ldots, n$$



To estimate (β0,β1) , we find values that minimize squared error

$$L = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

Solution:

$$\left.\frac{\partial L}{\partial \beta_0}\right|_{\hat\beta_0,\hat\beta_1} = -2\sum_{i=1}^{n}(y_i - \hat\beta_0 - \hat\beta_1 x_i) = 0$$

$$\left.\frac{\partial L}{\partial \beta_1}\right|_{\hat\beta_0,\hat\beta_1} = -2\sum_{i=1}^{n}(y_i - \hat\beta_0 - \hat\beta_1 x_i)x_i = 0$$

$$\Rightarrow$$

$$n\hat\beta_0 + \hat\beta_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$$

$$\hat\beta_0 \sum_{i=1}^{n} x_i + \hat\beta_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} y_i x_i$$

Least Square Error Solution

The least squares estimates of the intercept and slope in the simple linear regression model are

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} y_i x_i - \frac{\left(\sum_{i=1}^{n} y_i\right)\left(\sum_{i=1}^{n} x_i\right)}{n}}{\sum_{i=1}^{n} x_i^2 - \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n}}$$

where $\bar{y} = (1/n)\sum_{i=1}^{n} y_i$ and $\bar{x} = (1/n)\sum_{i=1}^{n} x_i$.

Linear Regression

The regression formula $\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j$

if $X_0 = 1$, can be written as a matrix product with X a row vector:

$$\hat{y} = \mathrm{X}\,\hat{\beta}$$

We get this by writing all of the input samples in a single matrix **X**:

i.e. **rows of $\mathbf{X}$** $= \begin{pmatrix} X_{11} & \cdots & X_{1n} \\ \vdots & \ddots & \vdots \\ X_{m1} & \cdots & X_{mn} \end{pmatrix}$

are **distinct observations, columns of X** are **input features**.

Least Squares Solution

The most common measure of fit between the line and the data is the **least-squares fit**.

There is a good reason for this: If the points are generated by an ideal line with additive Gaussian noise, the least squares solution is the ***maximum likelihood solution***.

Probability of a point y_j is $\Pr(y_j) = \exp\left(\frac{-(y_j - X_j\beta)^2}{2\sigma^2}\right)$ and the probability for all points is the product over j of $\Pr(y_j)$.

We can **easily maximize the log** of this expression $\frac{-(y_j - X_j\beta)^2}{2\sigma^2}$ for one point, or the sum of this expression at all points.

Least Squares Solution

To determine the model parameters $\hat{\beta}$ from some data, we write down the Sum of Squares:

$$SS(\beta) = \sum_{i=1}^{N} (y_i - X_i\beta)^2$$

or symbolically $SS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$. To minimize it, take the derivative wrt β which gives:

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0$$

And if $\mathbf{X}^T\mathbf{X}$ is non-singular, the unique solution is:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Least Squares Solutions

The exact method requires us to invert a matrix $(\mathbf{X}^T\mathbf{X})$ whose size is M^2 for M features and takes time $O(M^3)$. This is **too big** for large feature spaces like text or event data.

Gradient methods reduce the SS error using the **derivative wrt** β

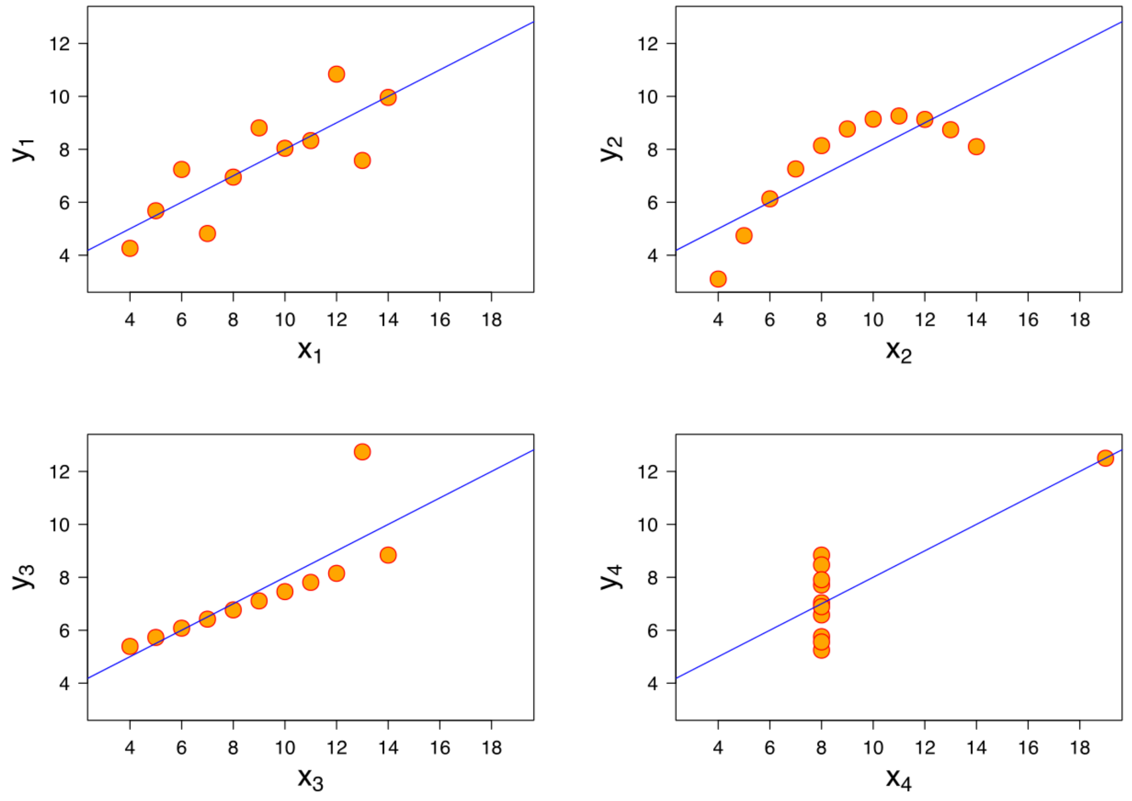
$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N} (y_i - \beta x_i)^2$$

which is

$$\nabla = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta)$$

R^2-values and P-values

We can **always** fit a linear model to any dataset, but how do we know if there is a **real linear relationship**?



R^2-values

Approach: Measure how much the total “noise” (variance) is reduced when we include the line as an offset.

R-squared: a suitable measure. Let $\hat{y} = X\hat{\beta}$ be a predicted value, and $\bar{y}$ be the sample mean. Then the R-squared value is

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

And can be described as the fraction of the total variance not explained by the model.

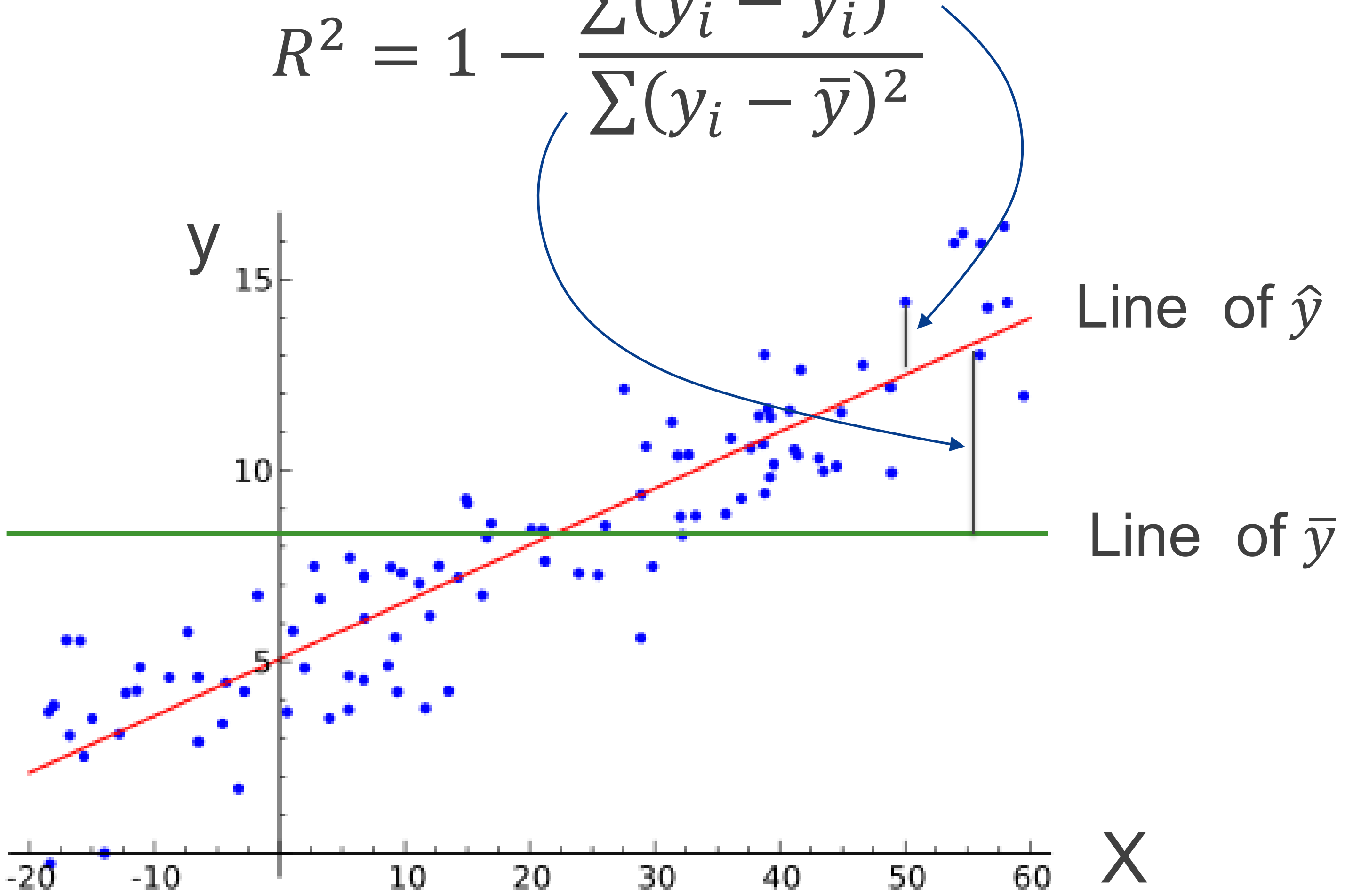
$R^2 = 0$: bad model. No evidence of a linear relationship.

$R^2 = 1$: good model. The line perfectly fits the data.

R-squared

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

Small if good fit



R^2-values and P-values

Statistic: From R-squared we can derive another statistic (using degrees of freedom) that has a standard distribution called an **F-distribution**.

From the CDF for the F-distribution, we can derive a **P-value** for the data.

The P-value is, as usual, the probability of observing the data under the null hypothesis of no linear relationship.

If **p is small**, say less than 0.05, we conclude that **there is a linear relationship**.

Logistic Regression

Supervised Learning

Logistic regression

- Name is somewhat misleading. Really a technique for classification, not regression.
  - “Regression” comes from fact that we fit a linear model to the feature space.
  - logit regression, maximum-entropy classification, log-linear classifier
- Involves a more probabilistic view of classification.

Modeling binary data

Often in medical studies, we encounter outcomes that are not continous, but instead fall into 1 of 2 categories. For example:

- Disease status (disease vs. no disease)
- Alive or dead
- Low birth weight
- Improved health status

Modeling binary data

In these cases, we have a binary outcome

$$y_i = \begin{cases} 0 \;\; with\, probability\; 1 - \pi_i \\ 1 \; with\, probability\; \pi_i \end{cases}$$

where

$$E[y_i] = \pi_i$$

and

$$\text{var}[y_i] = \pi_i(1 - \pi_i).$$

Usually, one of the categories is the outcome of interest, like death or disease. This category is usually coded as 1.

Modeling binary data

We can use linear regression to model this outcome, but this can present several problems as we will see.

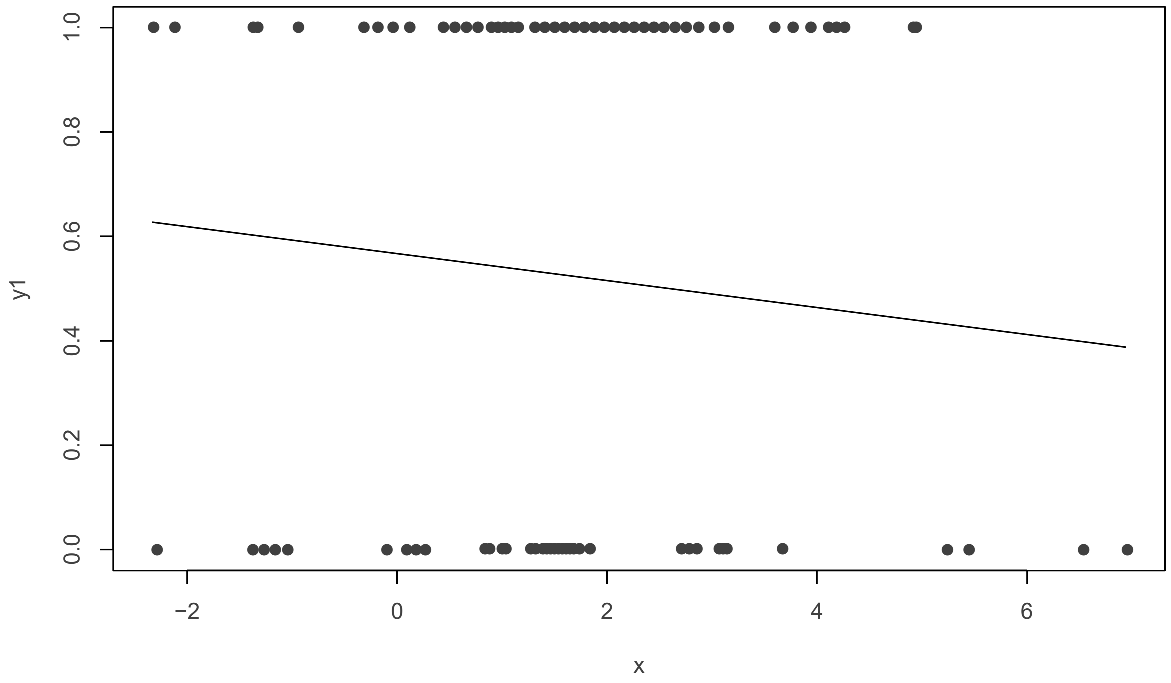
Usingthe linearmodelapproach, we relate the expected value of yi to a predictor xi as

$$E[y_i] = \beta_0 + \beta_1 x_i$$

Just looking at this relationship, we can see a potential problem. What is it?

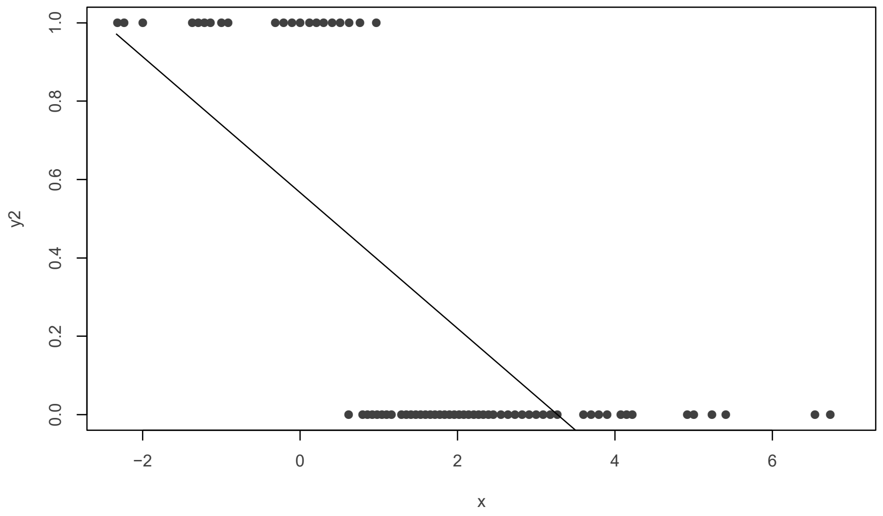
Modeling binary data

Over small ranges of the predictor or when the relationship between the predictor and the outcome is not strong, this may not be troubling.



Modeling binary data

However, if the association is strong, potential problems are more evident.



We could put constraints on the βs that would prevent this from happening, but this would be complicated and probably not the best way to address this problem.

Different ways of expressing probability

- Consider a two-outcome probability space, where:
  - $p(O_1) = p$
  - $p(O_2) = 1 – p = q$
- Can express probability of O_1 as:

	notation	range	equivalents	
standard probability	p	0	0.5	1
odds	p / q	0	1	+ ∞
log odds (logit)	log(p / q)	- ∞	0	+ ∞

Log odds

- Numeric treatment of outcomes O_1 and O_2 is equivalent
  - If neither outcome is favored over the other, then log odds = 0.
  - If one outcome is favored with log odds = x, then other outcome is disfavored with log odds = -x.
- Especially useful in domains where relative probabilities can be miniscule
  - Example: multiple sequence alignment in computational biology

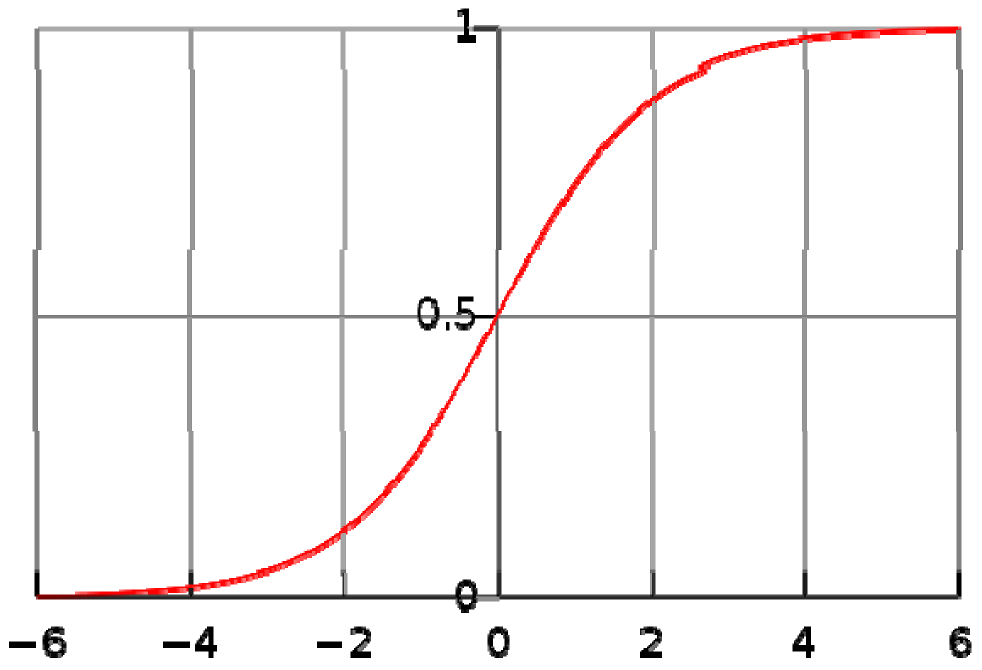
From probability to log odds (and back again)

$$z = \log\left(\frac{p}{1-p}\right)$$ logit function

$$\frac{p}{1-p} = e^z$$

$$p = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}}$$ logistic function

Standard logistic function



Logistic regression

- Scenario:
  - A multidimensional feature space (features can be categorical or continuous).
  - Outcome is discrete, not continuous.
    - We'll focus on case of two classes.
  - It seems plausible that a linear decision boundary (hyperplane) will give good predictive accuracy.

Using a logistic regression model

- Model consists of a vector β in *d*-dimensional feature space

- For a point **x** in feature space, project it onto β to convert it into a real number *z* in the range - ∞ to + ∞

$$z = \alpha + \boldsymbol{\beta} \cdot \mathbf{x} = \alpha + \beta_1 x_1 + \ldots + \beta_d x_d$$

- Map *z* to the range 0 to 1 using the logistic function

$$p = 1/(1 + e^{-z})$$

- Overall, logistic regression maps a point **x** in *d*-dimensional feature space to a value in the range 0 to 1

Using a logistic regression model

- Can interpret prediction from a logistic regression model as:
  - A probability of class membership
  - A class assignment, by applying threshold to probability
    - threshold represents decision boundary in feature space

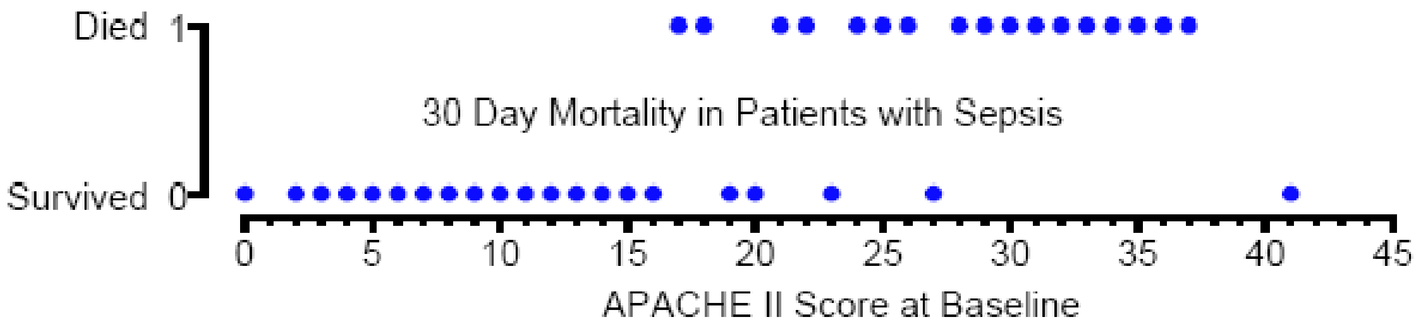
Training a logistic regression model

- Need to optimize β so the model gives the best possible reproduction of training set labels
  - Usually done by numerical approximation of maximum likelihood
  - On really large datasets, may use stochastic gradient descent

Logistic regression in one dimension

a) Example: APACHE II Score and Mortality in Sepsis

The following figure shows 30 day mortality in a sample of septic patients as a function of their baseline APACHE II Score. Patients are coded as 1 or 0 depending on whether they are dead or alive in 30 days, respectively.

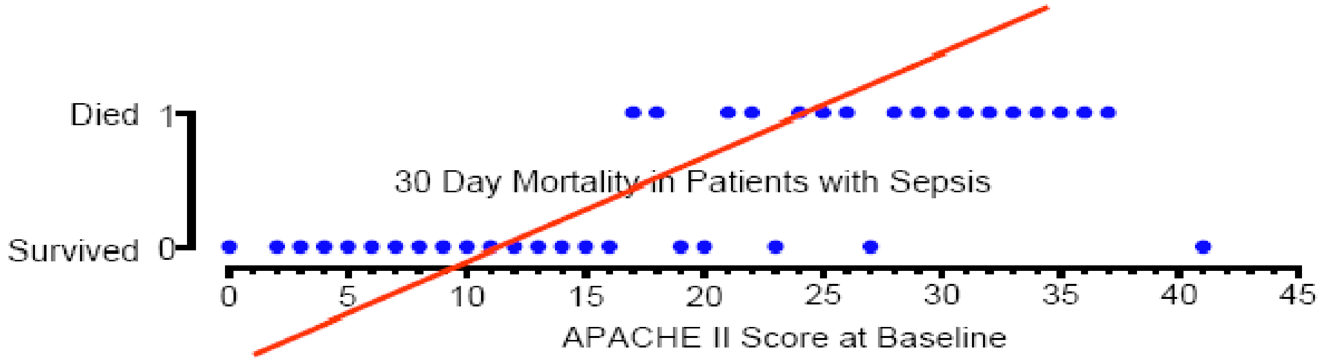


Logistic regression in one dimension

We wish to predict death from baseline APACHE II score in these patients.

Let $\pi(x)$ be the probability that a patient with score x will die.

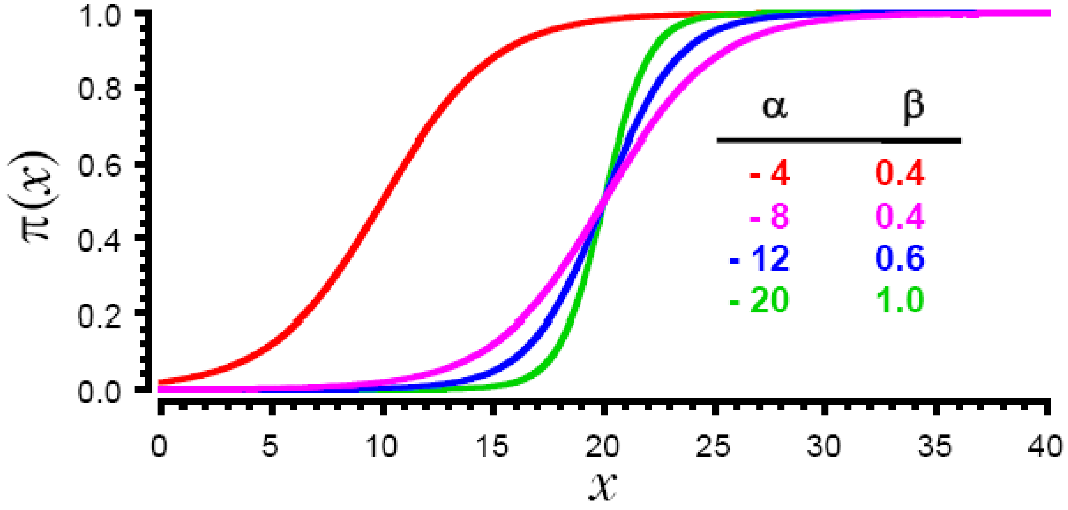
Note that linear regression would not work well here since it could produce probabilities less than zero or greater than one.



Logistic regression in one dimension

- Parameters control shape and location of sigmoid curve
  - α controls location of midpoint
  - β controls slope of rise

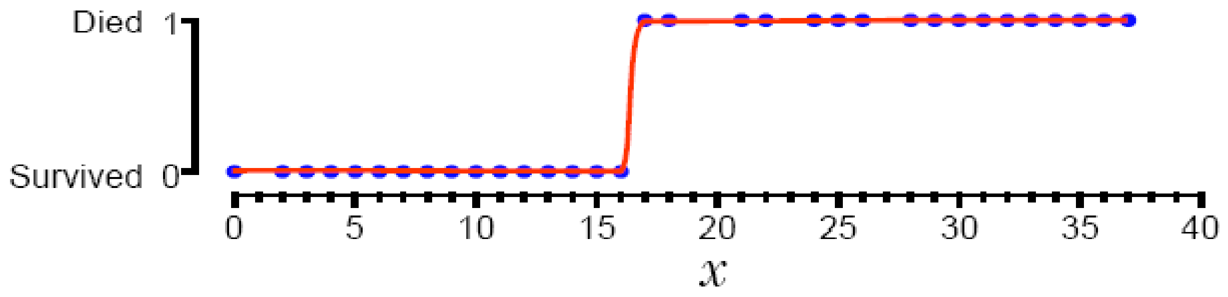
$$\pi(x) = \exp(\alpha + \beta x) / (1 + \exp(\alpha + \beta x))$$



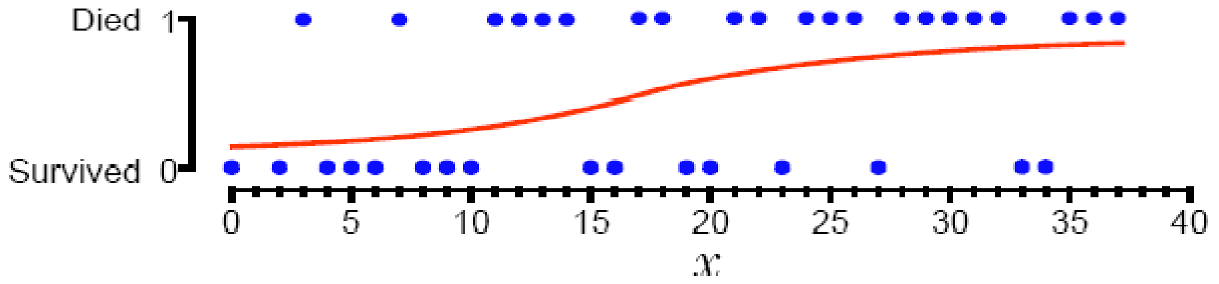
When $x = -\alpha / \beta, \quad \alpha + \beta x = 0$ and hence $\pi(x) = 1/(1+1) = 0.5$

Logistic regression in one dimension

Data that has a sharp survival cut off point between patients who live or die should have a large value of β.



Data with a lengthy transition from survival to death should have a low value of β.



Logistic regression in one dimension

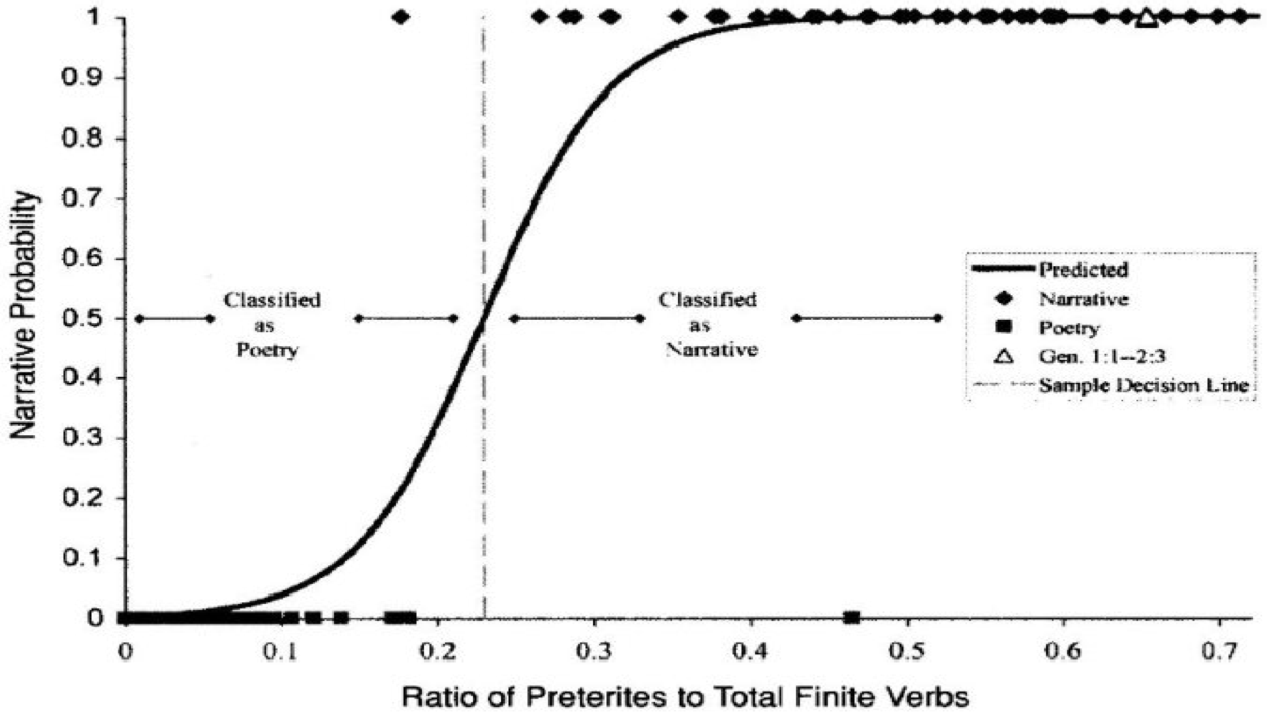
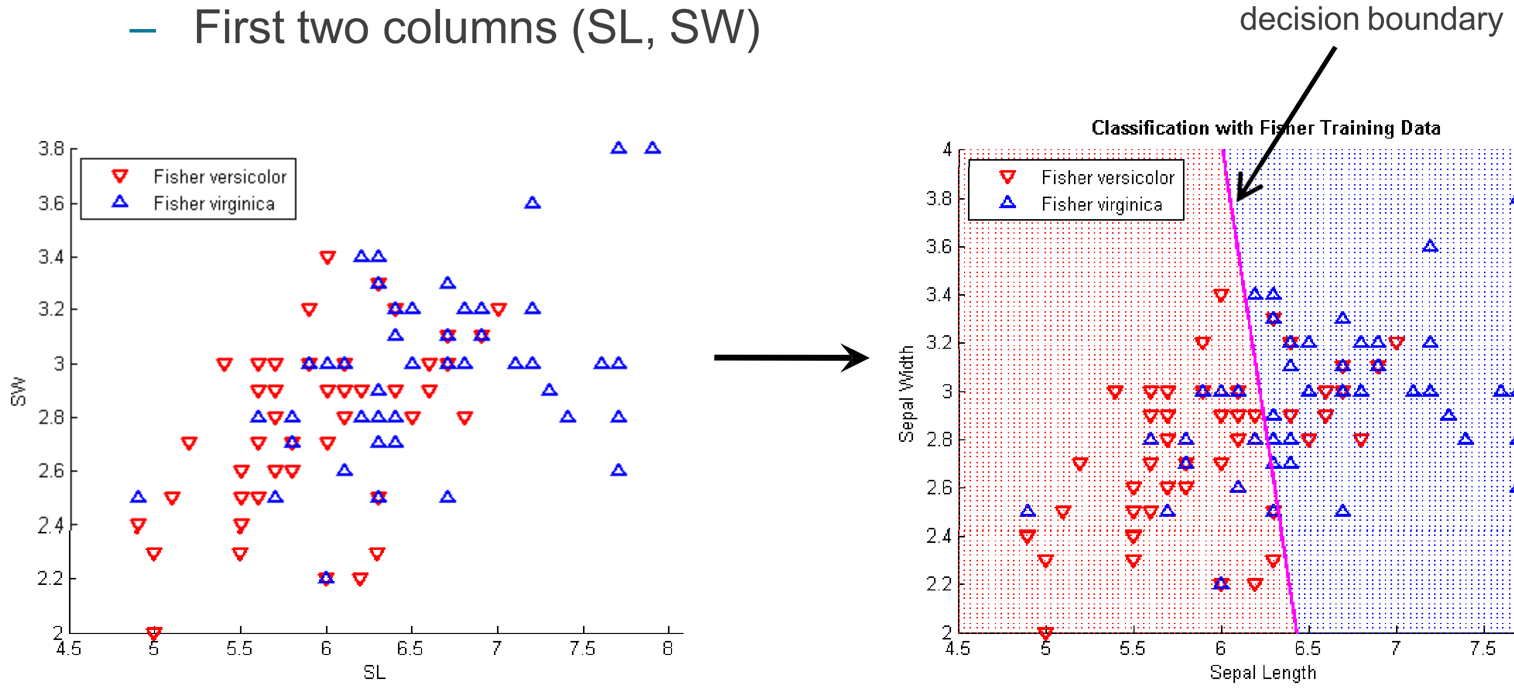


Figure 10-3. The solid curved line is called a logistic regression curve. The vertical axis measures the probability that an Old Testament passage is narrative, based on the use of preterite verbs. The probability is zero for poetry and unity or one for narrative. Passages with high preterite verb counts, falling to the right of the vertical dotted line, are likely narrative. The triangle on the upper right represents Genesis 1:1–2:3, which is clearly literal, narrative history.

Logistic regression in two dimensions

Subset of Fisher iris dataset

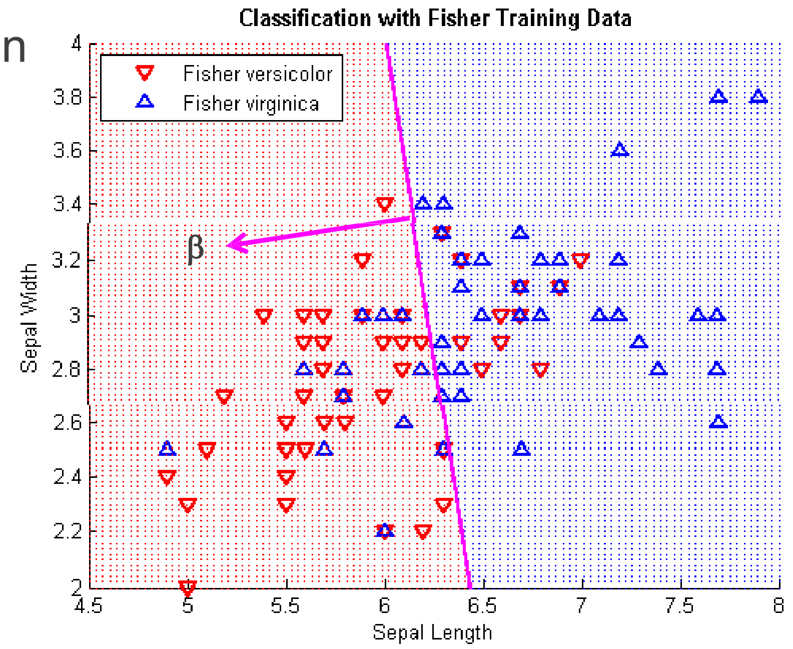
- Two classes
- First two columns (SL, SW)



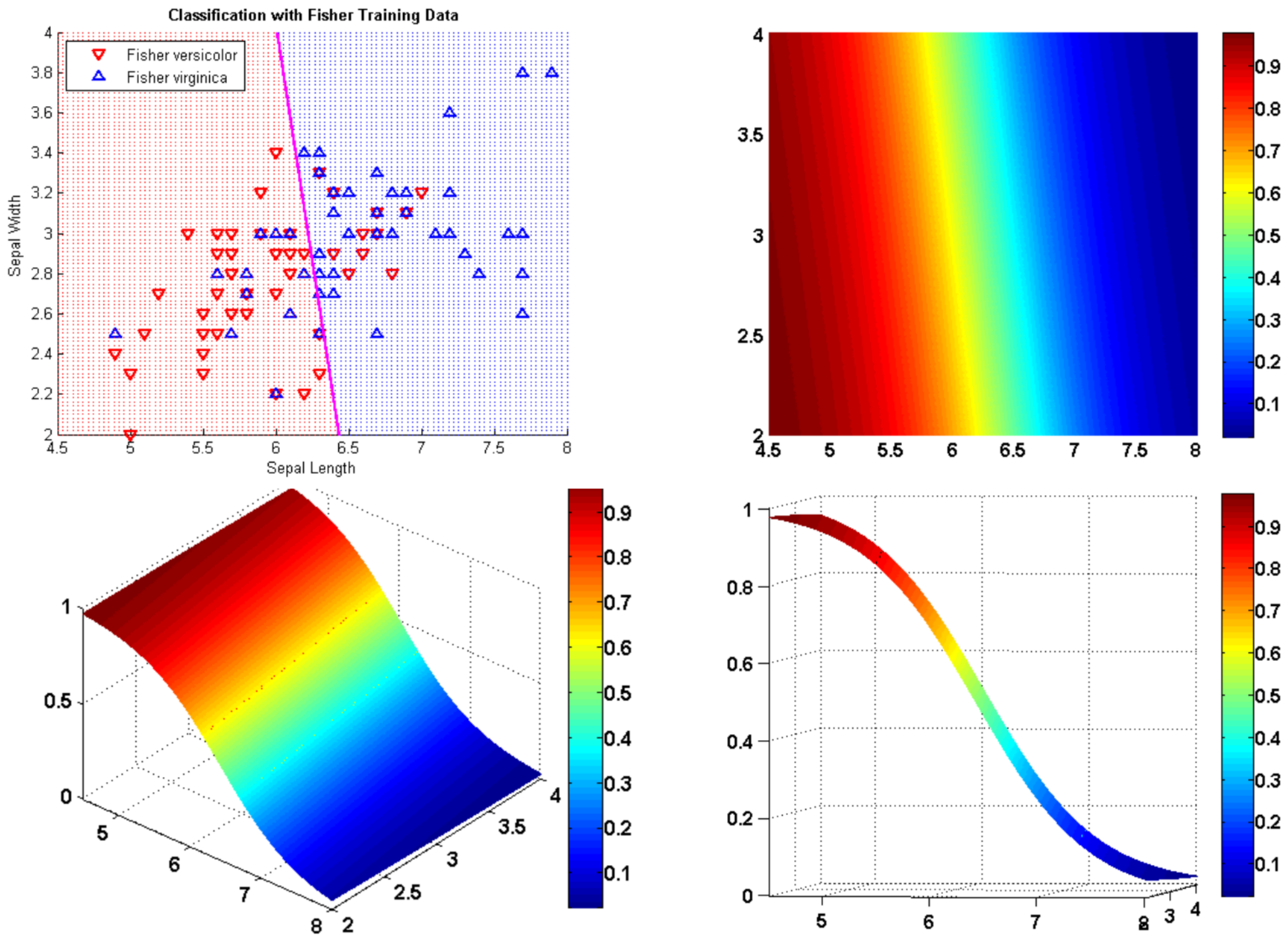
Logistic regression in two dimensions

Interpreting the model vector of coefficients

- From MATLAB: `B = [ 13.0460   -1.9024   -0.4047 ]`
- α = B(1), $\boldsymbol{\beta}$ = [β_1 β_2] = B(2 : 3)
- α, $\boldsymbol{\beta}$ define location and orientation of decision boundary
  - - α is distance of decision boundary from origin
  - decision boundary is perpendicular to $\boldsymbol{\beta}$
- magnitude of $\boldsymbol{\beta}$ defines gradient of probabilities between 0 and 1



Logistic regression in two dimensions



Logistic regression

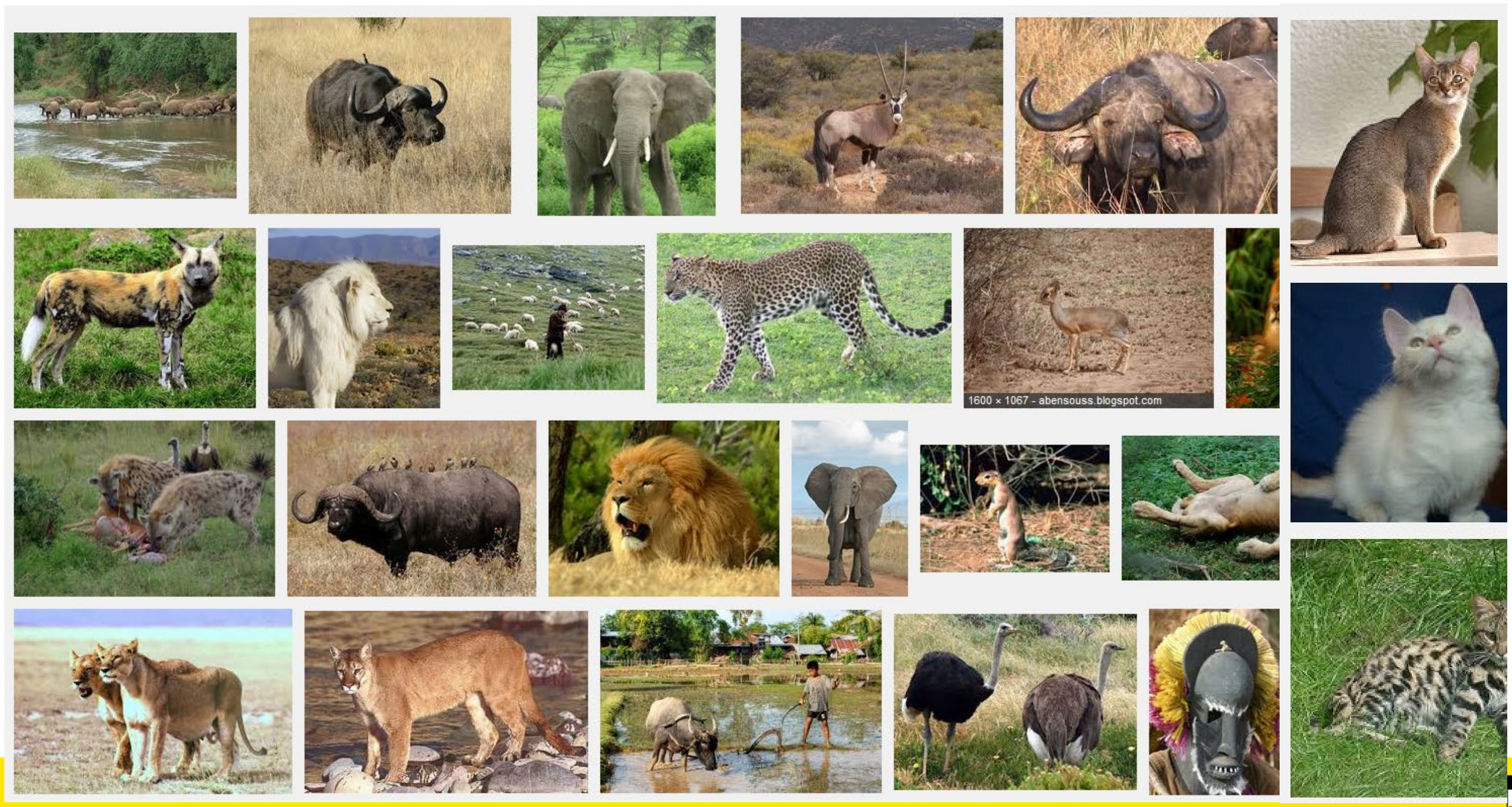
- Advantages:
  - Makes no assumptions about distributions of classes in feature space
  - Easily extended to multiple classes (multinomial regression)
  - Natural probabilistic view of class predictions
  - Quick to train
  - Very fast at classifying unknown records
  - Good accuracy for many simple data sets
  - Resistant to overfitting
  - Can interpret model coefficients as indicators of feature importance
- Disadvantages:
  - Linear decision boundary

k-Nearest Neighbour

Supervised Learning

k-Nearest Neighbors

Given a query item:
Find k closest matches
in a labeled dataset ↓



k-Nearest Neighbors

Given a query item:

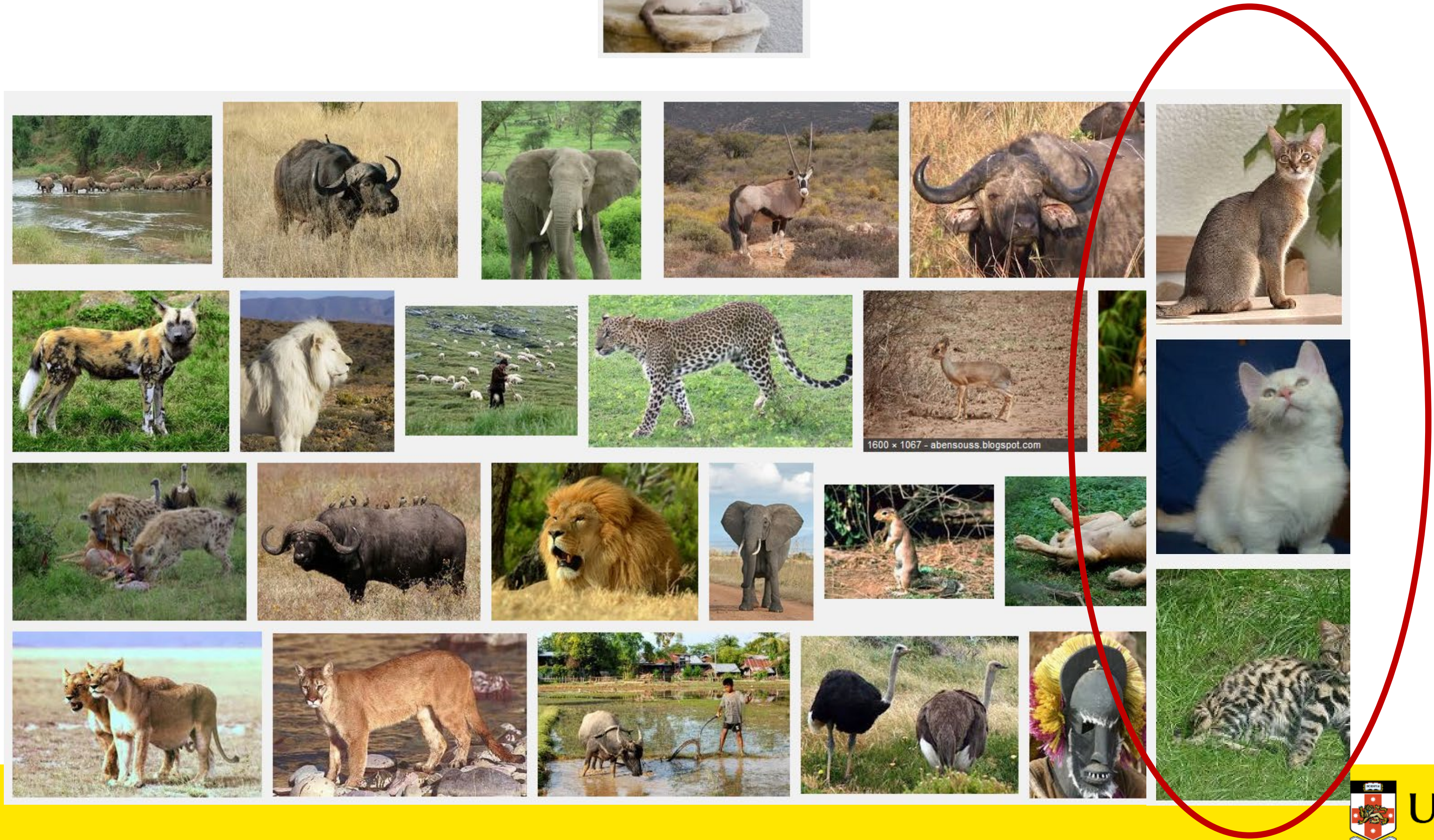
Find k closest matches

Return the most

Frequent label

k-Nearest Neighbors

k = 3 votes for “cat”



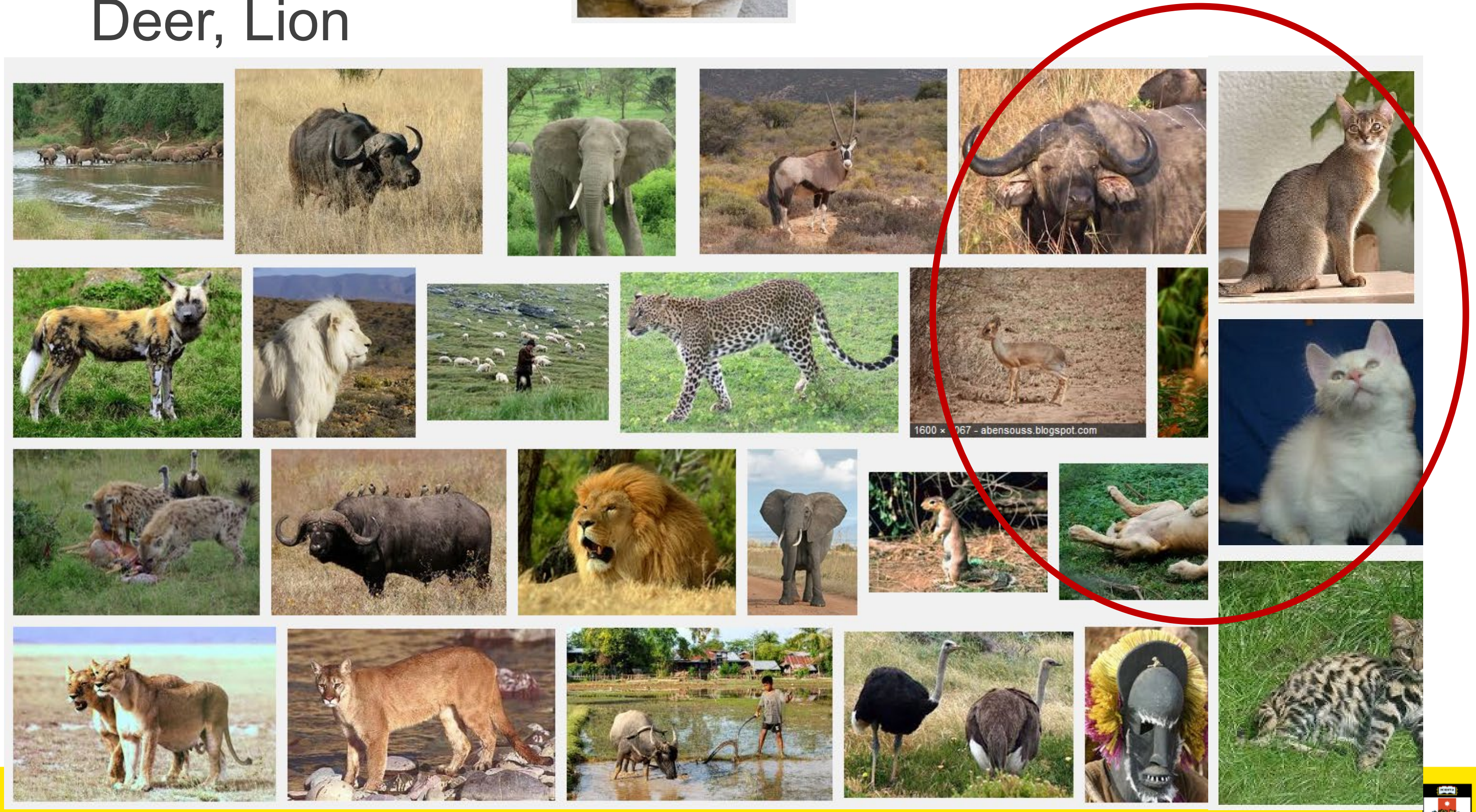
k-Nearest Neighbors

2 votes for cat,

1 each for Buffalo,

Deer, Lion

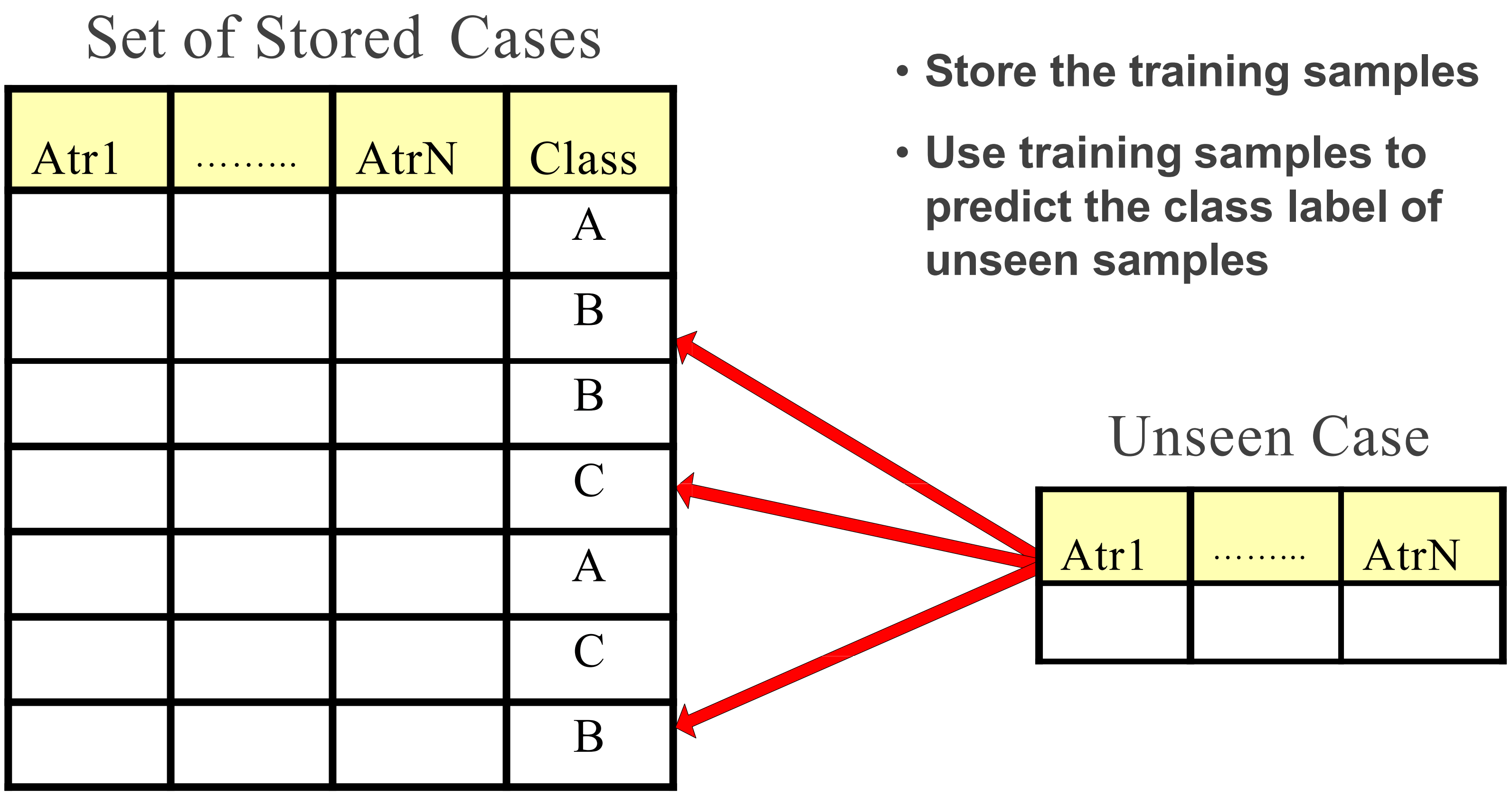
Cat wins…



Nearest Neighbor Algorithm

- Learning Algorithm:
  - Store training examples
- Prediction Algorithm:
  - To classify a new example **x** by finding the training example ($\mathbf{x}^i$,y^i) that is *nearest* to **x**
  - Guess the class $y = y^i$

Instance based classifiers



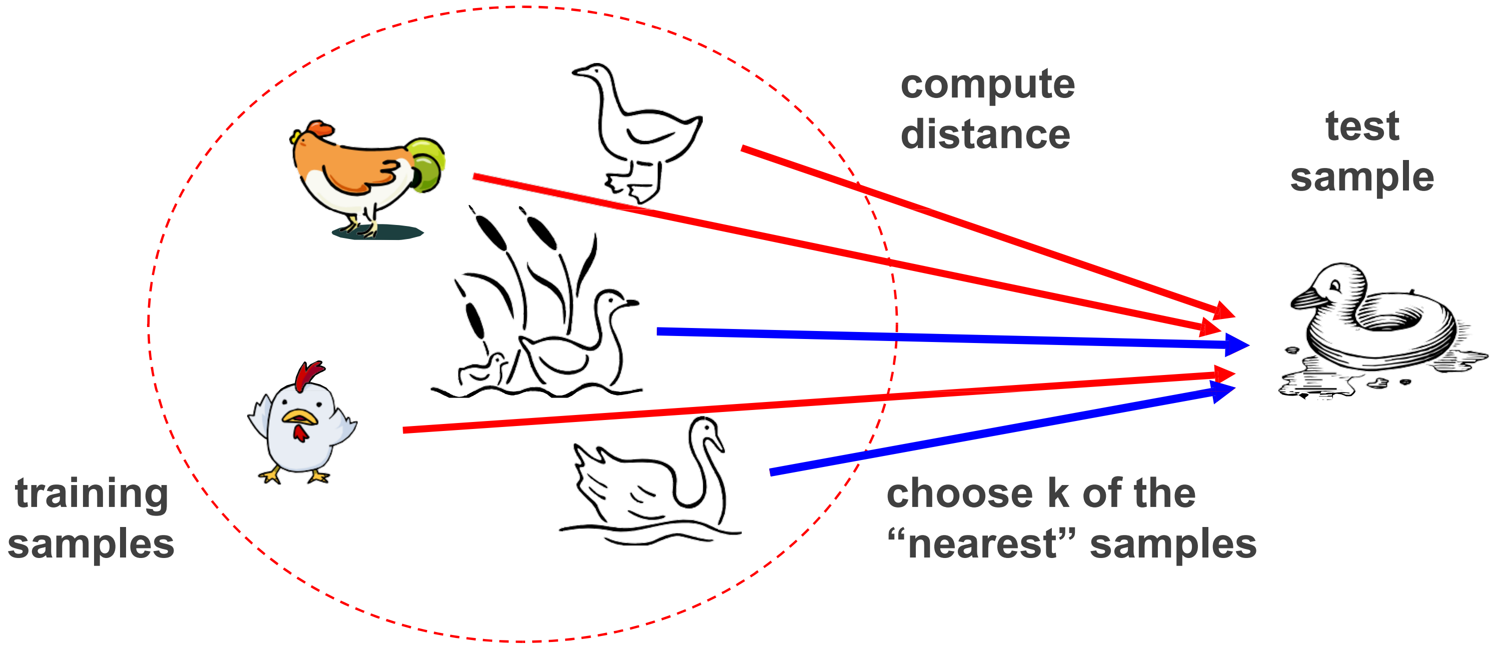
- Store the training samples
- Use training samples to predict the class label of unseen samples

Instance based classifiers

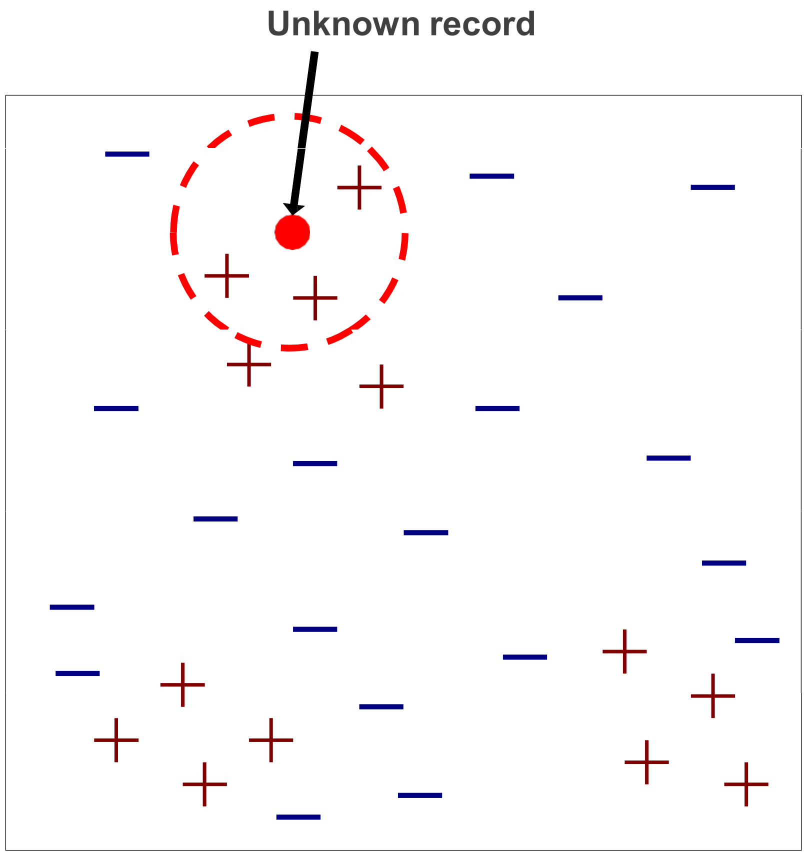
- Examples:
  - Rote learner
    - memorize entire training data
    - perform classification only if attributes of test sample match one of the training samples exactly
  - Nearest neighbor
    - use *k* “closest” samples (nearest neighbors) to perform classification

Nearest neighbor classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



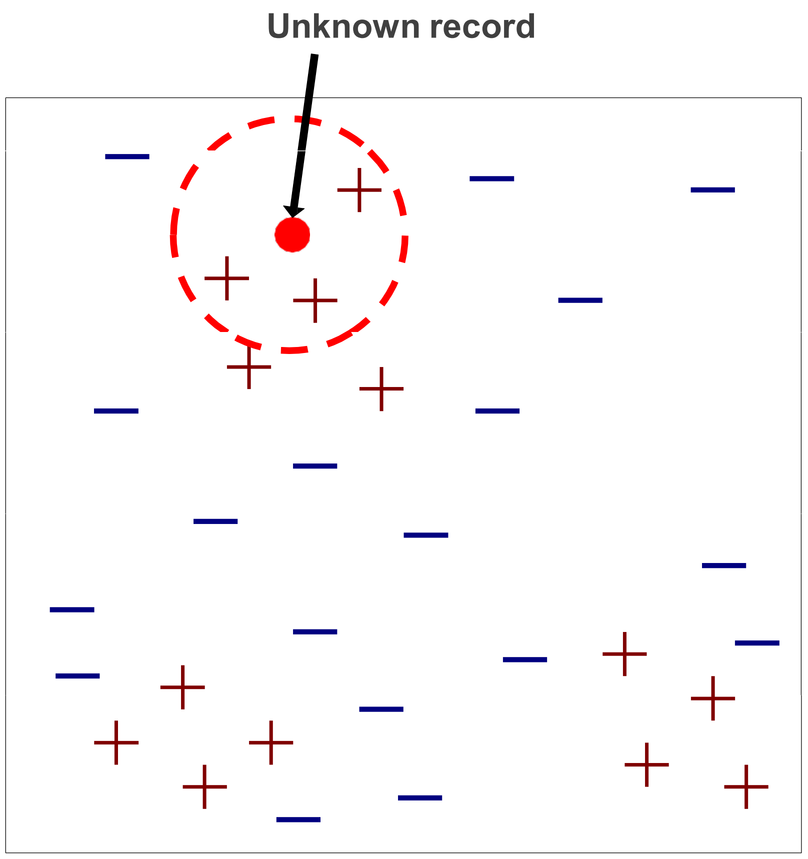
Nearest neighbor classifiers



Requires three inputs:

1. The set of stored samples
2. Distance metric to compute distance between samples
3. The value of *k*, the number of nearest neighbors to retrieve

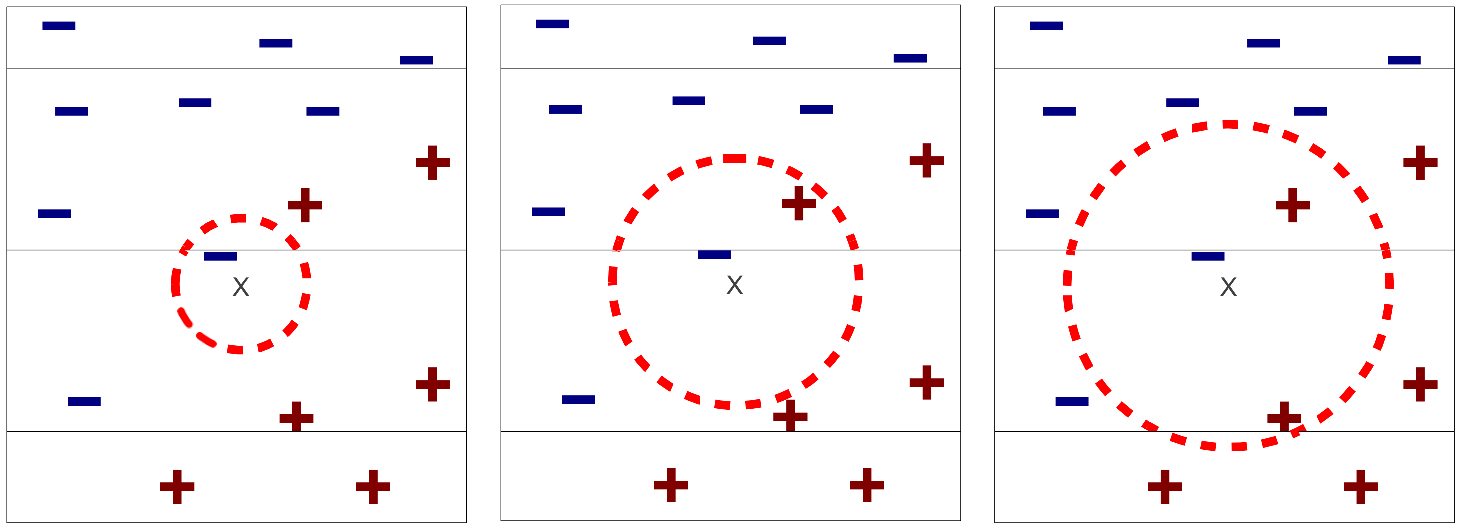
Nearest neighbor classifiers



To classify unknown record:

1. Compute distance to other training records
2. Identify *k* nearest neighbors
3. Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of nearest neighbor

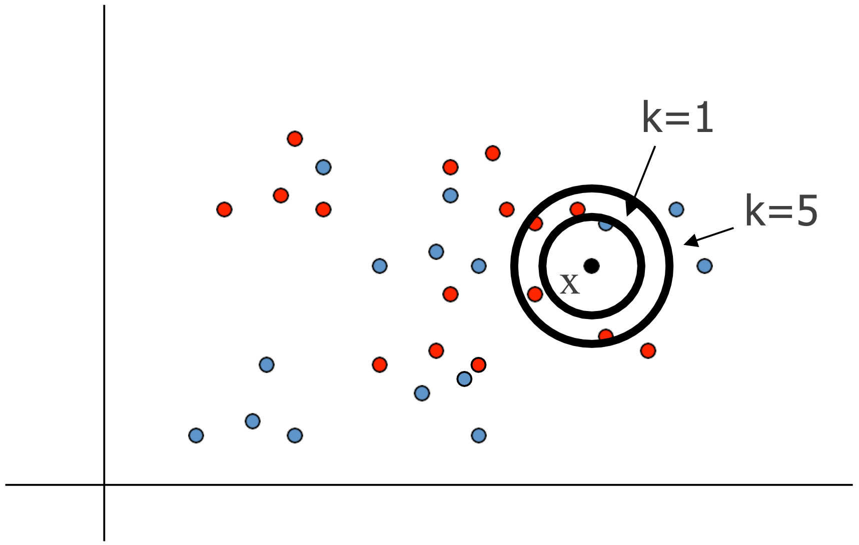


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

k-nearest neighbors of a sample x are datapoints that have the *k* smallest distances to x

K--Nearest Neighbor Methods

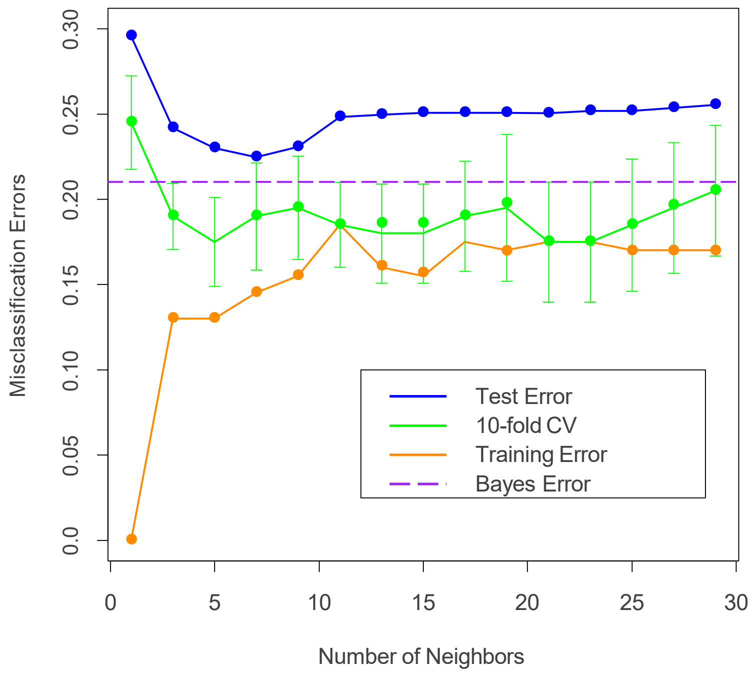
- To classify a new input vector x, examine the k--closest training data points to x and assign the object to the most frequently occurring class

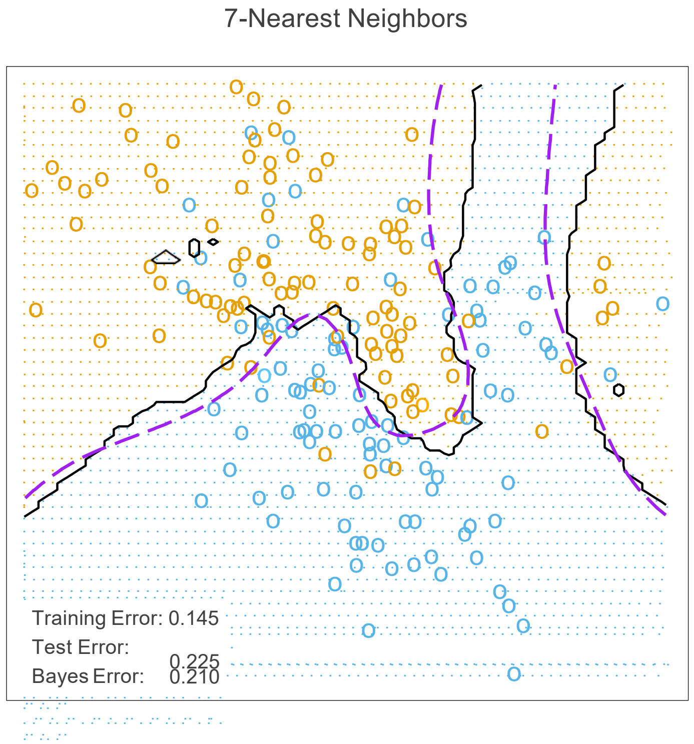


common values for k: 3, 5

Why?

Example results for k--NN





[Figures from Has8e and Tibshirani, Chapter 13]

Nearest Neighbor

When to Consider

- Instance map to points in R^n
- Less than 20 attributes per instance
- Lots of training data

Advantages

- Training is very fast
- Learn complex target functions
- Do not lose information

Disadvantages

- Slow at query
- Easily fooled by irrelevant attributes

Issues

- Distance measure
  - Most common: Euclidean
- Choosing k
  - Increasing k reduces variance, increases bias
- For high--dimensional space, problem that the nearest neighbor may not be very close at all!
- Memory--based technique. Must make a pass through the data for each classification. This can be prohibitive for large data sets.

Nearest Neighbors

Training example in Euclidean space: $\mathbf{x} \in R^d$

Idea: The value of the target function for a new query is estimated from the known value(s) of the nearest training example(s)

Distance typically defined to be Euclidean:

$$\left\|x^{(a)} - x^{(b)}\right\|_2 = \sqrt{\sum_{j=1}^{d} (x_j^{(a)} - x_j^{(b)})^2}$$

Algorithm:

1. Find example $(\mathbf{x}^*, t^*)$ (from the stored training set) closest to the test instance $\mathbf{x}$. That is:

$$\mathbf{x}^* = \underset{\mathbf{x}^{(i)} \in \text{train. set}}{\operatorname{argmin}} \ \text{distance}(\mathbf{x}^{(i)}, \mathbf{x})$$

2. Output $y = t^*$

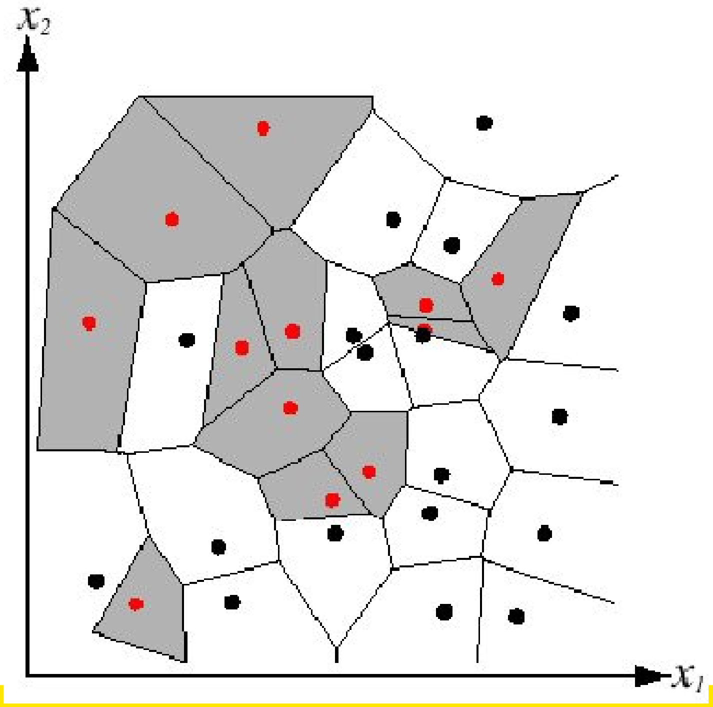
Note: we don't really need to compute the square root. Why?

Nearest Neighbors: Decision Boundaries

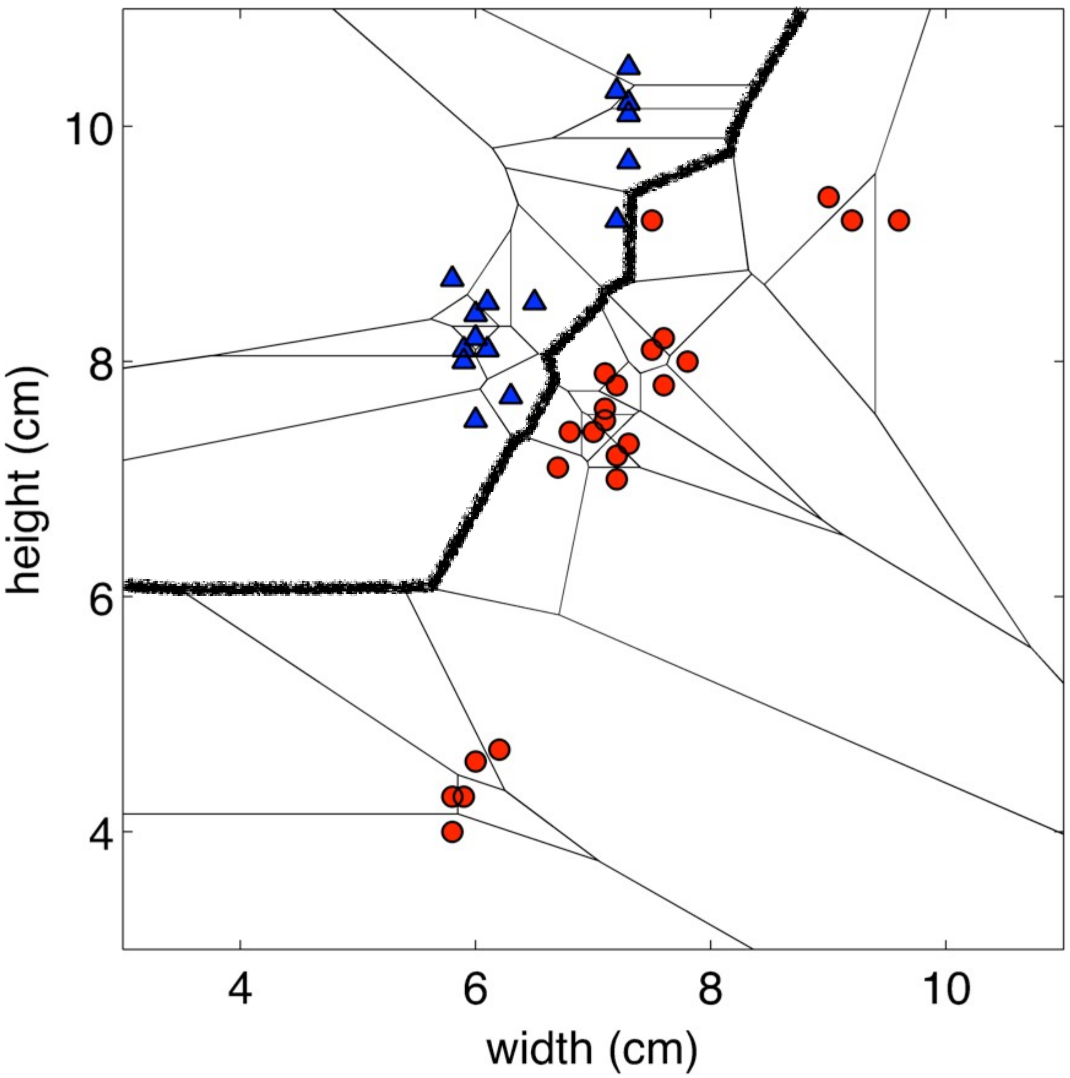
Nearest neighbor algorithm does not explicitly compute decision boundaries, but these can be inferred

Decision boundaries: Voronoi diagram visualization

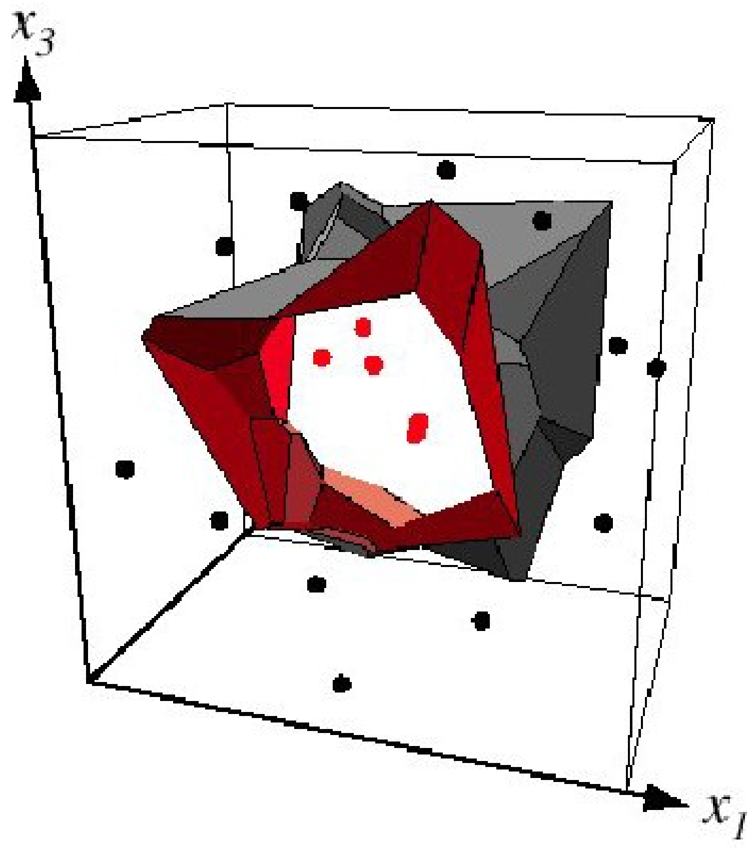
- show how input space divided into classes
- each line segment is equidistant between two points of opposite classes



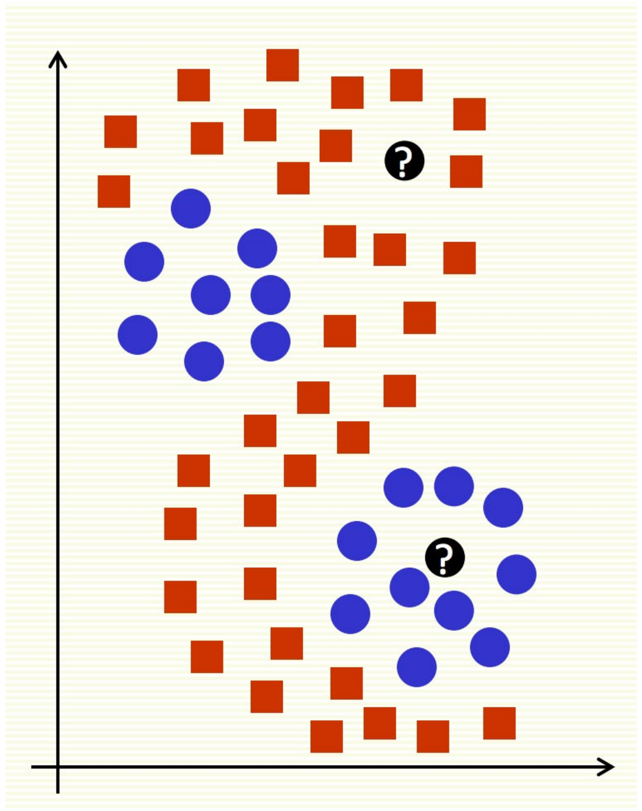
Example: 2D decision boundary



Example: 3D decision boundary

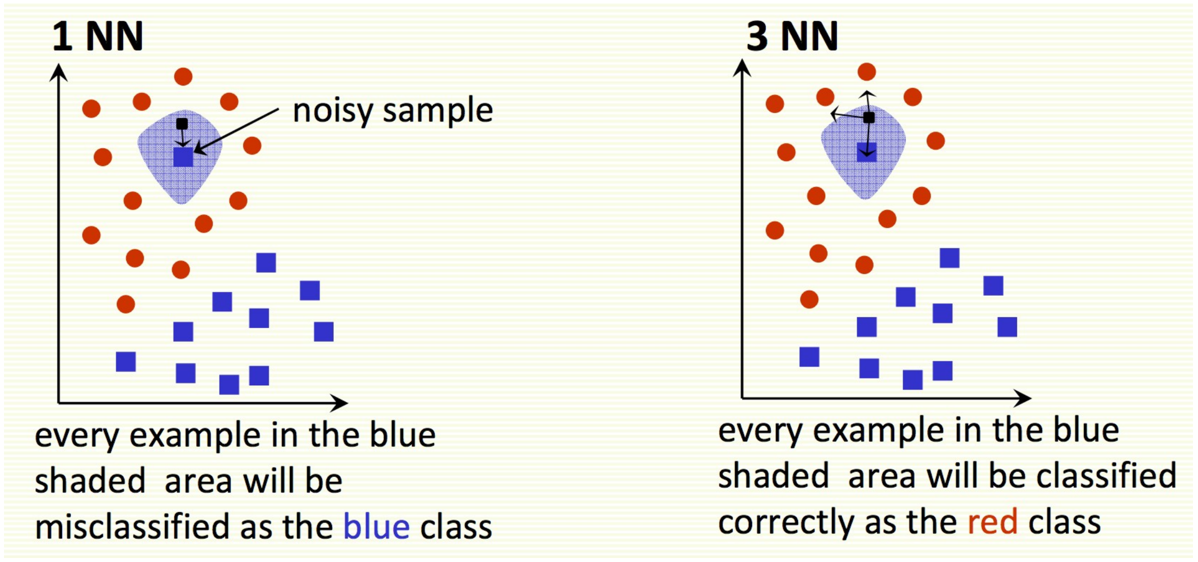


Nearest Neighbor approaches can work with multi-modal data



[Slide credit: O. Veksler]

k-Nearest Neighbors


1 NN
noisy sample
every example in the blue shaded area will be misclassified as the blue class
3 NN
every example in the blue shaded area will be classified correctly as the red class

Nearest neighbors sensitive to mis-labeled data ("class noise"). Solution?

Smooth by having k nearest neighbors vote

k-Nearest Neighbors

[Pic by Olga Veksler]

Algorithm (kNN):

1. Find k examples $\{\mathbf{x}^{(i)}, t^{(i)}\}$ closest to the test instance $\mathbf{x}$
2. Classification output is majority class

$$y = \underset{t^{(z)}}{\operatorname{argmax}} \sum_{r=1}^{k} \delta(t^{(z)}, t^{(r)})$$

k-Nearest Neighbors

How do we choose k?

Larger k may lead to better performance

But if we set k too large we may end up looking at samples that are not neighbors (are far away from the query)

We can use cross-validation to find k

Rule of thumb is $k < sqrt(n)$, where n is the number of training examples

[Slide credit: O. Veksler]

k-Nearest Neighbors: Issues & Remedies

If some attributes (coordinates of **x**) have larger ranges, they are treated as more important

- normalize scale
    - Simple option: Linearly scale the range of each feature to be, e.g., in range [0,1]
    - Linearly scale each dimension to have 0 mean and variance 1 (compute mean μ and variance σ^2 for an attribute x_j and scale: $(x_j - m)/\sigma$)
- be careful: sometimes scale matters

k-Nearest Neighbors: Issues & Remedies

- Irrelevant, correlated attributes add noise to distance measure
  - eliminate some attributes
  - or vary and possibly adapt weight of attributes
- Non-metric attributes (symbols)
  - Hamming distance

k-Nearest Neighbors: Issues (Complexity) & Remedies

Expensive at test time: To find one nearest neighbor of a query point **x**, we must compute the distance to all N training examples. Complexity: $O(kdN)$ for kNN

- Use subset of dimensions
- Pre-sort training examples into fast data structures (e.g., kd-trees)
- Compute only an approximate distance (e.g., LSH)
- Remove redundant data (e.g., condensing)

[Slide credit: David Claus]

k-Nearest Neighbors: Issues (Complexity) & Remedies

Storage Requirements: Must store all training data

- Remove redundant data (e.g., condensing)
- Pre-sorting often increases the storage requirements

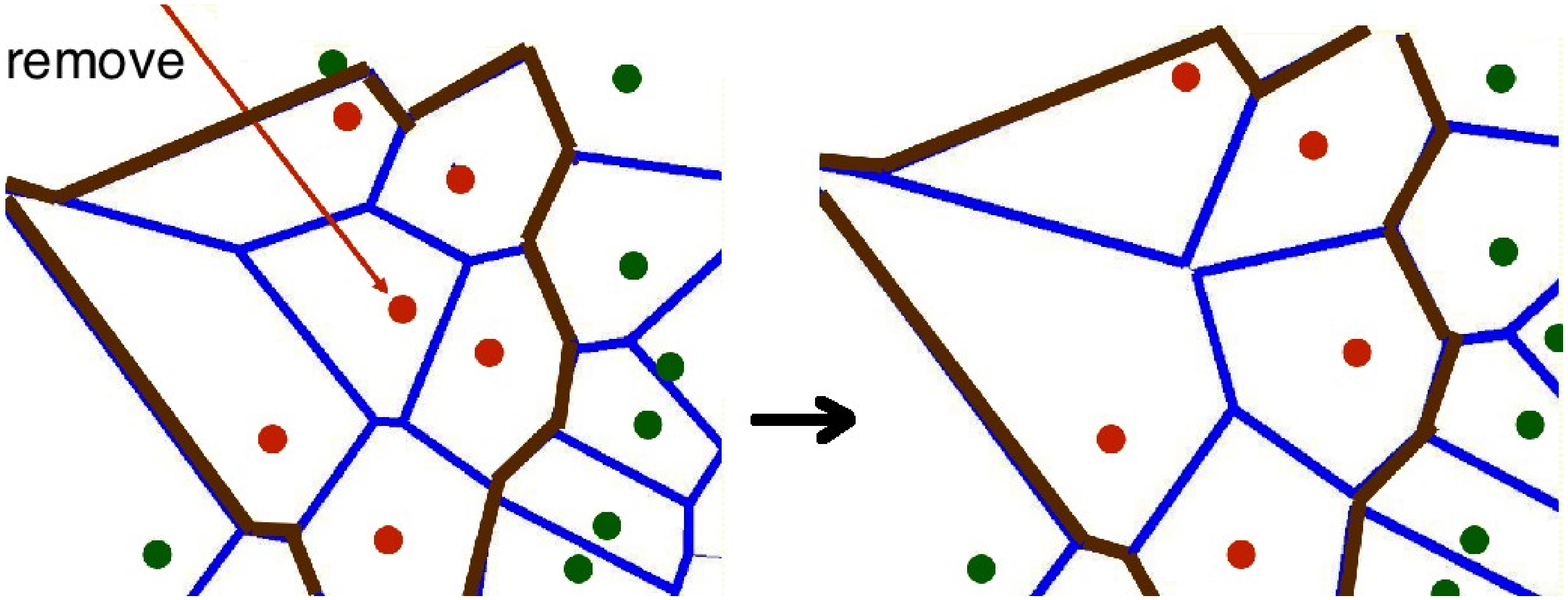
High Dimensional Data: "Curse of Dimensionality"

- Required amount of training data increases exponentially with dimension
- Computational cost also increases

[Slide credit: David Claus]

k-Nearest Neighbors Remedies: Remove Redundancy

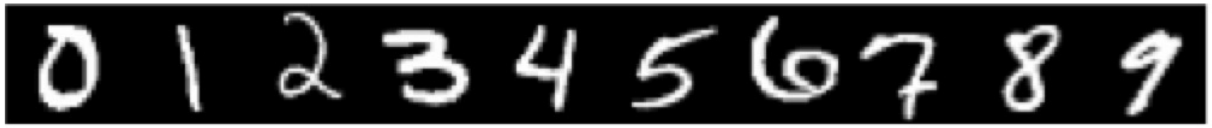
If all Voronoi neighbors have the same class, a sample is useless, remove it



[Slide credit: O. Veksler]

Example: Digit Classification

Decent performance when lots of data



- Yann LeCunn – MNIST Digit Recognition
  - Handwritten digits
  - 28x28 pixel images: *d* = 784
  - 60,000 training samples
  - 10,000 test samples
- Nearest neighbour is competitive

	Test Error Rate (%)
Linear classifier (1-layer NN)	12.0
K-nearest-neighbors, Euclidean	5.0
K-nearest-neighbors, Euclidean, deskewed	2.4
K-NN, Tangent Distance, 16x16	1.1
K-NN, shape context matching	0.67
1000 RBF + linear classifier	3.6
SVM deg 4 polynomial	1.1
2-layer NN, 300 hidden units	4.7
2-layer NN, 300 HU, [deskewing]	1.6
LeNet-5, [distortions]	0.8
Boosted LeNet-4, [distortions]	0.7

Fun Example: Where on Earth is this Photo From?

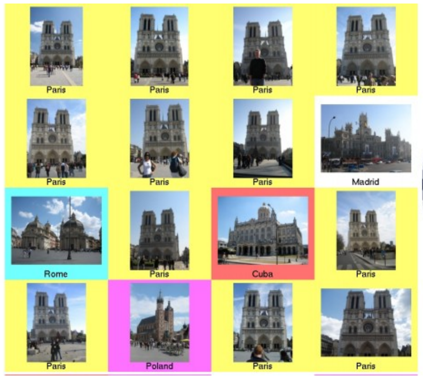
Problem: Where (e.g., which country or GPS location) was this picture taken?

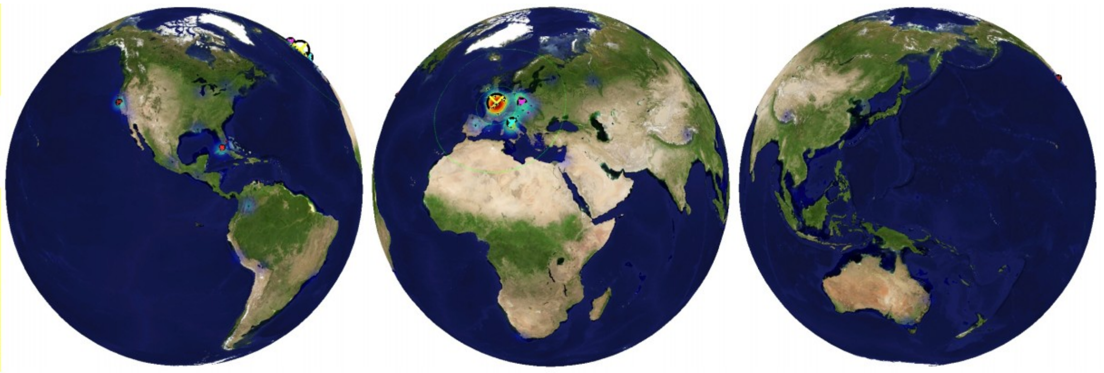
[Paper: James Hays, Alexei A. Efros. im2gps: estimating geographic information from a single image. CVPR'08. Project page: http://graphics.cs.cmu.edu/projects/im2gps/]

Fun Example:
Where on Earth is this Photo From?

Problem: Where (e.g., which country or GPS location) was this picture taken?

- Get 6M images from Flickr with GPs info (dense sampling across world)
- Represent each image with meaningful features
- Do kNN!



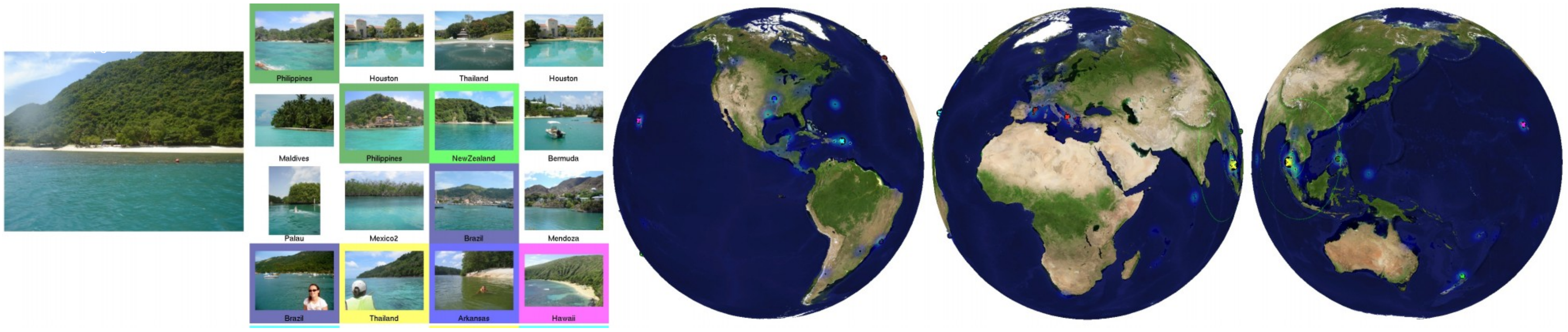


[Paper: James Hays, Alexei A. Efros. im2gps: estimating geographic information from a single image. CVPR'08. Project page: http://graphics.cs.cmu.edu/projects/im2gps/]

Fun Example: Where on Earth is this Photo From?

Problem: Where (eg, which country or GPS location) was this picture taken?

- Get 6M images from Flickr with gps info (dense sampling across world)
- Represent each image with meaningful features
- Do kNN (large *k* better, they use *k* = 120)!



[Paper: James Hays, Alexei A. Efros. im2gps: estimating geographic information from a single image. CVPR'08. Project page: http://graphics.cs.cmu.edu/projects/im2gps/]

Example: PEBLS

- PEBLS: Parallel Examplar-Based Learning System (Cost & Salzberg)
  - Works with both continuous and nominal features
    - For nominal features, distance between two nominal values is computed using modified value difference metric (MVDM)
  - Each sample is assigned a weight factor
  - Number of nearest neighbor, *k* = 1

Example: PEBLS

Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
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

Distance between nominal attribute values:

d(Single,Married)

= | 2/4 – 0/4 | + | 2/4 – 4/4 | = 1

d(Single,Divorced)

= | 2/4 – 1/2 | + | 2/4 – 1/2 | = 0

d(Married,Divorced)

= | 0/4 – 1/2 | + | 4/4 – 1/2 | = 1

d(Refund=Yes,Refund=No)

= | 0/3 – 3/7 | + | 3/3 – 4/7 | = 6/7

Class	Marital Status		
	Single	Married	Divorced
Yes	2	0	1
No	2	4	1

Class	Refund	
	Yes	No
Yes	0	3
No	3	4

$$d(V_1,V_2) = \sum_i \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$$

Example: PEBLS

Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
X	Yes	Single	125K	**No**
Y	No	Married	100K	**No**

Distance between record X and record Y:

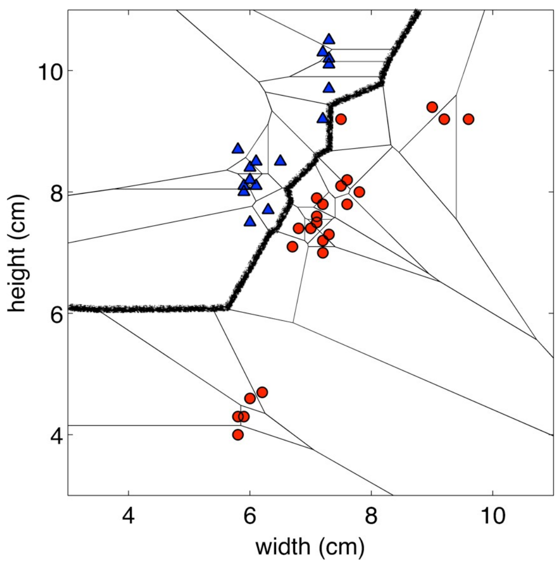
$$\Delta(X,Y) = w_X w_Y \sum_{i=1}^{d} d(X_i,Y_i)^2$$

where: $$w_X = \frac{\text{Number of times X is used for prediction}}{\text{Number of times X predicts correctly}}$$

$w_X \cong 1$ if X makes accurate prediction most of the time

$w_X > 1$ if X is not reliable for making predictions

K-NN Summary



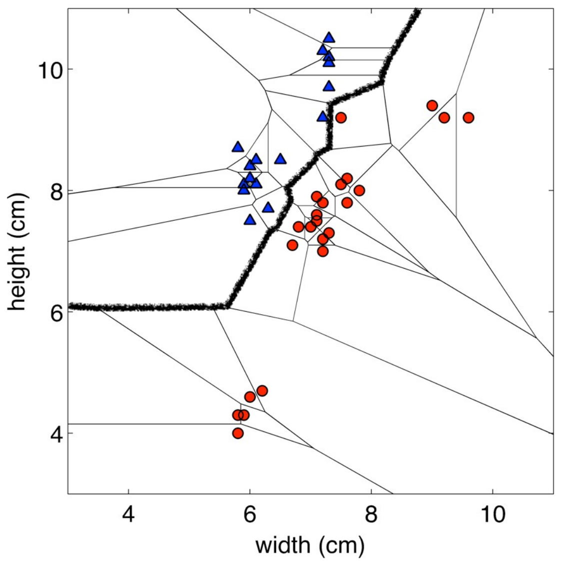
Naturally forms complex decision boundaries; adapts to data density

If we have lots of samples, kNN typically works well

Problems:

- Sensitive to class noise
- Sensitive to scales of attributes
- Distances are less meaningful in high dimensions
- Scales linearly with number of examples

K-NN Summary



Naturally forms complex decision boundaries; adapts to data density

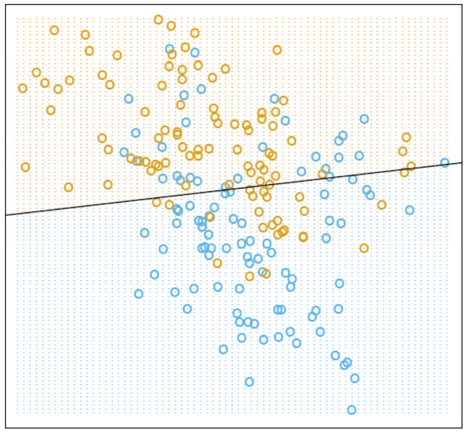
If we have lots of samples, kNN typically works well

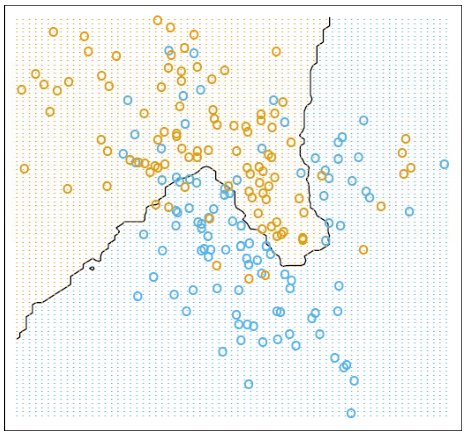
Problems:

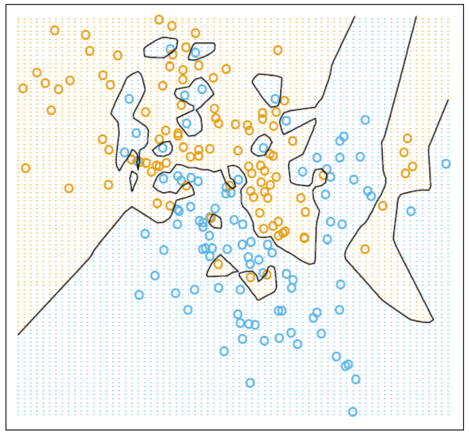
- Sensitive to class noise
- Sensitive to scales of attributes
- Distances are less meaningful in high dimensions
- Scales linearly with number of examples

Inductive Bias: What kind of decision boundaries do we expect to find?

Decision boundaries in global vs. local models


linear regression


15-nearest neighbor


1-nearest neighbor

- global
- stable
- can be inaccurate

- local
- accurate
- unstable

What ultimately matters: ***GENERALIZATION***

KNN – summary

Non-parametric: makes no assumptions about the probability distribution the examples come from

Does not assume data is linearly separable

Derives decision rule directly from training data

"Lazy learning":

- During learning little "work" is done by the algorithm: the training instances are simply stored in memory in some efficient manner.
- During prediction the test instance is compared to the training instances, the neighborhood is calculated, and the majority label assigned

No information discarded: "exceptional" and low frequency training instances are available for prediction

kNN Demo

http://vision.stanford.edu/teaching/cs231n-demos/knn/