



CS 175: Project in Artificial Intelligence

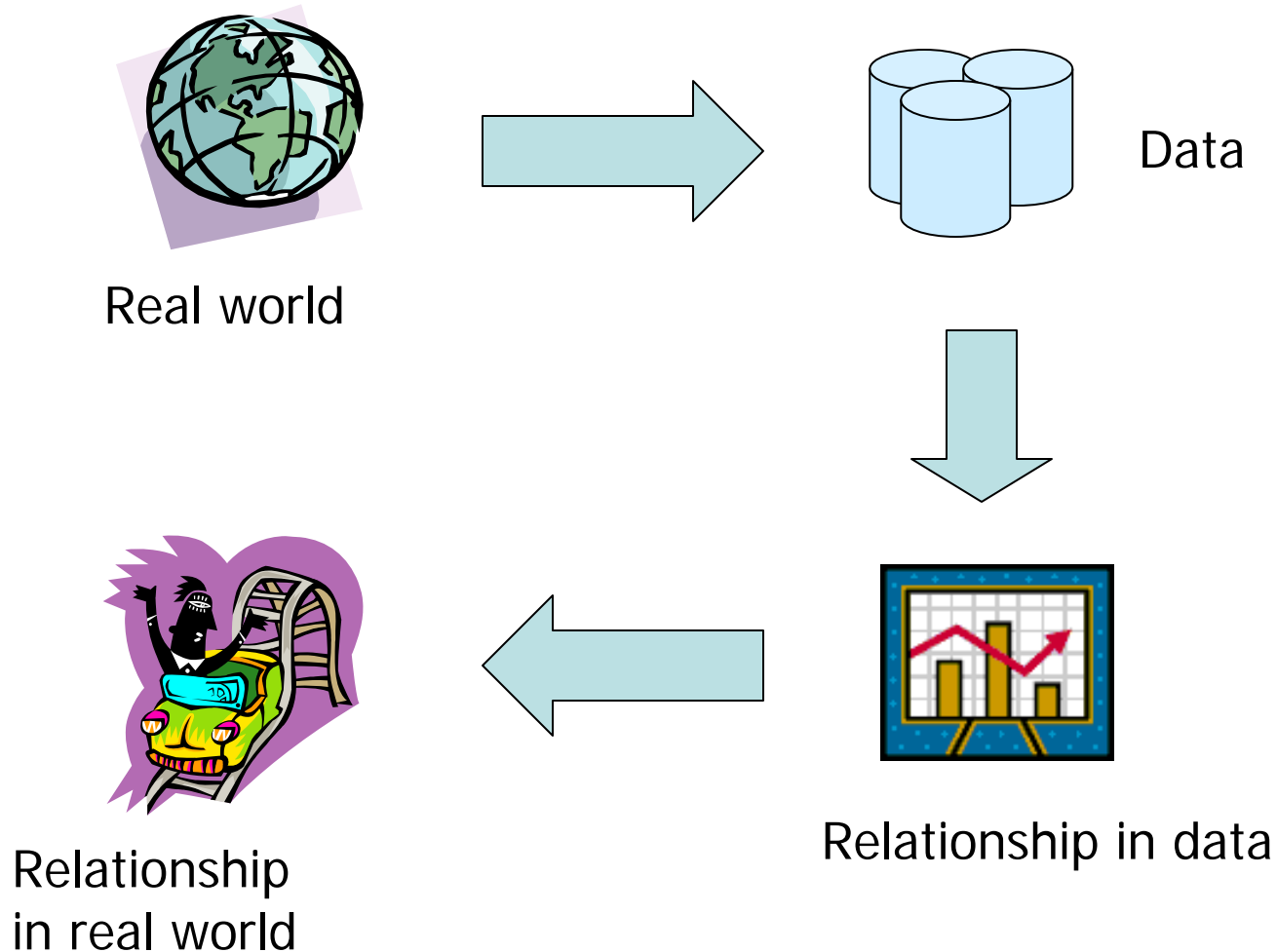
Slides 2: Measurement and Data,
and Classification

Topic 3: Measurement and Data

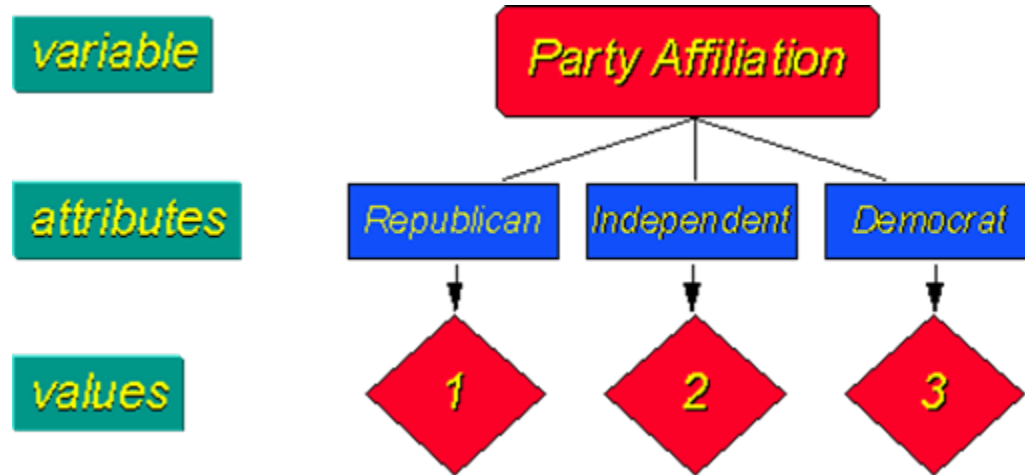
Slides taken from Prof. Smyth
(with slight modifications)

Measurement

Mapping domain entities to symbolic representations



Nominal or Categorical Variables



Numerical values here have no semantic meaning, just indices
No ordering implied

Another example:

Jersey numbers in basketball; a player with number 30 is not more of anything than a player with number 15 .

Ordinal Measurements

ordinal measurement -

attributes can be rank-ordered.

Distances between attributes do not have any meaning.

e.g., on a survey you might code Educational Attainment as
0=less than H.S.; 1=some H.S.; 2=H.S. degree; 3=some college;
4=college degree; 5=post college.

In this measure, higher numbers mean *more* education.

But is distance from 0 to 1 same as 3 to 4? No.

Interval distances are not interpretable in an ordinal measure.

Interval Measurements

interval measurement –

distance between attributes *does* have meaning.

e.g., when we measure temperature (in Fahrenheit), the distance from 30-40 is same as distance from 70-80.

The interval between values is interpretable.

Average makes sense.

However ratios don't: 80 degrees is not twice as hot as 40 degrees

Ratio Measurements

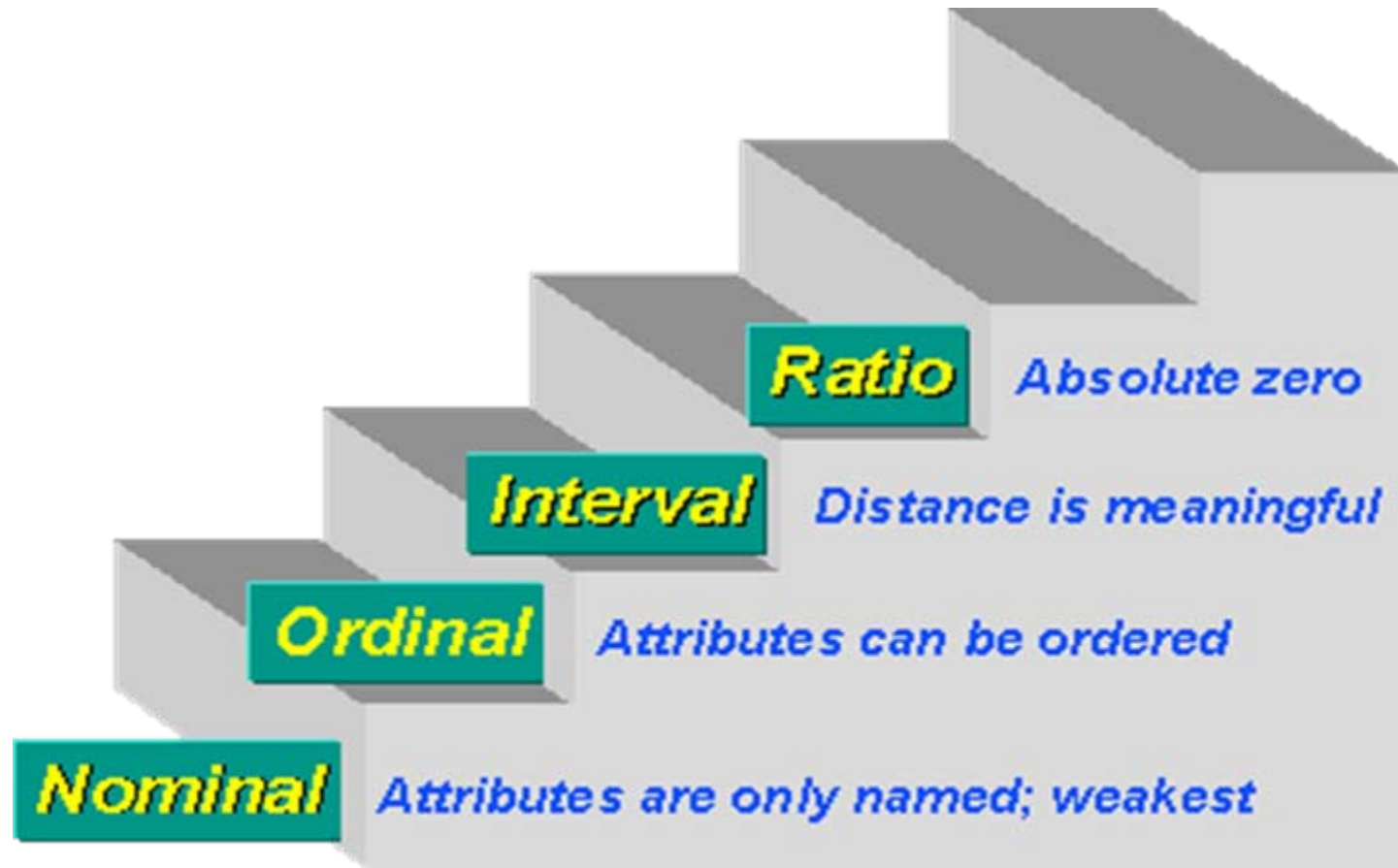
ratio measurement –
an absolute zero that is meaningful.

This means that you can construct a meaningful fraction (or ratio) with a ratio variable.

e.g., weight is a ratio variable.

Typically "count" variables are ratio, for example, the number of clients in past six months.

Hierarchy of Measurements



Scales and Transformations

scale	Legal transforms	example
Nominal/ Categorical	Any one-one mapping	Hair color, employment
ordinal	Any order preserving transform	Severity, preference
interval	Multiply by constant, add a constant	Temperature, calendar time
ratio	Multiply by constant	Weight, income

Why is this important?

- As we will see....
 - Many models require data to be represented in a specific form
 - e.g., real-valued vectors
 - Linear regression, neural networks, support vector machines, etc
 - These models implicitly assume interval-scale data (at least)
 - What do we do with non-real valued inputs?
 - Nominal with M values:
 - Not appropriate to “map” to 1 to M (maps to an interval scale)
 - Why? $w_1 \times \text{employment_type} + w_2 \times \text{city_name}$
 - Could use M binary “indicator” variables
 - » But what if M is very large? (e.g., cluster into groups of values)
 - Ordinal?

Mixed data

- Many real-world data sets have multiple types of variables,
 - e.g., demographic data sets for marketing
 - Nominal: employment type, ethnic group
 - Ordinal: education level
 - Interval/Ratio: income, age
- Unfortunately, many data analysis algorithms are suited to only one type of data (e.g., interval)
- Exception: decision trees
 - Trees operate by subgrouping variable values at internal nodes
 - Can operate effectively on binary, nominal, ordinal, interval
 - We will see more details later.....

DISTANCE MEASURES

Vector data and distance matrices

- Data may be available as n “vectors” each d -dimensional
- Or “data” itself could be provided as an $n \times n$ matrix of similarities or distances
 - E.g., as similarity judgements in psychological experiments
 - Less common

Distance Measures

- Many data mining techniques are based on similarity or distance measures between objects i and j .
- Metric: $d(i,j)$ is a metric iff
 1. $d(i,j) \geq 0$ for all i, j and $d(i,j) = 0$ iff $i = j$
 2. $d(i,j) = d(j,i)$ for all i and j
 3. $d(i,j) \leq d(i,k) + d(k,i)$ for all i, j and k

Distance

- Notation: n objects with d measurements, $i = 1, \dots, n$
 $x(i) = (x_1(i), x_2(i), \dots, x_d(i))$

- Most common distance metric is *Euclidean* distance:

$$d_E(i, j) = \left(\sum_{k=1}^d (x_k(i) - x_k(j))^2 \right)^{\frac{1}{2}}$$

- Makes sense in the case where the different measurements are commensurate; each variable measured in the same units.
- If the measurements are different, say length in different units, Euclidean distance is not necessarily meaningful.

Standardization

When variables are not commensurate, we can standardize them by dividing by scaling, e.g., divide by the sample standard deviation of each variable

The estimate for the standard deviation of x_k :

$$\hat{\sigma}_k = \left(\frac{1}{n} \sum_{i=1}^n (x_k(i) - \bar{x}_k)^2 \right)^{\frac{1}{2}}$$

where \bar{x}_k is the sample mean:

$$\bar{x}_k = \frac{1}{n} \sum_{i=1}^n x_k(i)$$

Weighted Euclidean distance

If we have some idea of the relative importance of each variable, we can weight them:

$$d_{WE}(i, j) = \left(\sum_{k=1}^d w_k (x_k(i) - x_k(j))^2 \right)^{\frac{1}{2}}$$

Extensions

- L_p metric:

$$dist(i, j) = \left(\sum_{k=1}^d |x_k(i) - x_k(j)|^p \right)^{\frac{1}{p}} \quad \text{where } p \geq 1$$

- Manhattan, city block or L_1 metric:

$$dist(i, j) = \sum_{k=1}^d |x_k(i) - x_k(j)|$$

- L_∞

$$dist(i, j) = \max_k |x_k(i) - x_k(j)|$$

Additive Distances

- Each variable contributes independently to the measure of distance.
- May not always be appropriate...

object i



diameter(i)

height(i)

height₂(i)

⋮

height₁₀₀(i)



object j

diameter(j)

height(j)

height₂(j)

⋮

height₁₀₀(j)

Dependence among Variables

- Covariance and correlation measure linear dependence
- Assume we have two variables or attributes X and Y and n objects taking on values $x(1), \dots, x(n)$ and $y(1), \dots, y(n)$. The sample covariance of X and Y is:

$$\text{Cov}(X, Y) = \frac{1}{n} \sum_{i=1}^n (x(i) - \bar{x})(y(i) - \bar{y})$$

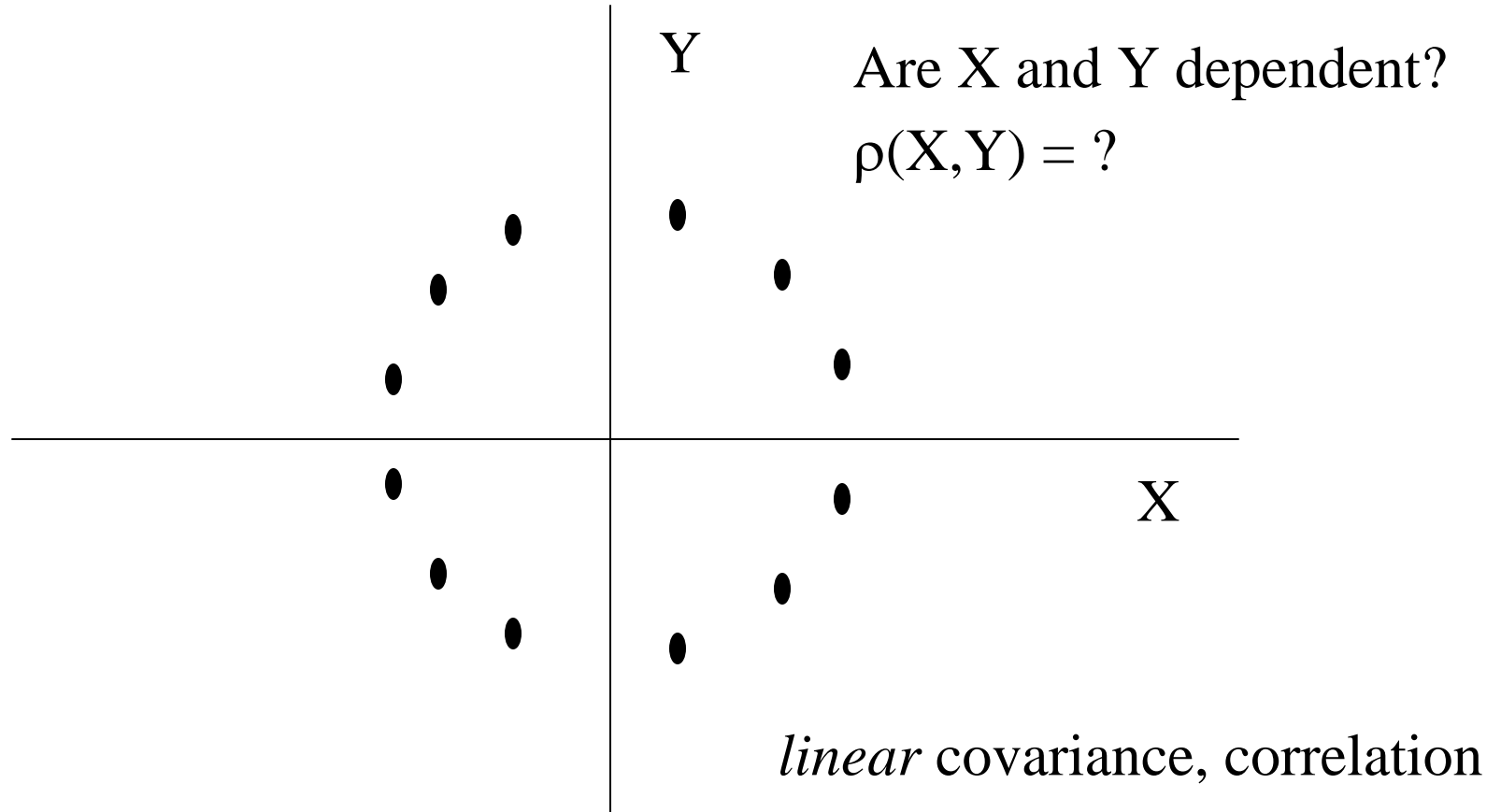
- The covariance is a measure of how X and Y vary together.
 - it will be large and positive if large values of X are associated with large values of Y , and small $X \Rightarrow$ small Y

Correlation coefficient

- Covariance depends on ranges of X and Y
- Standardize by dividing by standard deviation

$$\rho(X, Y) = \frac{\sum_{i=1}^n (x(i) - \bar{x})(y(i) - \bar{y})}{\left(\sum_{i=1}^n (x(i) - \bar{x})^2 \sum_{i=1}^n (y(i) - \bar{y})^2 \right)^{\frac{1}{2}}}$$

What about...



A simple data set

Data

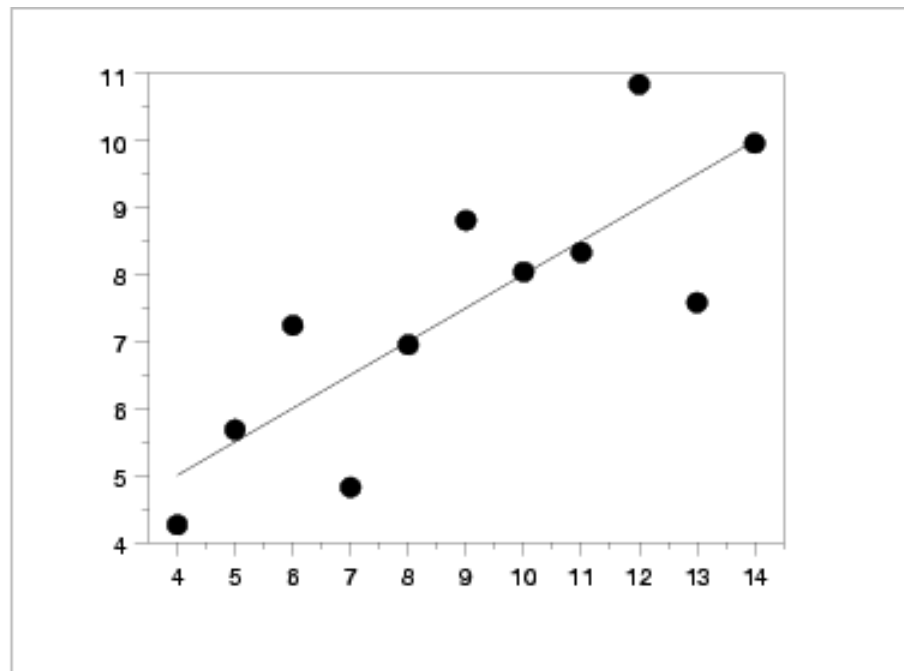
X	10.00	8.00	13.00	9.00	11.00	14.00	6.00	4.00	12.00	7.00	5.00
Y	8.04	6.95	7.58	8.81	8.33	9.96	7.24	4.26	10.84	4.82	5.68

Anscombe, Francis (1973), *Graphs in Statistical Analysis*,
The American Statistician, pp. 195-199.

A simple data set

Data

X	10.00	8.00	13.00	9.00	11.00	14.00	6.00	4.00	12.00	7.00	5.00
Y	8.04	6.95	7.58	8.81	8.33	9.96	7.24	4.26	10.84	4.82	5.68



A simple data set

Data

X	10.00	8.00	13.00	9.00	11.00	14.00	6.00	4.00	12.00	7.00	5.00
Y	8.04	6.95	7.58	8.81	8.33	9.96	7.24	4.26	10.84	4.82	5.68

Summary Statistics

$N = 11$

Mean of $X = 9.0$

Mean of $Y = 7.5$

Intercept = 3

Slope = 0.5

Residual standard deviation = 1.237

Correlation = 0.816

3 more data sets

X2	Y2	X3	Y3	X4	Y4
10.00	9.14	10.00	7.46	8.00	6.58
8.00	8.14	8.00	6.77	8.00	5.76
13.00	8.74	13.00	12.74	8.00	7.71
9.00	8.77	9.00	7.11	8.00	8.84
11.00	9.26	11.00	7.81	8.00	8.47
14.00	8.10	14.00	8.84	8.00	7.04
6.00	6.13	6.00	6.08	8.00	5.25
4.00	3.10	4.00	5.39	19.00	12.50
12.00	9.13	12.00	8.15	8.00	5.56
7.00	7.26	7.00	6.42	8.00	7.91
5.00	4.74	5.00	5.73	8.00	6.89

Summary Statistics

Summary Statistics of Data Set 2

$N = 11$

Mean of $X = 9.0$

Mean of $Y = 7.5$

Intercept = 3

Slope = 0.5

Residual standard deviation = 1.237

Correlation = 0.816

Summary Statistics

Summary Statistics of Data Set 2

$N = 11$

Mean of $X = 9.0$

Mean of $Y = 7.5$

Intercept = 3

Slope = 0.5

Residual standard deviation = 1.237

Correlation = 0.816

Summary Statistics of Data Set 3

$N = 11$

Mean of $X = 9.0$

Mean of $Y = 7.5$

Intercept = 3

Slope = 0.5

Residual standard deviation = 1.237

Correlation = 0.816

Summary Statistics of Data Set 4

$N = 11$

Mean of $X = 9.0$

Mean of $Y = 7.5$

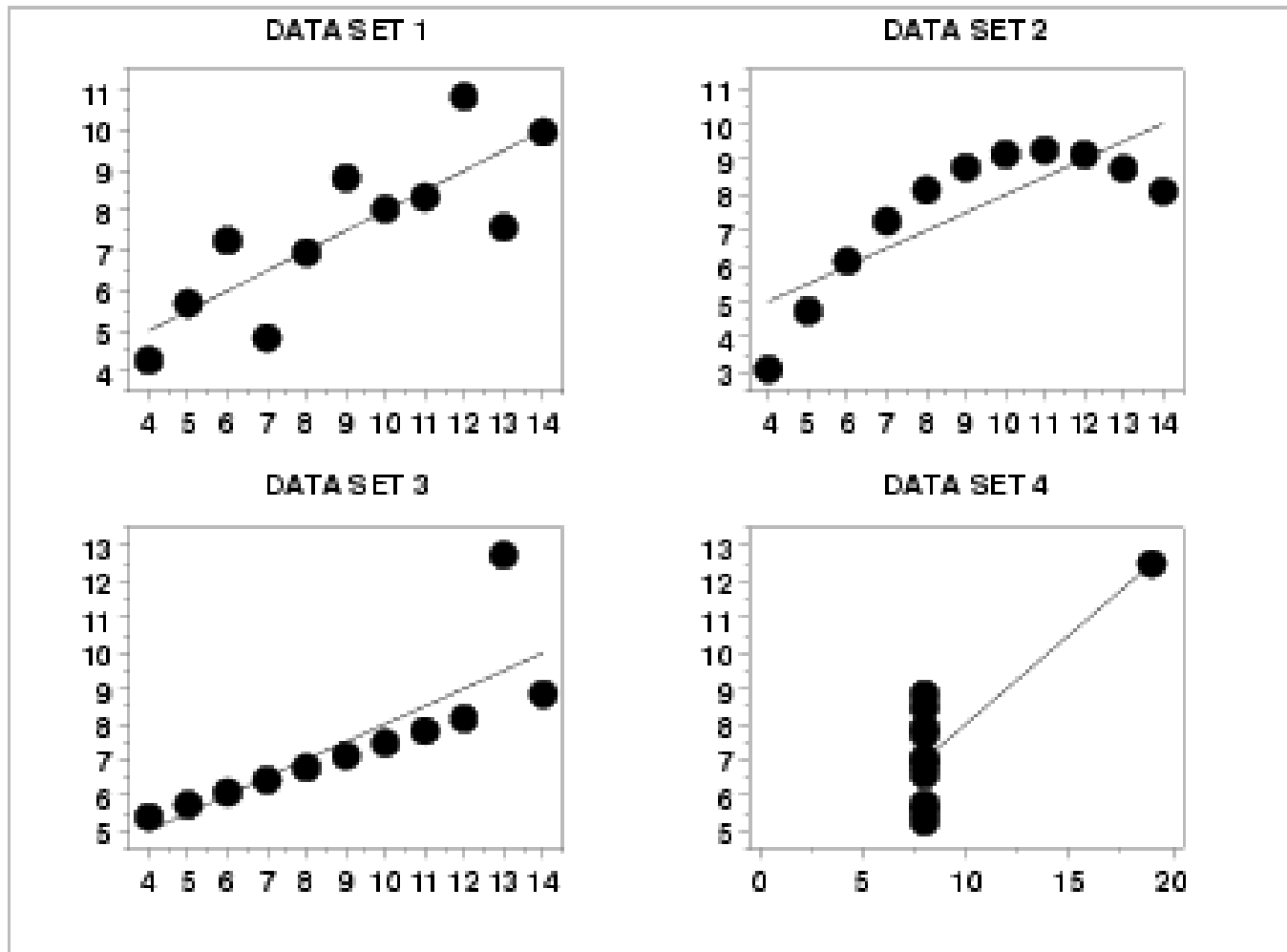
Intercept = 3

Slope = 0.5

Residual standard deviation = 1.237

Correlation = 0.816

Visualization really helps!

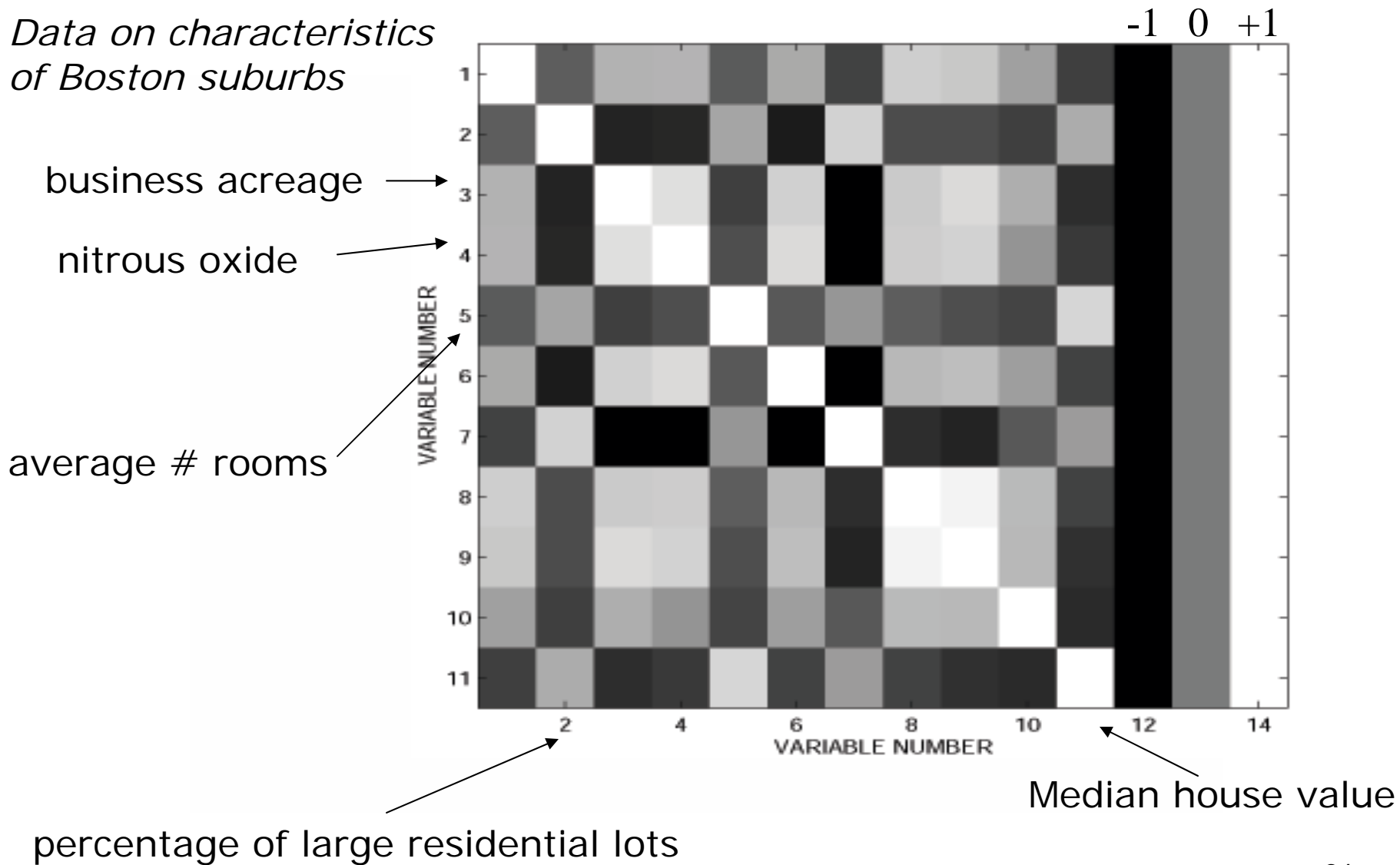


Sample Correlation/Covariance Matrices

- With d variables, we can compute d^2 pairwise correlations or covariances
 - > $d \times d$ correlation/covariance matrices
 - d diagonal elements = d variances, one per variable (for covariance)
 - $\text{covariance}(i,j) = \text{covariance}(j, i)$ - so matrix is symmetric
 - use symbol Σ to indicate covariance matrix
(closely associated with multivariate Gaussian density,
but can also be computed empirically for any data set, Gaussian or not)

Sample Correlation Matrix

*Data on characteristics
of Boston suburbs*



Mahalanobis distance (between data rows)

$$d_{MH}(x, y) = \left((x - y)^T \Sigma^{-1} (x - y) \right)^{\frac{1}{2}}$$

Evaluates to a
scalar distance

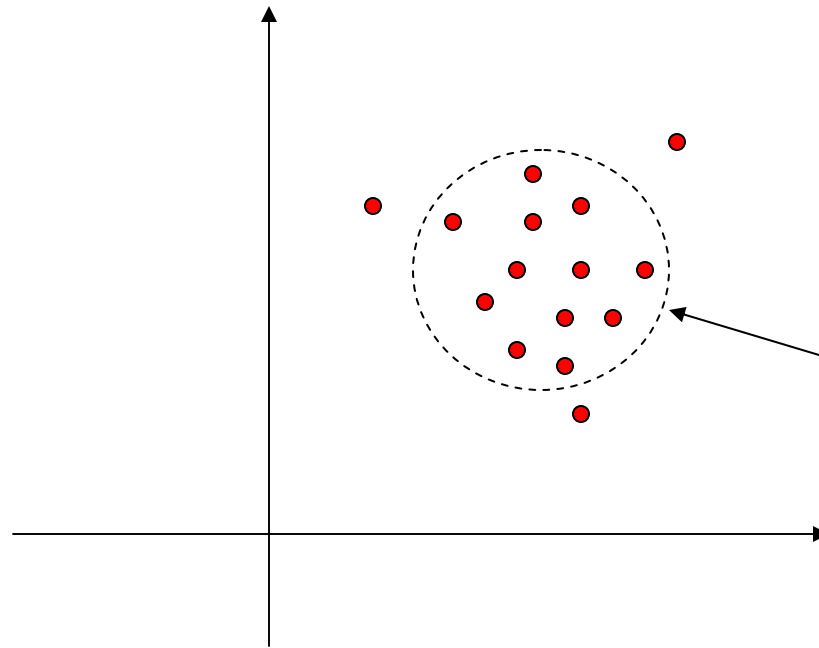
Vector difference in
d-dimensional space

Inverse covariance matrix

1. It automatically accounts for the scaling of the coordinate axes
2. It corrects for correlation between the different features

(assumes however that all pairs of variables data are at least approximately linearly correlated pairwise)

Example 1 of Mahalanobis distance

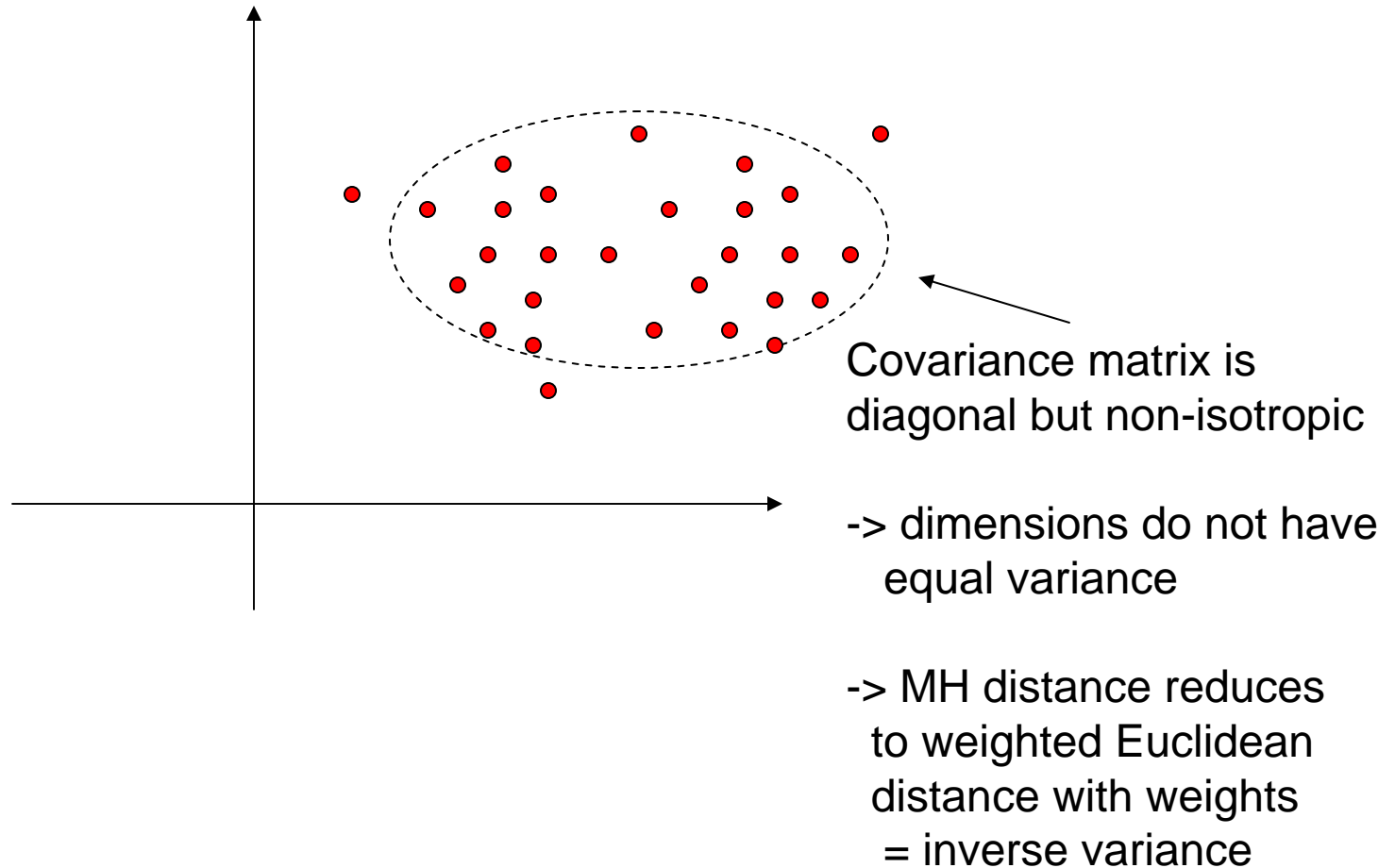


Covariance matrix is diagonal and isotropic

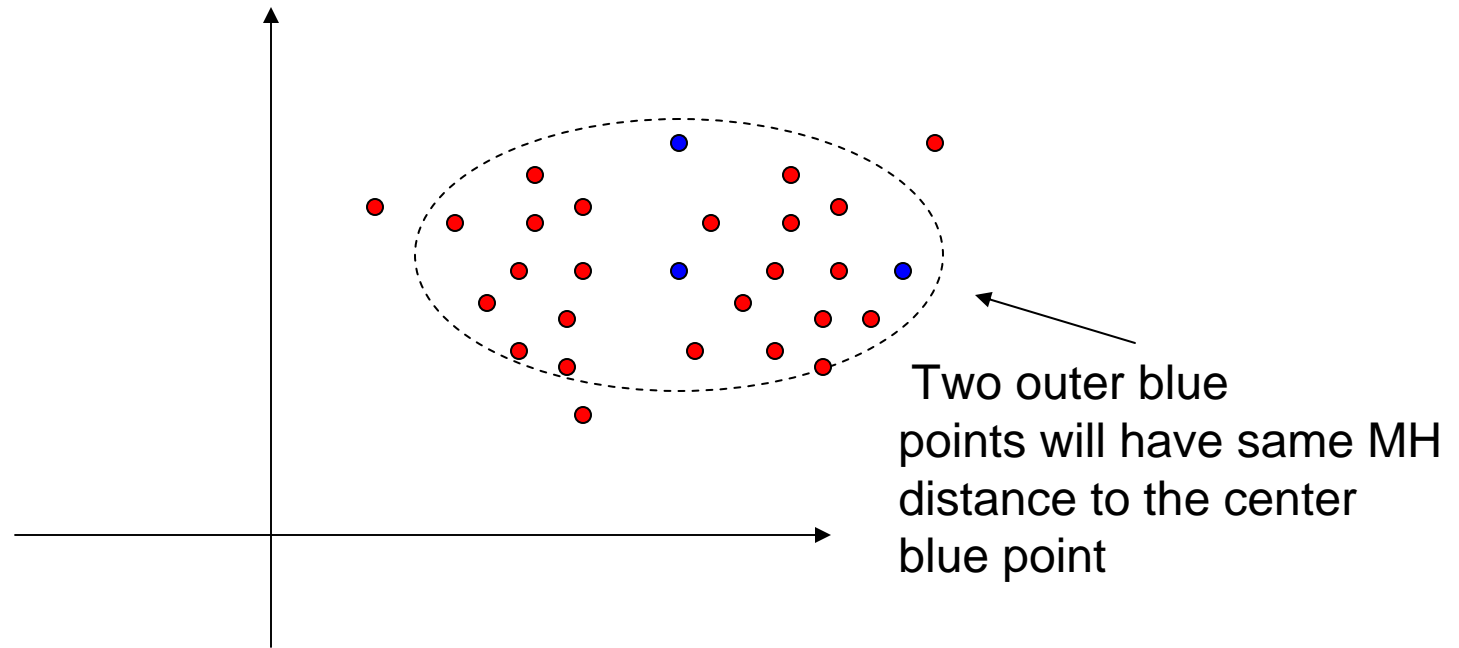
-> all dimensions have equal variance

-> MH distance reduces to Euclidean distance

Example 2 of Mahalanobis distance



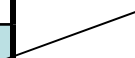
Example 2 of Mahalanobis distance



Binary Vectors

	j=1	j=0
i=1	n_{11}	n_{10}
i=0	n_{01}	n_{00}

Number of variables where item $j = 1$ and item $i = 0$



- matching coefficient

$$\frac{n_{11} + n_{00}}{n_{11} + n_{10} + n_{01} + n_{00}}$$

- Jaccard coefficient

$$\frac{n_{11}}{n_{11} + n_{10} + n_{01}}$$

Other distance metrics

- Categorical variables
 - Number of matches divided by number of dimensions
- Distances between strings of different lengths
 - e.g., “Patrick J. Smyth” and “Padhraic Smyth”
 - Edit distance
- Distances between images and waveforms
 - Shift-invariant, scale invariant
 - i.e., $d(x,y) = \min_{a,b} (|ax+b| - y)$

Transforming Data

- Duality between form of the data and the model
 - Useful to bring data onto a “natural scale”
 - Some variables are very skewed, e.g., income
- Common transforms: square root, reciprocal, logarithm, raising to a power
 - Often very useful when dealing with skewed real-world data
- Logit: transforms from 0 to 1 to real-line

$$\text{logit} (p) = \frac{p}{1 - p}$$

Topic 4: Machine Learning -- Classification

Slides taken from Prof. Welling
(with slight modifications)

Machine Learning

according to Google

- The ability of a machine to improve its performance based on previous results.
- The process by which computer systems can be directed to improve their performance over time.
- Subspecialty of artificial intelligence concerned with developing methods for software to learn from experience or extract knowledge from examples in a database.
- The ability of a program to learn from experience — that is, to modify its execution on the basis of newly acquired information.

Some Examples

- ZIP code recognition
- Loan application classification
- Signature recognition
- Voice recognition over phone
- Credit card fraud detection
- Spam filter
- Collaborative Filtering: suggesting other products at Amazon.com
- Marketing
- Stock market prediction
- Expert level chess and checkers systems
- biometric identification (fingerprints, DNA, iris scan, face)
- machine translation
- web-search
- document & information retrieval
- camera surveillance
- robosoccer
- and so on and so on...

Types of Learning

- **Supervised Learning**

- Labels are provided, there is a strong learning signal.
- e.g. classification, regression.

- **Semi-supervised Learning.**

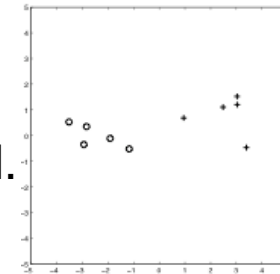
- Only part of the data have labels.
- e.g. a child growing up.

- **Reinforcement learning.**

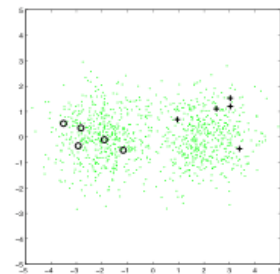
- The learning signal is a (scalar) reward and may come with a delay.
- e.g. trying to learn to play chess, a mouse in a maze.

- **Unsupervised learning**

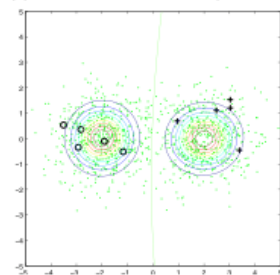
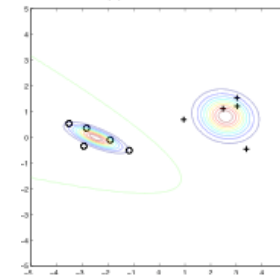
- There is no direct learning signal. We are simply trying to find structure in data.
- e.g. clustering, dimensionality reduction.



(a) labeled data



(b) labeled and unlabeled data (small dots)



Ingredients

- **Data:**
 - what kind of data do we have?
- **Prior assumptions:**
 - what do we know a priori about the problem?
- **Representation:**
 - How do we represent the data?
- **Model / Hypothesis space:**
 - What hypotheses are we willing to entertain to explain the data?
- **Feedback / learning signal:**
 - what kind of learning signal do we have (delayed, labels)?
- **Learning algorithm:**
 - How do we update the model (or set of hypothesis) from feedback?
- **Evaluation:**
 - How well did we do, should we change the model?

Supervised Learning I

Example: Imagine you want to classify



versus



Data: 100 car images and 200 truck images with labels what is what.

$$\{\vec{x}_i, y_i = 0\}, \quad i = 1, \dots, 100$$

$$\{\vec{x}_j, y_j = 1\}, \quad j = 1, \dots, 200$$

where x represents the greyscale of the image pixels and $y=0$ means “car” while $y=1$ means “truck”.

Task: Here is a new image:



car or truck?

1 nearest neighbors (1-NN)

(your first ML algorithm!)

Idea:

1. Find the picture in the database which is closest your query image.
2. Check its label.
3. Declare the class of your query image to be the same as that of the closest picture.



query

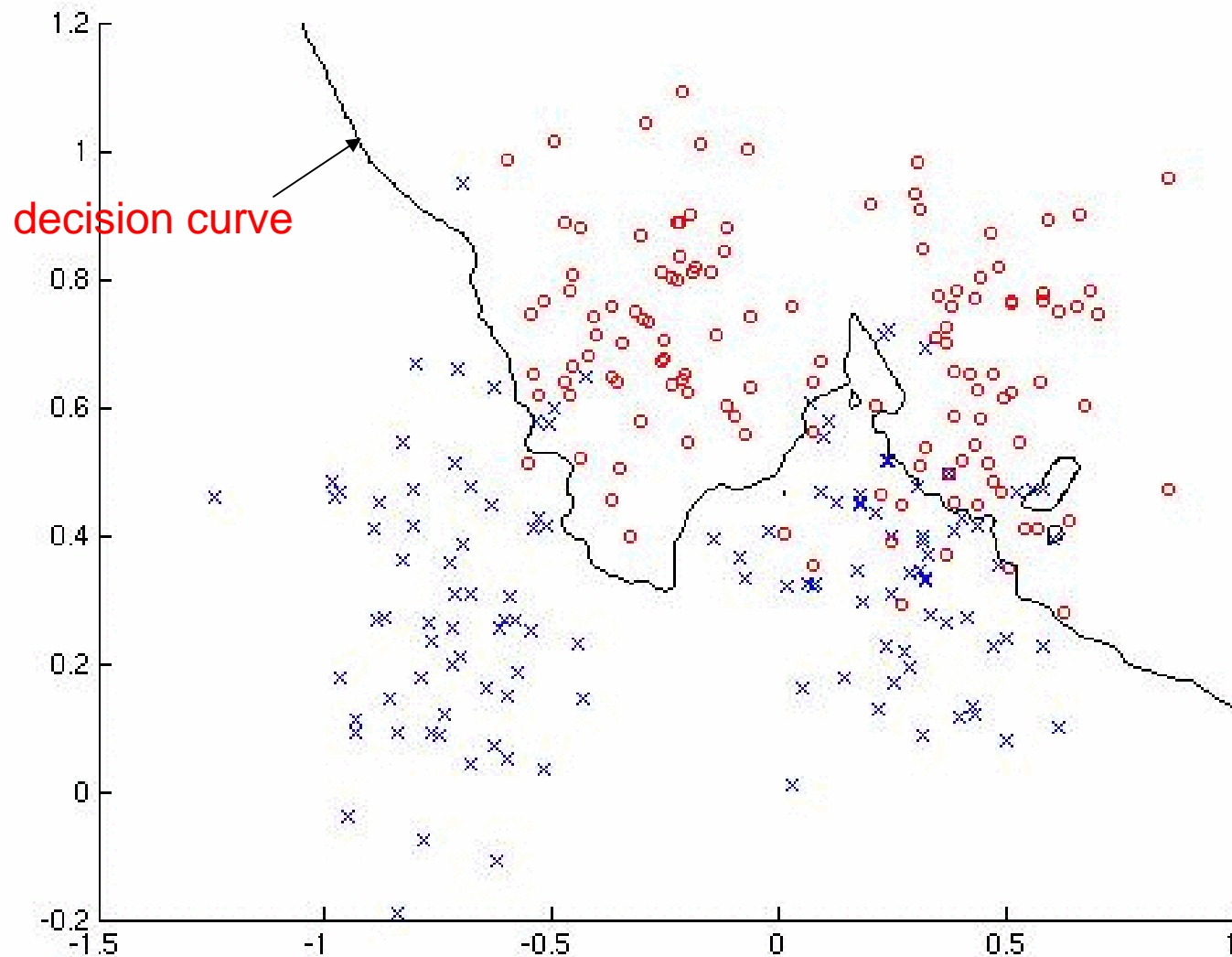


closest image

k-Nearest-Neighbors

- Find the k nearest neighbors
- Predict the majority vote
 - e.g. if 3 out of 5 neighbors are cars, then predict the test image to be a car too.
- More robust than 1-NN

kNN Decision Surface



Distance Metric

- How do we measure what it means to be “close”?
- Depending on the problem we should choose an appropriate distance metric.

Hamming distance:

$$D(\vec{x}_n, \vec{x}_m) = | \vec{x}_n - \vec{x}_m | \quad \{x = \text{discrete}\} ;$$

Scaled Euclidean Distance:

$$D(\vec{x}_n, \vec{x}_m) = (\vec{x}_n - \vec{x}_m)^T A (\vec{x}_n - \vec{x}_m) \quad \{x = \text{cont.}\} ;$$

Remarks on NN methods

- We only need to construct a classifier that works locally for each query. Hence: We don't need to construct a classifier everywhere in space.
- Classifying is done at query time. This can be computationally taxing at a time where you might want to be fast.
- Memory inefficient (you have to keep all data around).
- Curse of dimensionality: imagine many features are irrelevant / noisy
→ distances are always large.
- Very flexible, not many prior assumptions.
- k-NN variants robust against “bad examples”.

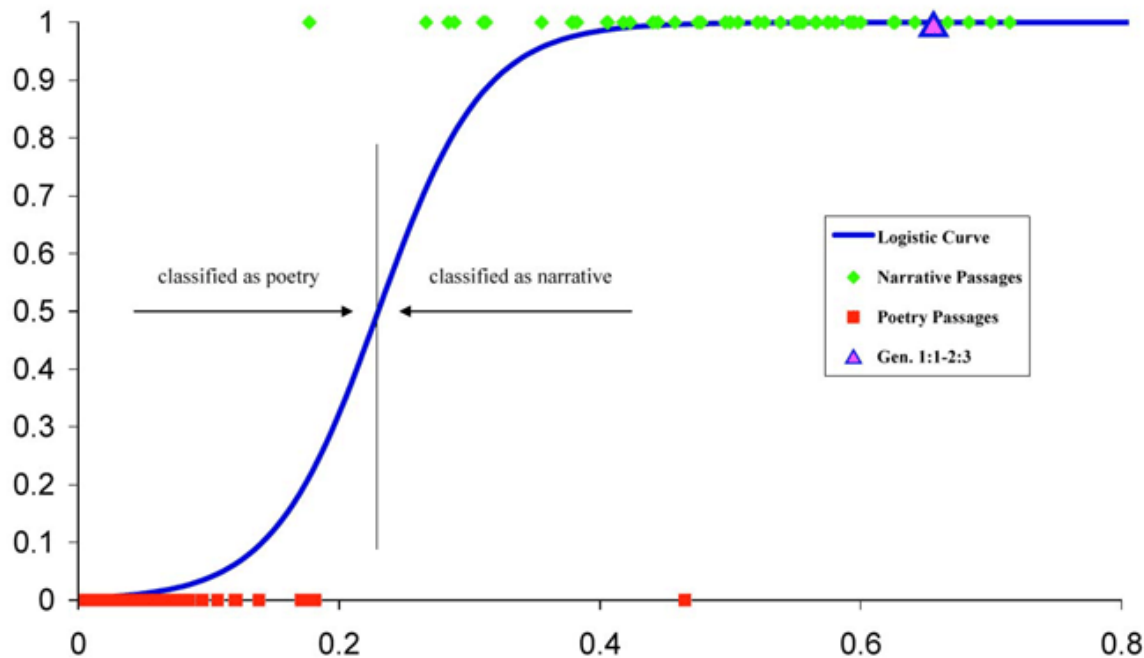
Non-parametric Methods

- Non-parametric methods keep all the data cases/examples in memory.
- A better name is: “instance-based” learning
- As the data-set grows, the complexity of the decision surface grows.
- Sometimes, non-parametric methods have some parameters to tune...
- Very few assumptions (we let the data speak).

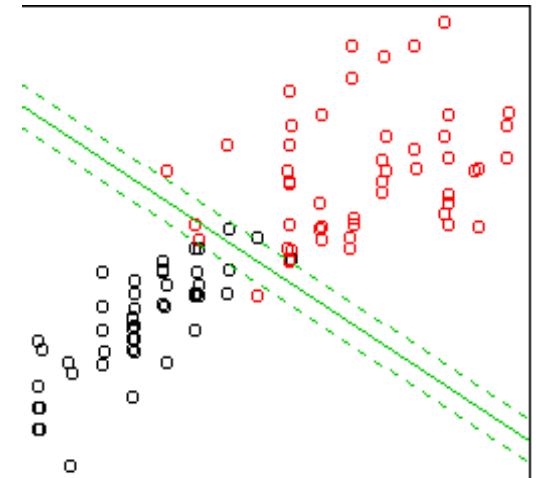
Logistic Regression / Perceptron

(your second ML algorithm!)

- Fits a soft decision boundary between the classes.

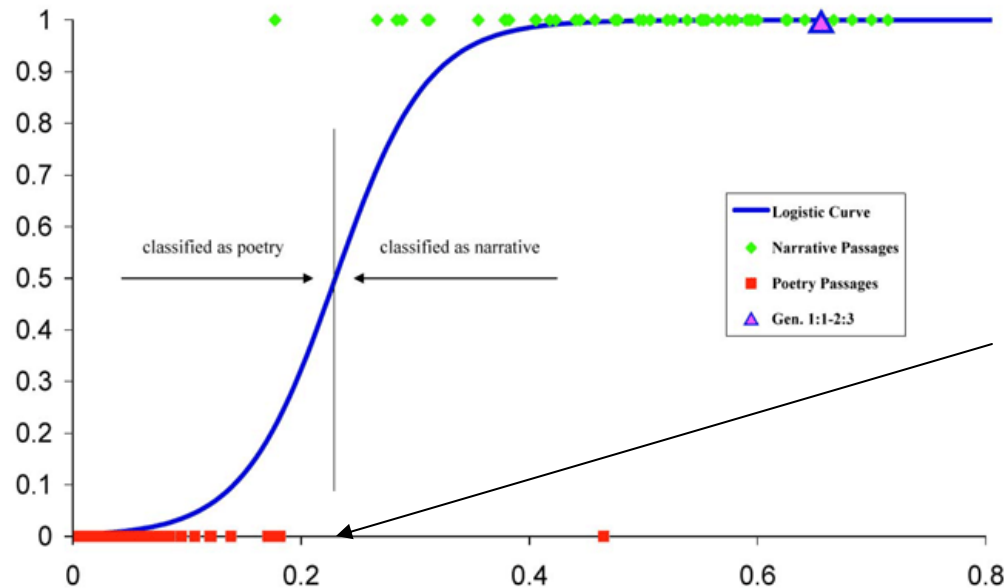
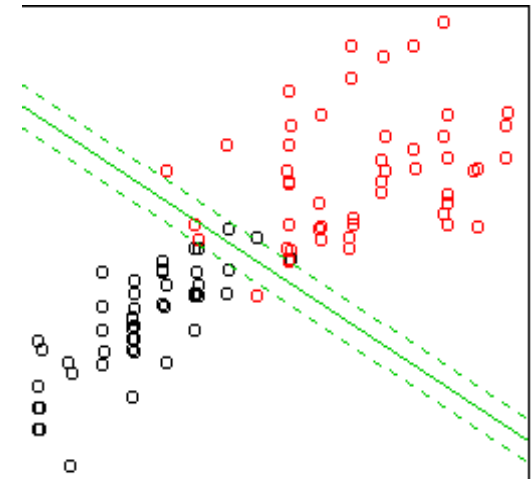


1 dimension



2 dimensions

The logit / sigmoid



$$h(X) = \frac{1}{1 + \exp[-(W^T X + b)]}$$

Determines the offset

Determines the angle and the steepness.

Objective

- We interpret $h(x)$ as the probability of classifying a data case as positive.
- We want to maximize the total probability of the data-vectors:

$$\mathcal{O} = \sum_{\substack{\text{positive} \\ \text{examples} \\ (y_n=1)}} \log[h(x_n)] + \sum_{\substack{\text{negative} \\ \text{examples} \\ (y_n=0)}} \log[1 - h(x_n)]$$

Algorithm in detail

- Repeat until convergence (gradient descent):

$$W \leftarrow W + \eta \frac{\partial O}{\partial W}$$

$$\frac{\partial O}{\partial W} = \sum_{\substack{\text{positive} \\ \text{examples} \\ (y_n=1)}} (1 - h(x_n)) x_n - \sum_{\substack{\text{negative} \\ \text{examples} \\ (y_n=0)}} h(x_n) x_n$$

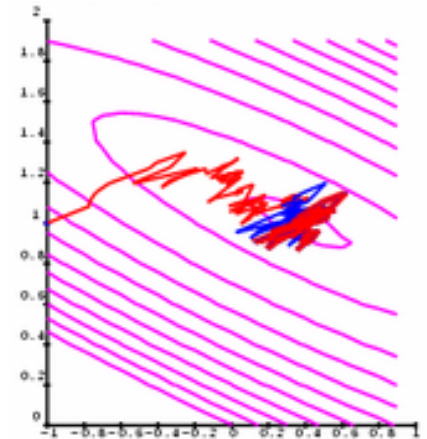
$$b \leftarrow b + \eta \frac{\partial O}{\partial b}$$

$$\frac{\partial O}{\partial b} = \sum_{\substack{\text{positive} \\ \text{examples} \\ (y_n=1)}} (1 - h(x_n)) - \sum_{\substack{\text{negative} \\ \text{examples} \\ (y_n=0)}} h(x_n)$$

“step size”
“learning rate”
typically small, e.g. 0.1

A Note on Stochastic GD


- For very large problems it is more efficient to compute the gradient using a small (random) subset of the data.
- For every new update you pick a new random subset.
- Towards convergence, you decrease the stepsize.
- Why is this more efficient?
 - The gradient is an average over many data-points.
 - If your parameters are very “bad”, every data-point will tell you to move in the same direction, so you need only a few data-points to find that direction.
 - Towards convergence you need all the data-points.
 - A small step-size effectively averages over many data-points.

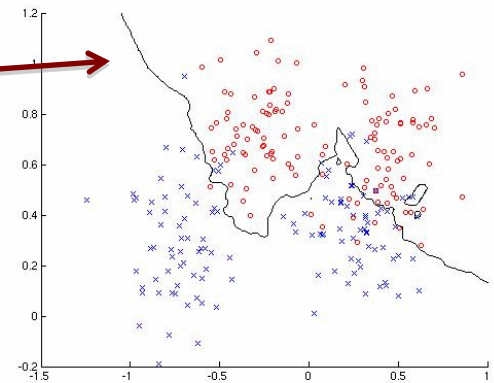


Parametric Methods

- Parametric methods fit a finite set of parameters to the data.
- Unlike NP methods, this implies a maximum complexity for the model.
- “Assumption heavy”: by choosing the parameterized model you impose your prior assumptions (this can be an advantage when you have sound assumptions!)
- Classifier is build off-line. Classification is fast at query time.
- Easy on memory: samples are summarized through model parameters.

Hypothesis Space

- An hypothesis $h: X \rightarrow [0,1]$ for a binary classifier is a function that maps all possible input values to either class 0 or class 1.
- E.g. for 1-NN the hypothesis $h(X)$ is given by: 
- The hypothesis space H , is the space of all hypotheses that you are willing to consider/search over.
- For instance, for logistic regression, H is given by all classifiers of the form (parameterized by W, b):



$$h(X;W,b) = \frac{1}{1 + \exp[-(W^T X + b)]}$$

Inductive Bias

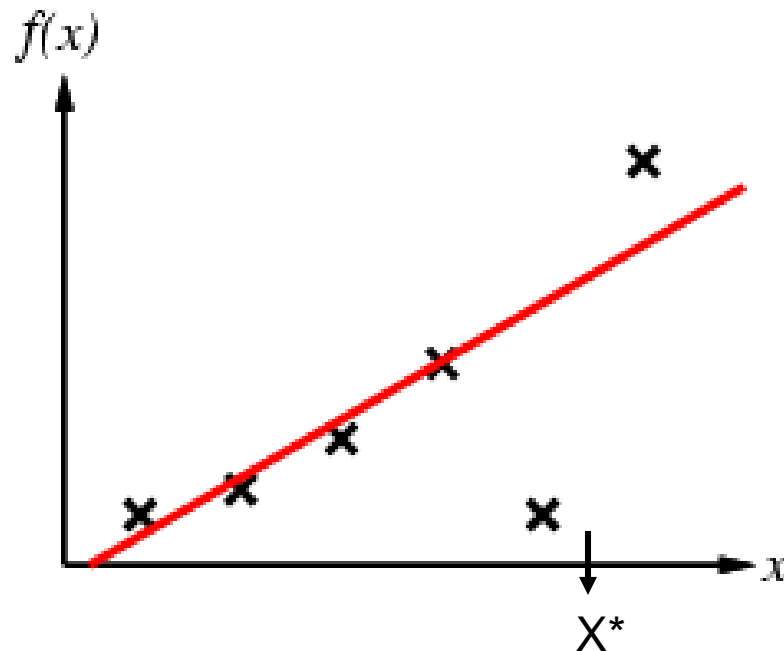
- The assumption one makes to generalize beyond the training data.
- Examples:
 - 1-NN: the label is the same as that of the closest training example.
 - LL: the classification function is a smooth function of the form:

$$h(X;W,b) = \frac{1}{1 + \exp[-(W^T X + b)]}$$

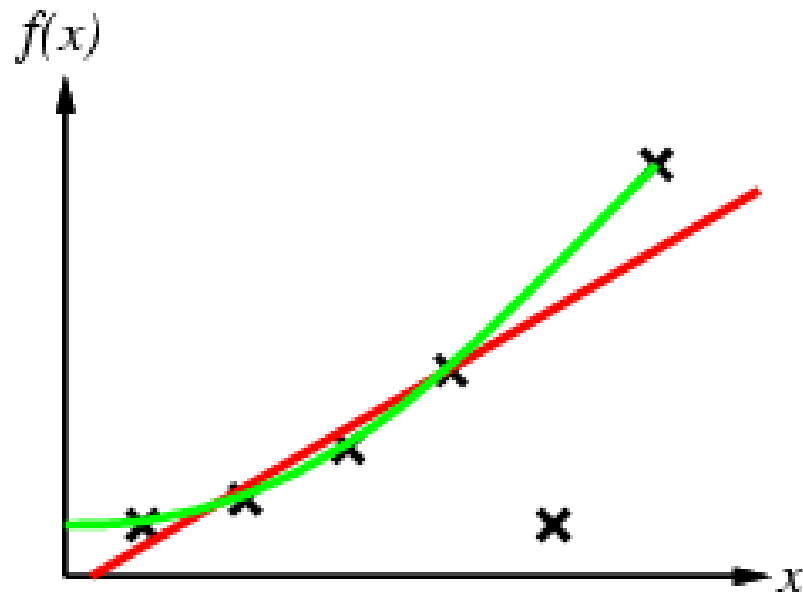
- Without inductive bias (i.e. without assumptions) there is no generalization possible! (you have not expressed preference for unseen data in any way).
- Learning is hence converting your prior assumptions + the data into a classifier for new data.

Generalization

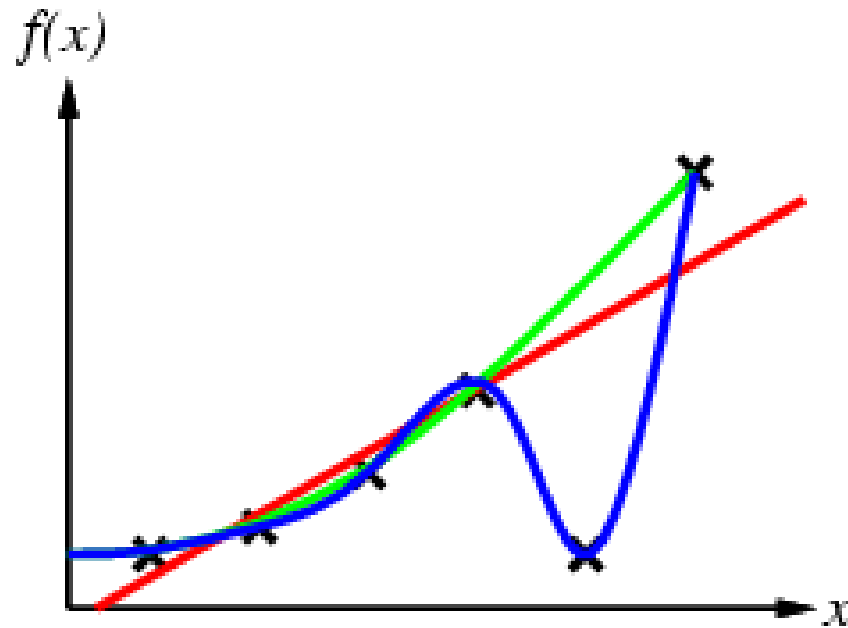
- Consider the following *regression* problem:
- Predict the real value on the y-axis from the real value on the x-axis.
- You are given 6 examples: $\{X_i, Y_i\}$.
- What is the y-value for a new query point X^* ?



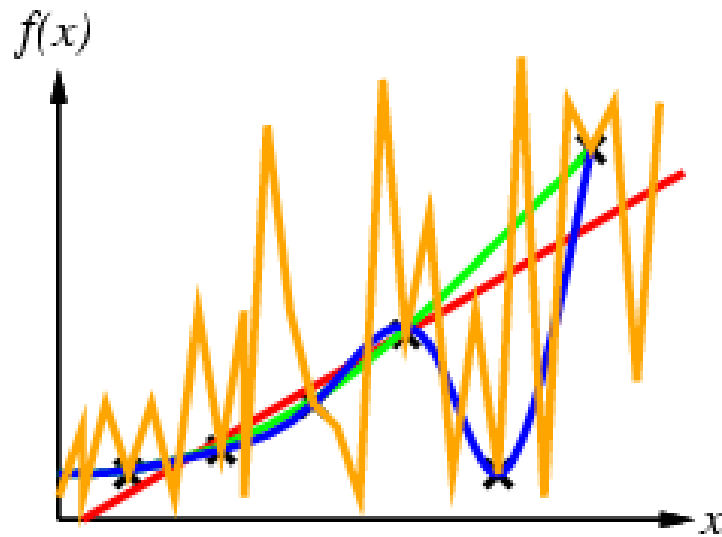
Generalization



Generalization

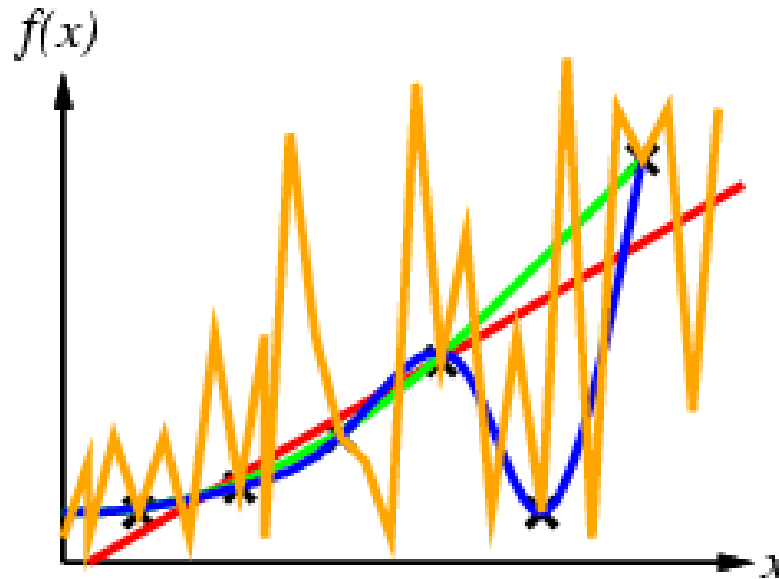


Generalization



which curve is best?

Generalization



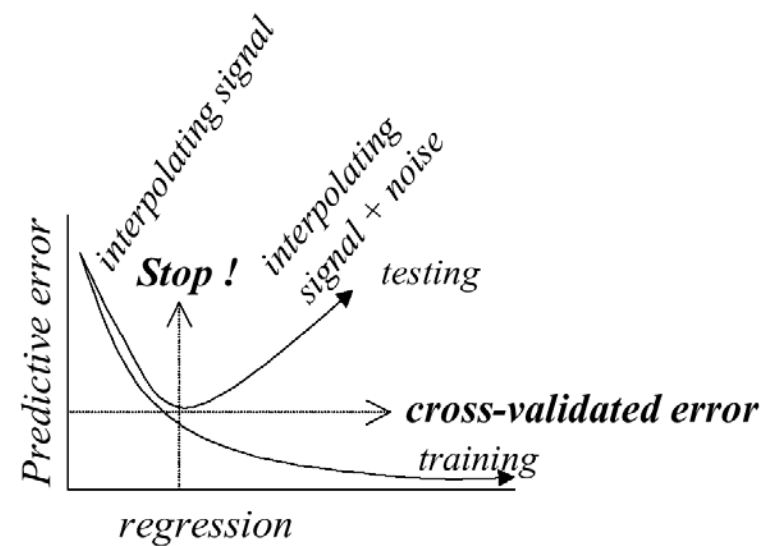
- **Ockham's razor**: prefer the simplest hypothesis consistent with data.

Generalization

Learning is concerned with accurate prediction of future data, *not* accurate prediction of training data.

Cross-validation

How do we ensure good generalization, i.e. avoid “over-fitting” on our particular data sample.



- You are ultimately interested in good performance on new (unseen) test data.
- To estimate that, split off a (smallish) subset of the training data (called validation set).
- Train without validation data and “test” on validation data.
- Repeat this over multiple splits of the data and average results.
- Reasonable split: 90% train, 10% test, average over the 10 splits.