Statistical Geometry Processing

Winter Semester 2011/2012

Machine Learning

UNIVERSITÄT DES SAARLANDES

CLUSTER OF EXCELLENCE

max planck institut informatik

Topics

Topics

- Machine Learning Intro
  - Learning is density estimation
  - The curse of dimensionality
- Bayesian inference and estimation
  - Bayes rule in action
  - Discriminative and generative learning
- Markov random fields (MRFs) and graphical models
- Learning Theory
  - Bias and Variance / No free lunch
  - Significance

Machine Learning

& Bayesian Statistics

Statistics

How does machine learning work?

- **Learning:** learn a probability distribution
- **Classification:** assign probabilities to data

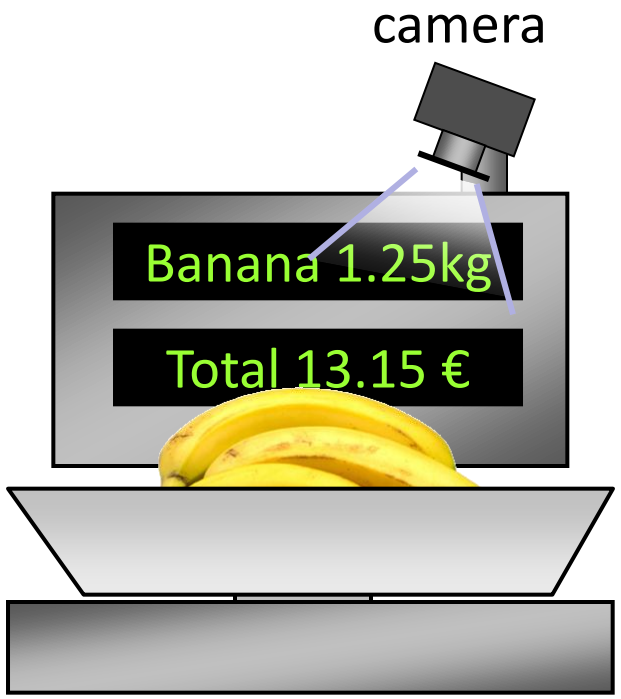
We will look only at classification problems:

- Distinguish two classes of objects
- From ambiguous data

Application

Application Scenario:

- Automatic scales at supermarket
- Detect type of fruit using a camera



Learning Probabilities

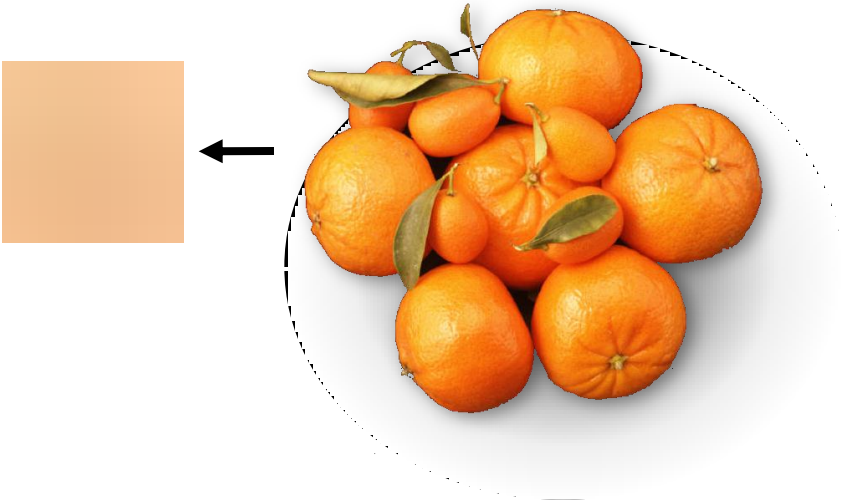
Toy Example:

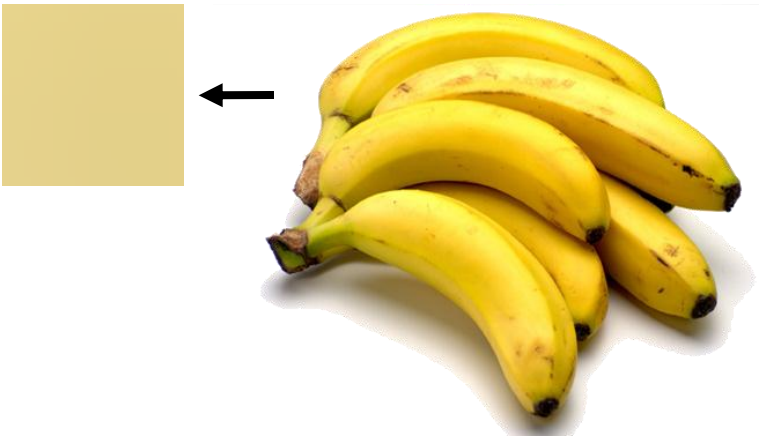
- We want to distinguish pictures of oranges and bananas
- We have 100 training pictures for each fruit category
- From this, we want to derive a rule to distinguish the pictures automatically

Learning Probabilities

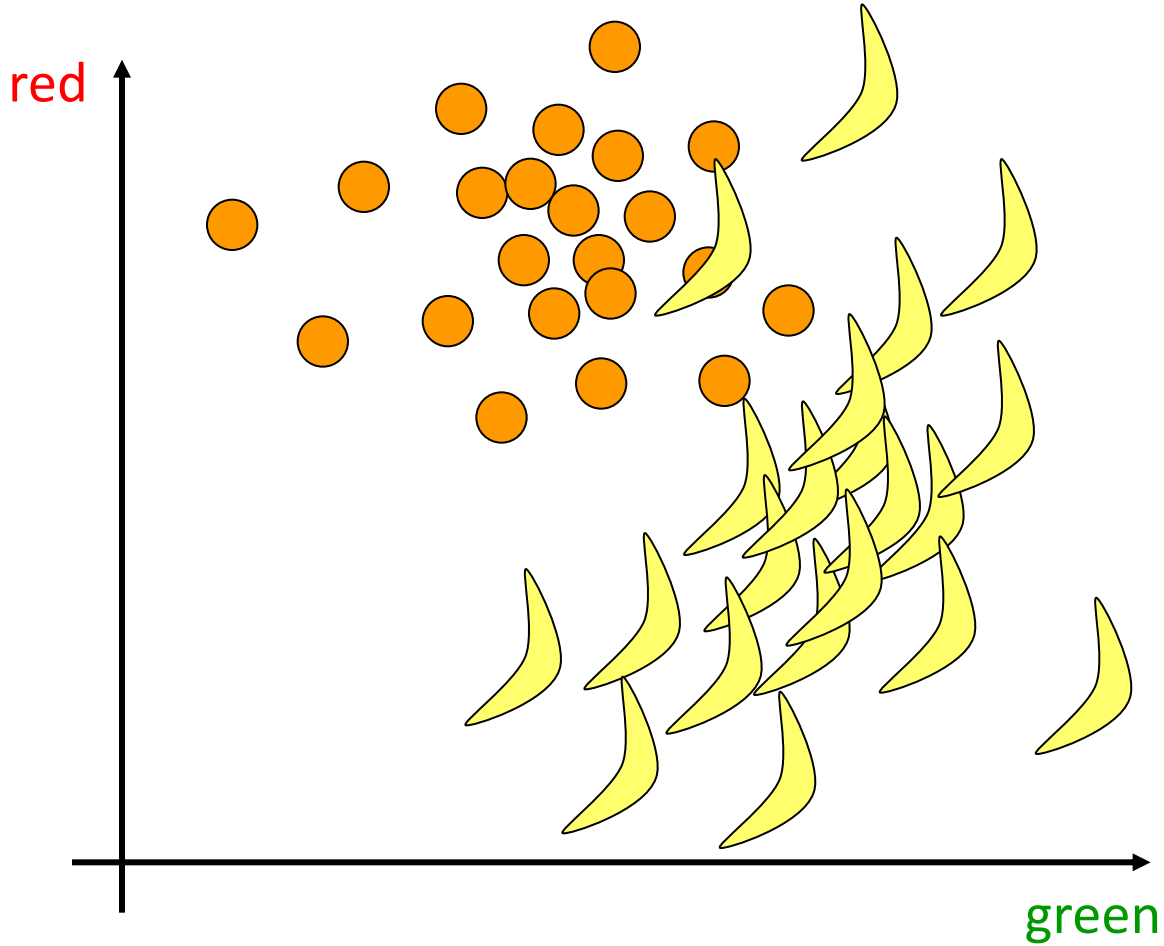
Very simple algorithm:

- Compute average color
- Learn distribution





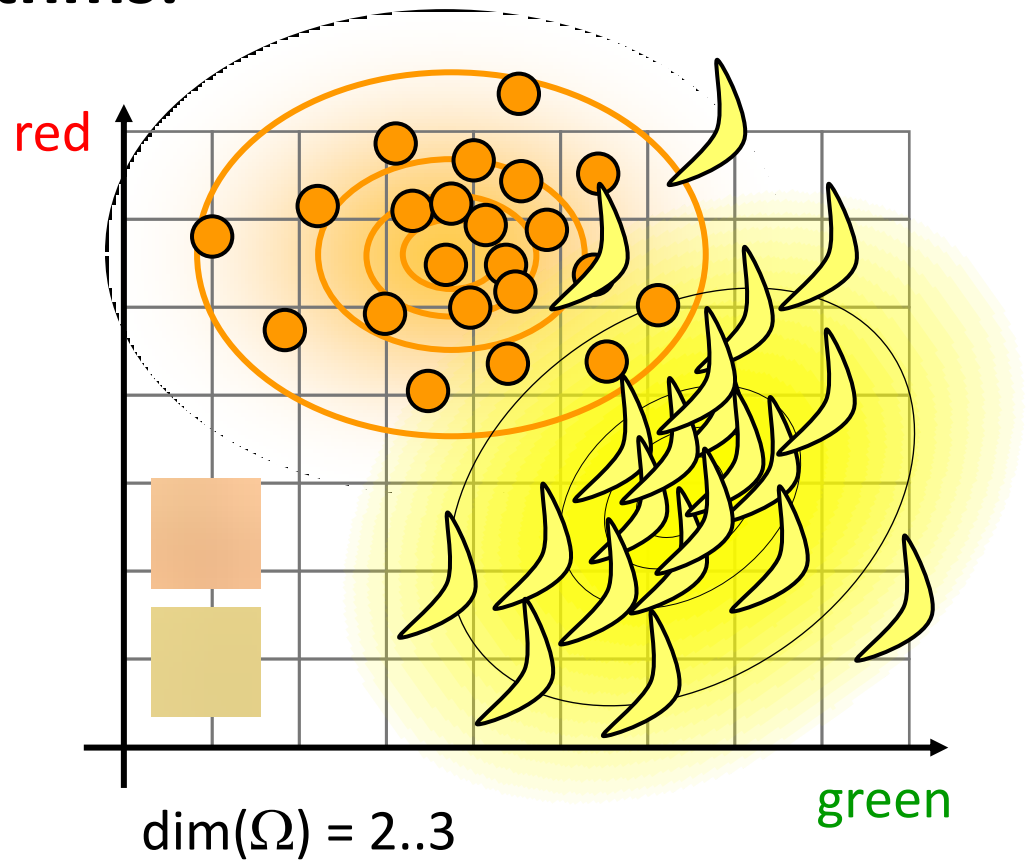
Learning Probabilities



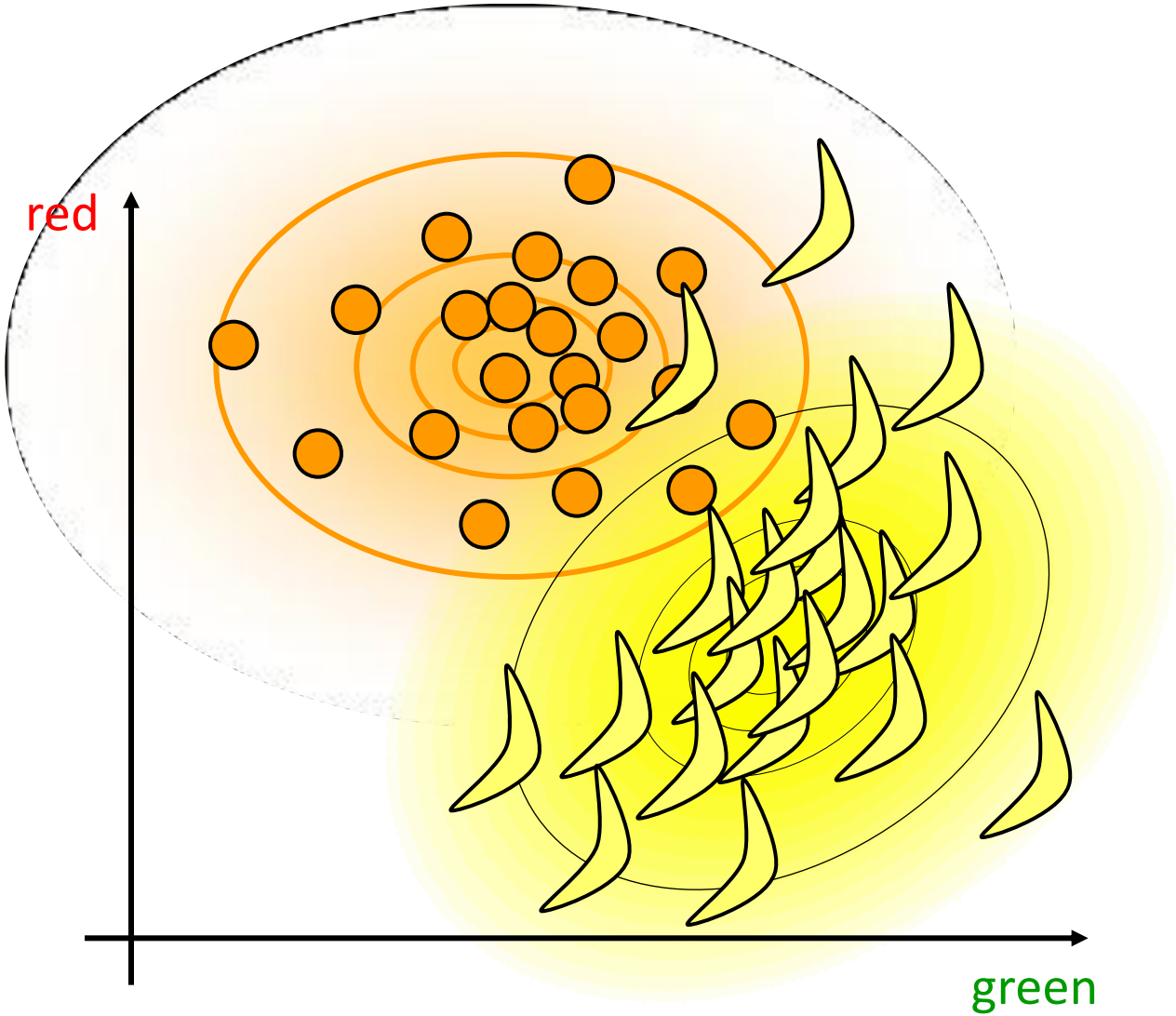
Simple Learning

Simple Learning Algorithms:

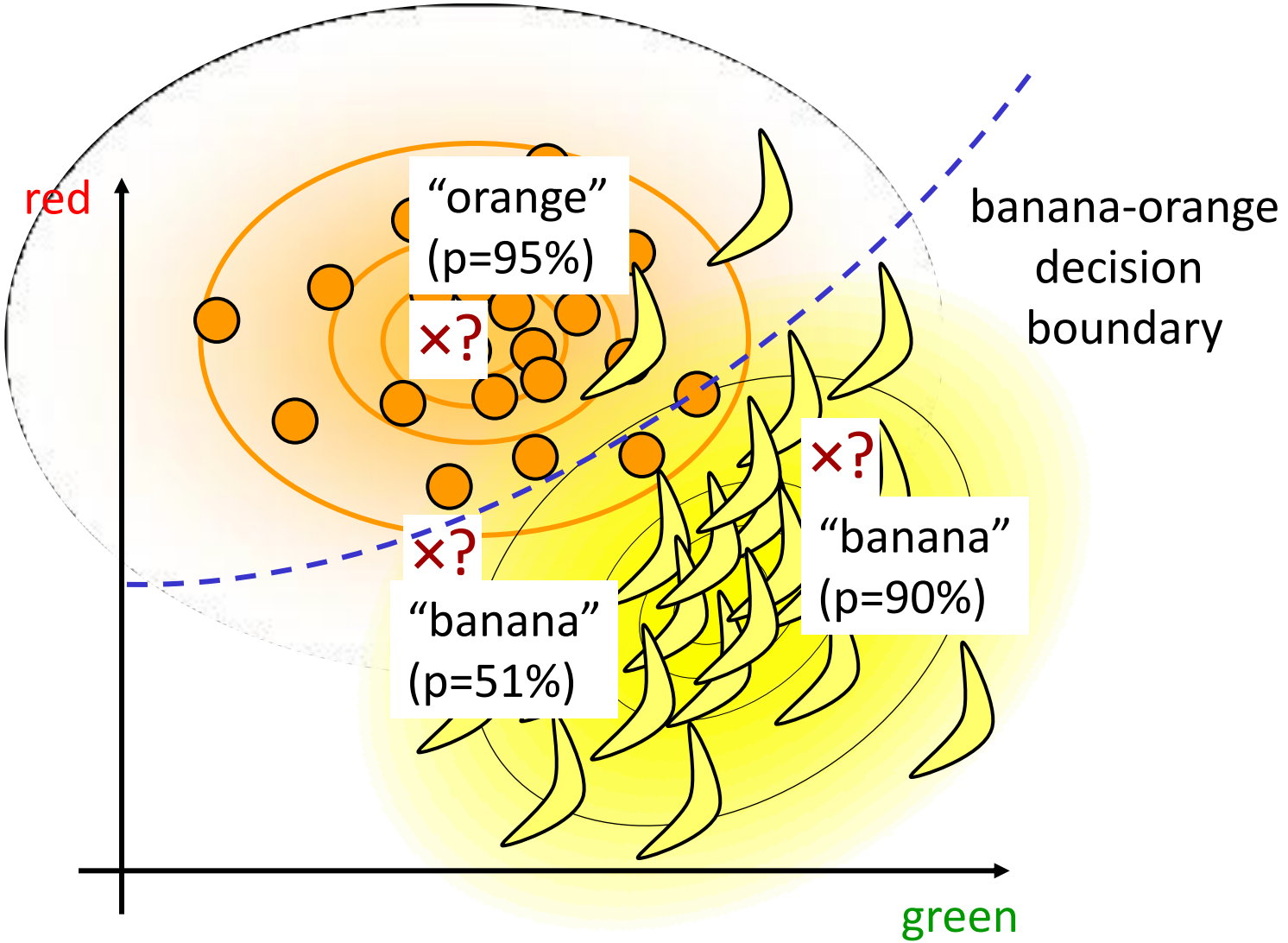
- Histograms
- Fitting Gaussians
- We will see more



Learning Probabilities



Learning Probabilities



Machine Learning

Very simple idea:

- Collect data
- Estimate probability distribution
- Use learned probabilities for classification (etc.)
- We always decide for the most likely case (largest probability)

Easy to see:

- If the probability distributions are known exactly, this decision is optimal (in expectation)
- “Minimal Bayesian risk classifier”

What is the problem?

Why is machine learning difficult?

- We need to learn the probabilities
- Typical problem: High dimensional input data

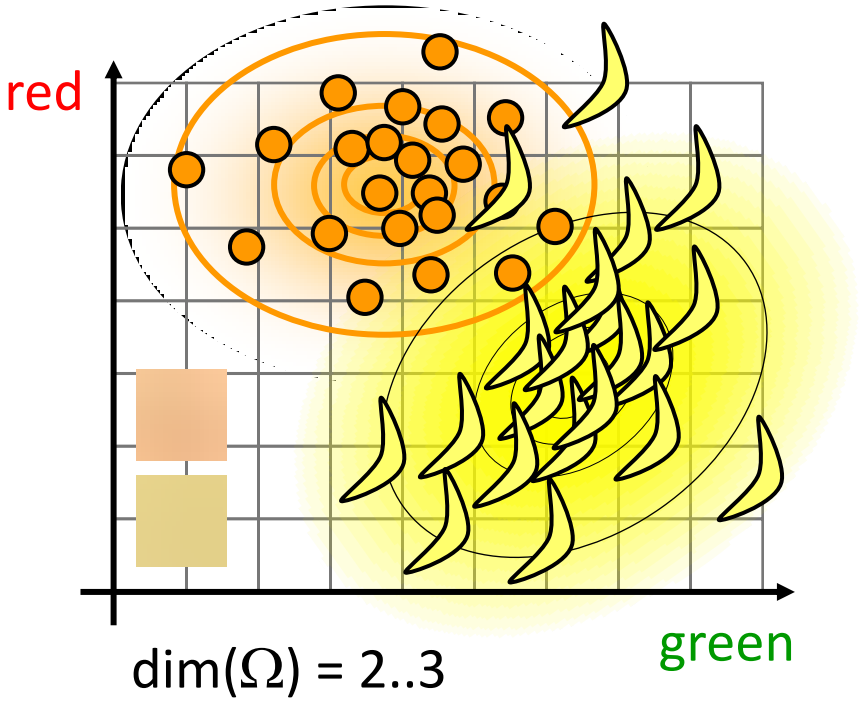
High Dimensional Spaces

color:
3D (RGB)

image: 100 x 100 pixel
30000 dimensions

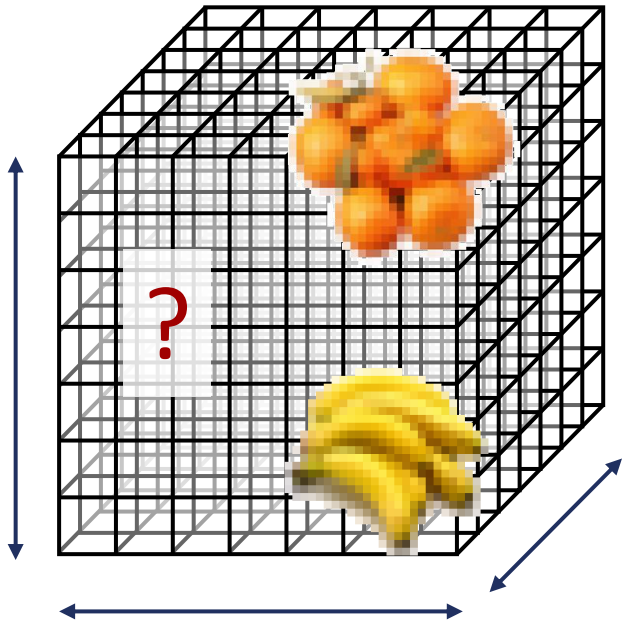
High Dimensional Spaces

average color learning



$\dim(\Omega) = 2..3$

full image learning



30000 dimensions

High Dimensional Spaces

High dimensional probability spaces:

- Too much space to fill
- We can never get a sufficient number of examples
- Learning is almost impossible

What can we do?

- We need additional assumptions
- Simplify probability space
- Model statistical dependencies

This makes machine learning a hard problem.

Learn From High Dimensional Input

Learning Strategies:

- **Features** to reduce the dimension
  - Average color
  - Boundary shape
  - Other heuristics

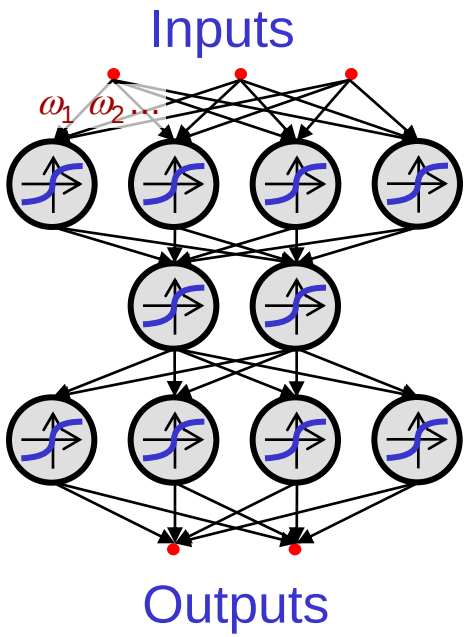
  Usually chosen manually. (black magic?)
- High-dimensional learning techniques
  - Neural networks (old school)
  - Support vector machines (current “standard” technique)
  - Ada-boost, decision trees, ... (many other techniques)
- Usually used in combination

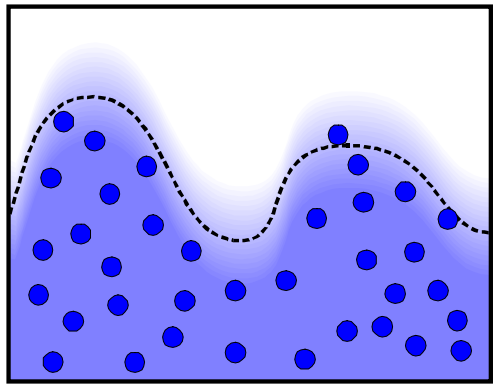
Basic Idea: Neural Networks

Classic Solution:

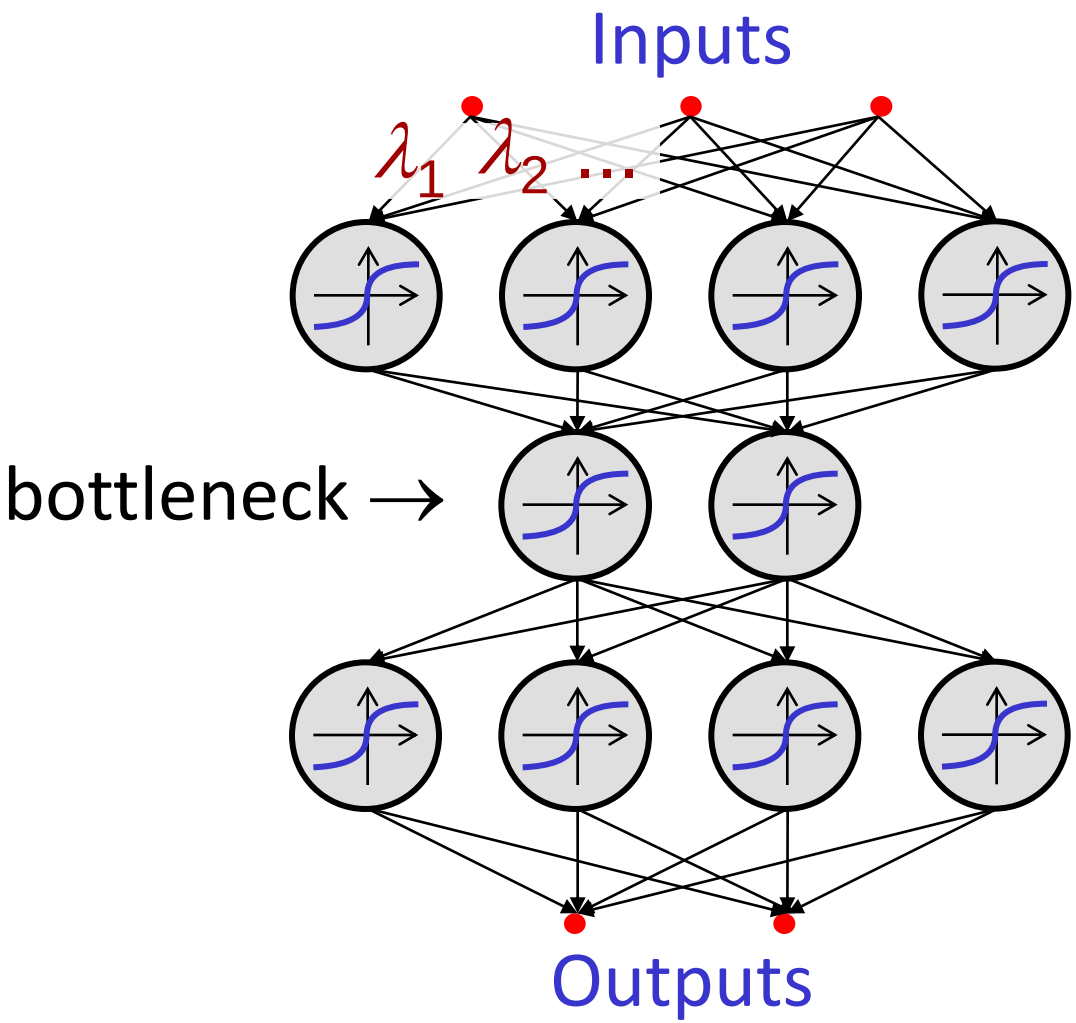
Neural Networks

- Non-linear functions
  - Features as input
  - Combine basic functions with weights
- Optimize to yield
  - (1,0) on bananas
  - (0,1) on oranges
- Fit non-linear decision boundary to data

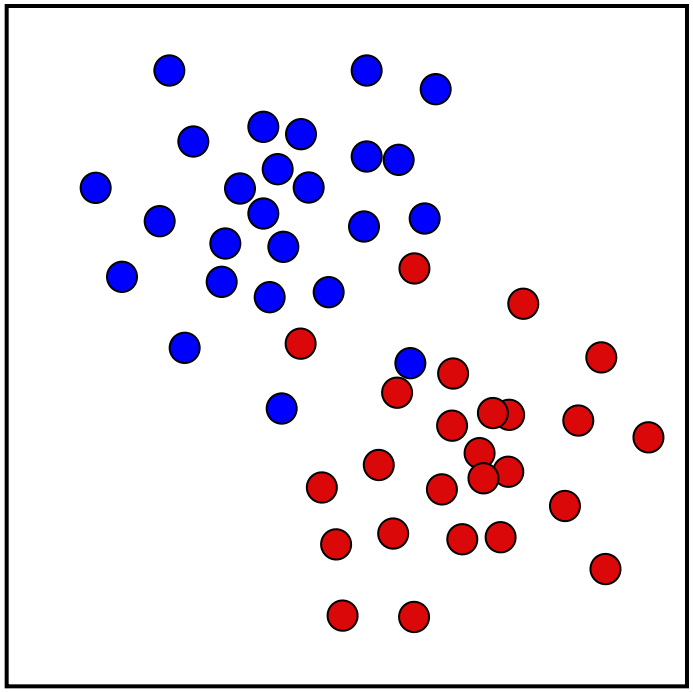




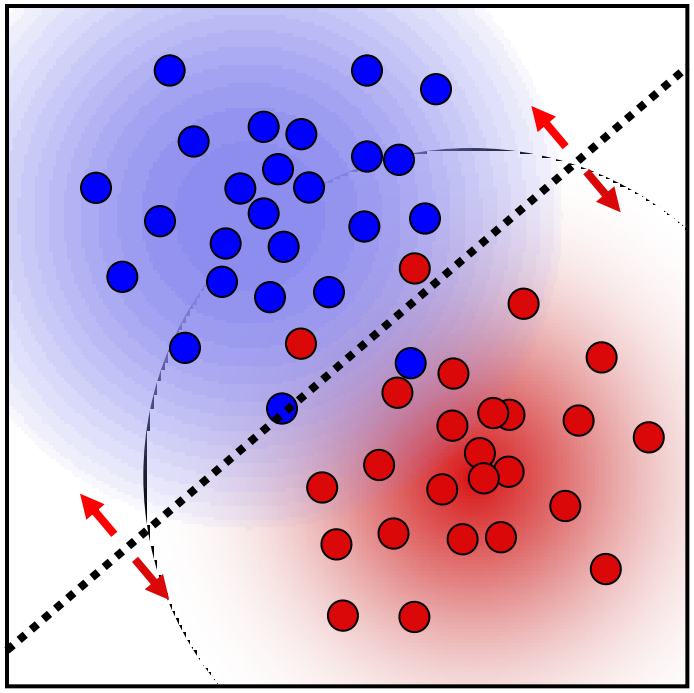
Neural Networks



Support Vector Machines

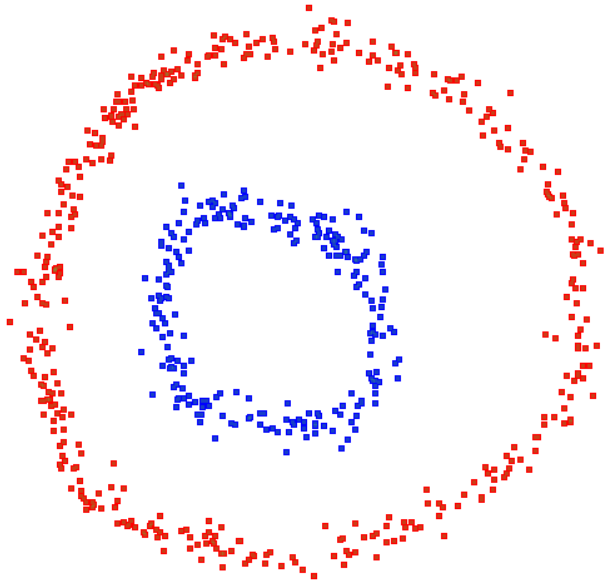


training set

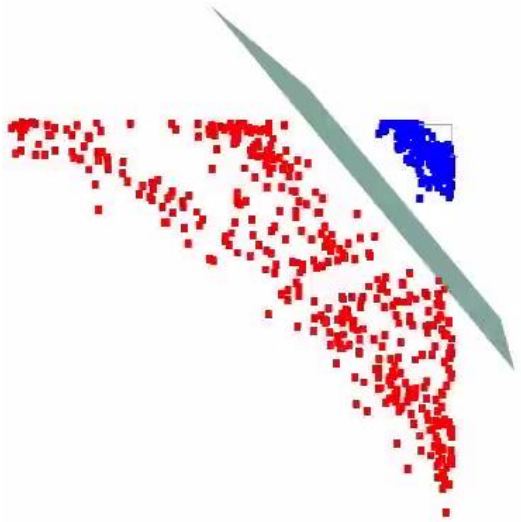


best separating hyperplane

Kernel Support Vector Machine



original space



“feature space”

Example Mapping:

$$\phi: \mathbb{R}^2 \to \mathbb{R}^3$$

$$(x, y) \mapsto (x^2, xy, y^2)$$

Other Learning Algorithms

Popular Learning Algorithms

- Fitting Gaussians
- Linear discriminant functions
- Ada-boost
- Decision trees
- …

More Complex Learning Tasks

Learning Tasks

Examples of Machine Learning Problems

- **Pattern recognition**
  - Single class (banana / non-banana)
  - Multi class (banana, orange, apple, pear)
  - *Howto:* Density estimation, highest density minimizes risk
- **Regression**
  - Fit curve to sparse data
  - *Howto:* Curve with parameters, density estimation for parameters
- **Latent variable regression**
  - Regression between observables and hidden variables
  - *Howto:* Parametrize, density estimation

Supervision

Supervised learning

- Training set is labeled

Semi-supervised

- Part of the training set is labeled

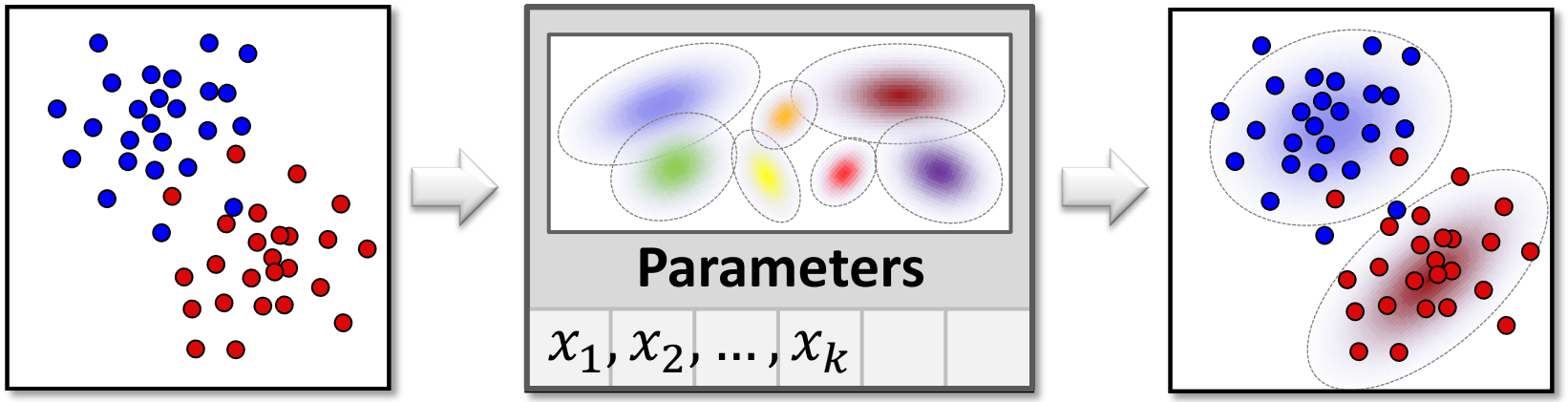
Unsupervised

- No labels, find structure on your own (“Clustering”)

Reinforcement learning

- Learn from experience (losses/gains; robotics)

Principle



training set

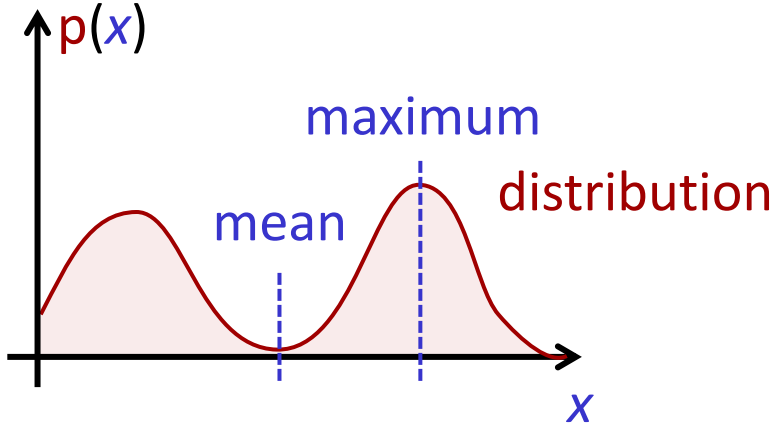
Model

hypothesis

Two Types of Learning

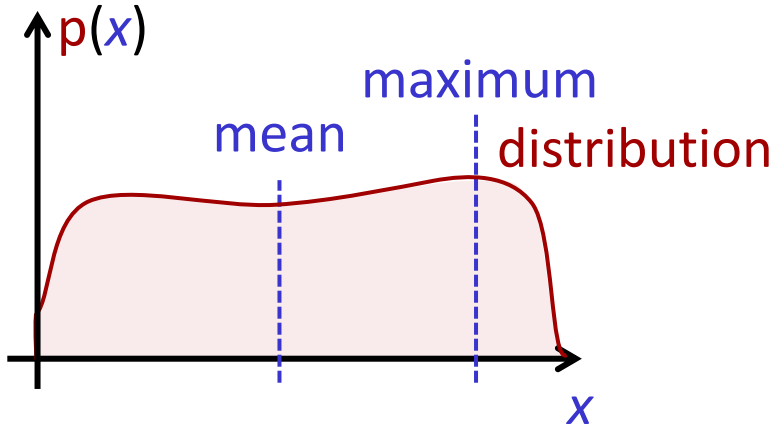
Estimation:

- Output *most likely parameters*
  - Maximum density
    - "Maximum likelihood"
    - "Maximum a posteriori"
  - Mean of the distribution



Inference:

- Output probability density
  - *Distribution for parameters*
  - More information
- Marginalize to reduce dimension



Bayesian Models

Scenario

- Customer picks banana  ($X = 0$) or orange  ($X = 1$)
- Object X creates image D

Modeling

- Given image D (observed), what was X (latent)?

$$P(X|D) = \frac{P(D|X)P(X)}{P(D)}$$

$$P(X|D) \sim P(D|X)P(X)$$

Bayesian Models

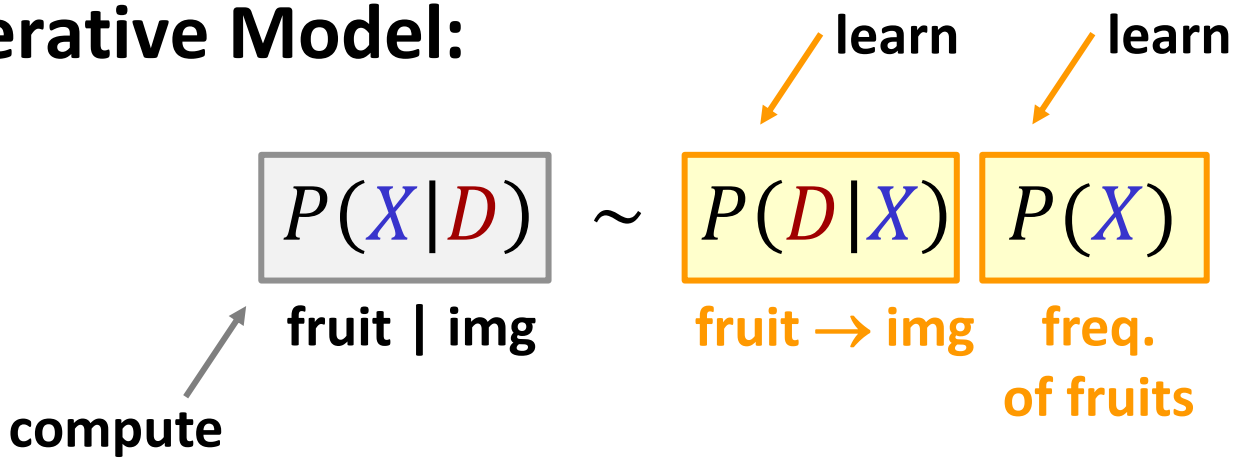
Model for Estimating X

$$P(X|D) \sim P(D|X)\,P(X)$$

posterior — **data term, likelihood** — **prior**

Generative vs. Discriminative

Generative Model:

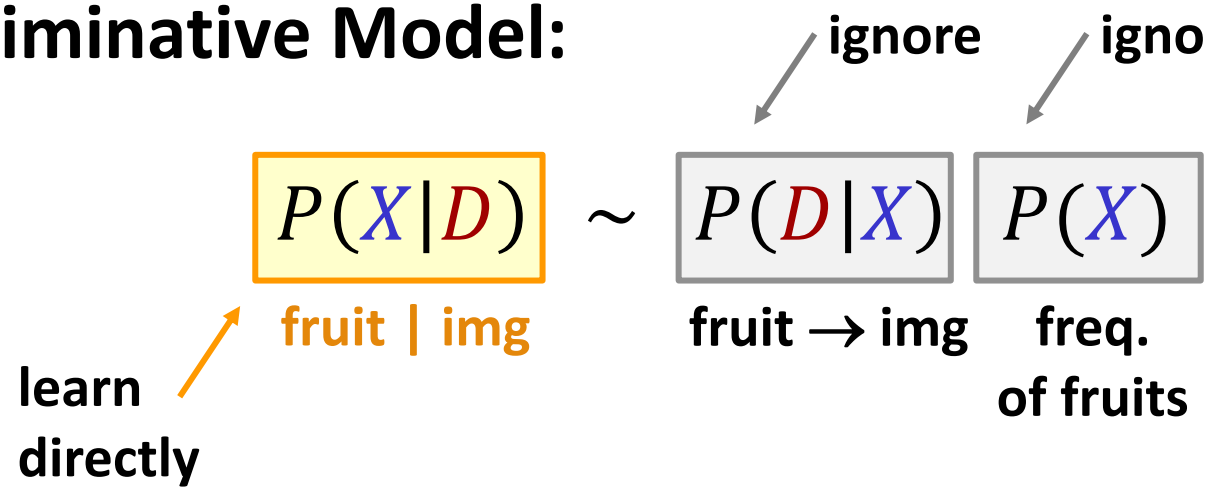


Properties

- Comprehensive model:
  Full description of how data is created
- Might be complex (how to create images of fruit?)

Generative vs. Discriminative

Discriminative Model:



Properties

- Easier:
  - Learn mapping from phenomenon to explanation
  - Not trying to explain / understand the whole phenomenon
- Often easier, but less powerful

Statistical Dependencies

Markov Random Fields and Graphical Models

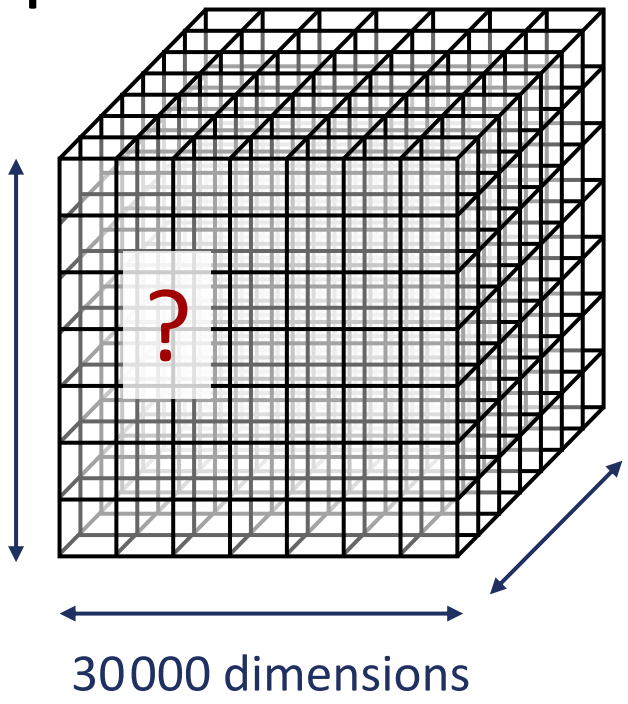
Problem

Estimation Problem:

$$P(X|D) \sim P(D|X)\,P(X)$$

posterior — **data term, likelihood** — **prior**

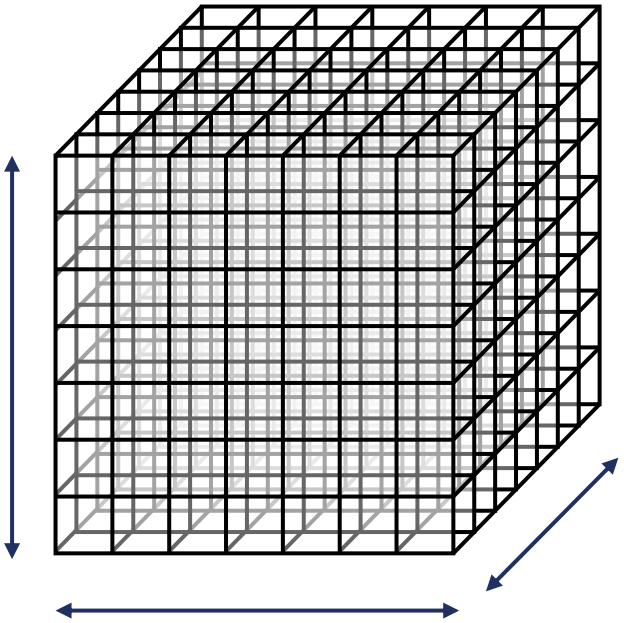
- X = 3D mesh (10K vertices)
- D = noisy scan (or the like)
- Assume P($D|X$) is known
- But: Model P(X) cannot be build
  - Not even enough training data
  - In this part of the universe :-)



Reducing dependencies

Problem:

- $p(x_1, x_2, \ldots, x_{10000})$ is to high-dimensional
- *k* States, *n* variables: O(k^n) density entries
- General dependencies kill the model



Idea

- Hand-craft decencies
- We might know or guess what actually depends on each other and what not
- This is the art of machine learning

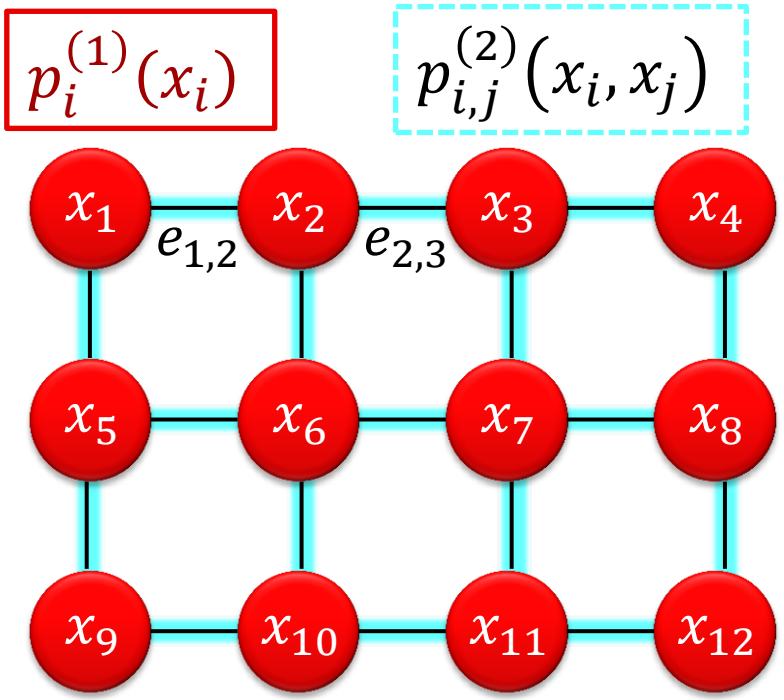
Graphical Models

Factorize Models

- Pairwise models:

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{i=1}^{n} p_i^{(1)}(x_i) \prod_{i,j \in E} p_{i,j}^{(2)}(x_i, x_j)$$

- Model complexity:
  - O(nk^2) parameters
- Higher order models:
  - Triplets, quadruples as factors
  - Local neighborhoods



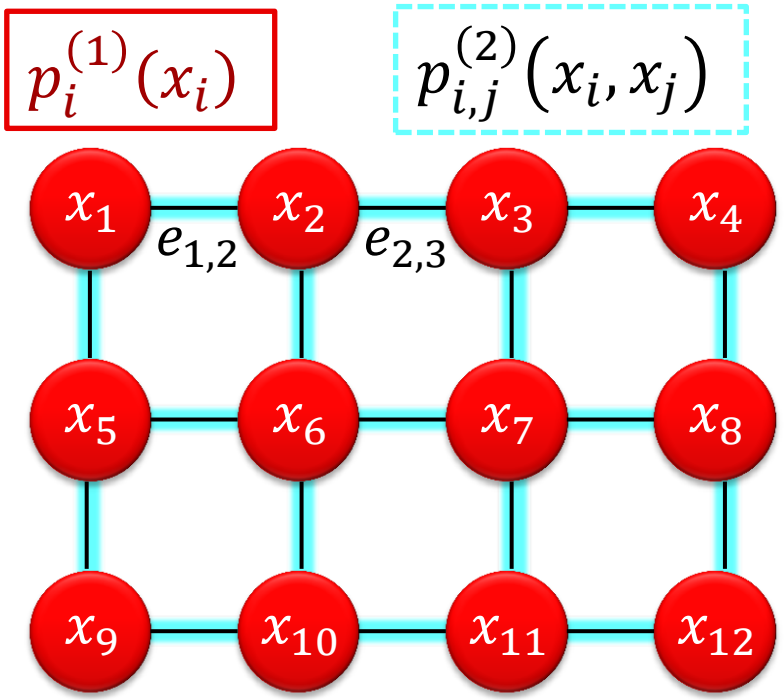
Graphical Models

Markov Random fields

- Factorize density in local "cliques"

Graphical model

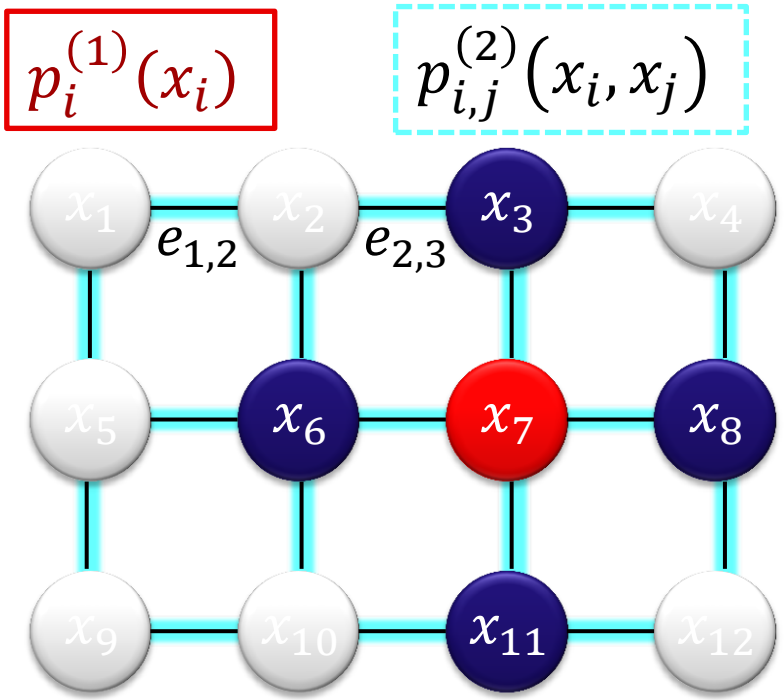
- Connect variables that are directly dependent
- Formal model: Conditional independence



Graphical Models

Conditional Independence

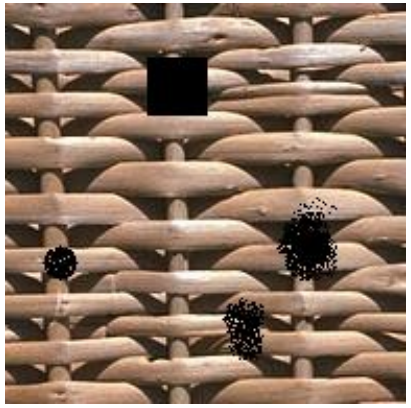
- A node is conditionally independent of all others given the values of its direct neighbors
- I.e. set these values to constants, x_7 is independent of all others



Theorem (Hammersley–Clifford):

- Given conditional independence as graph, a (positive) probability density factors over cliques in the graph

Example: Texture Synthesis



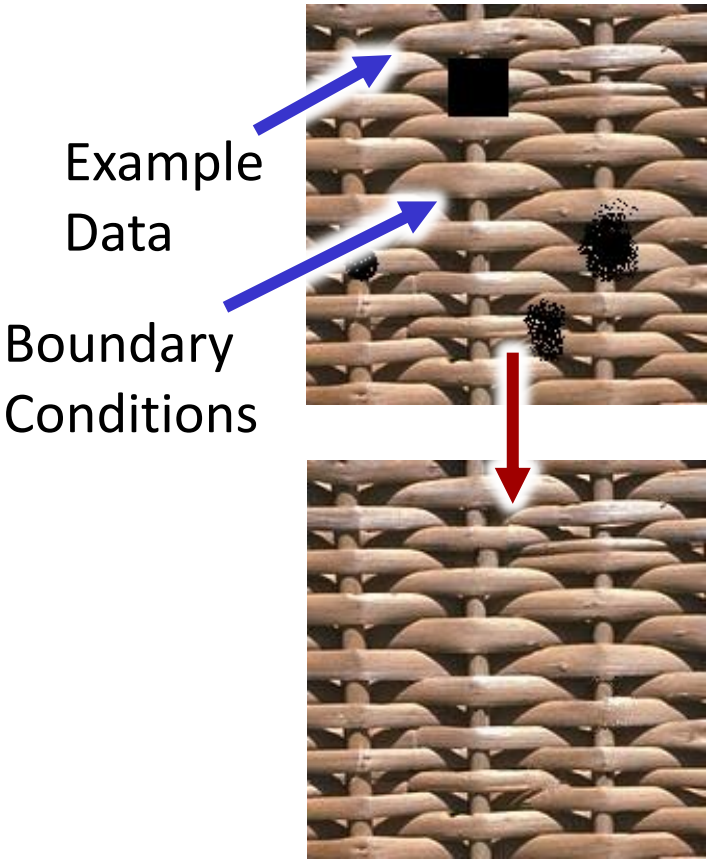
region selected

completion

Texture Synthesis

Idea

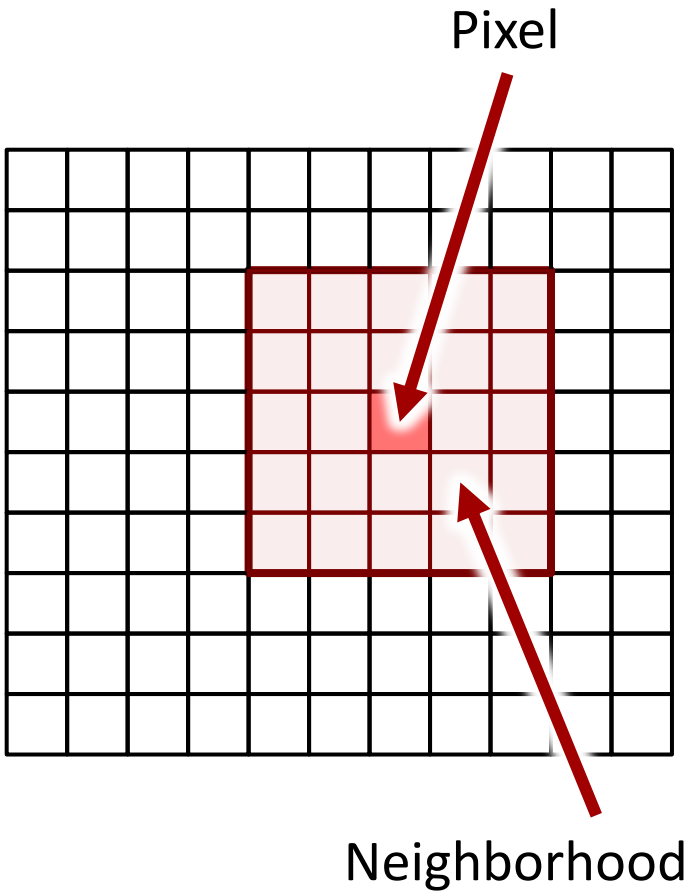
- One or more images as examples
- Learn image statistics
- Use knowledge:
  - Specify boundary conditions
  - Fill in texture



The Basic Idea

Markov Random Field Model

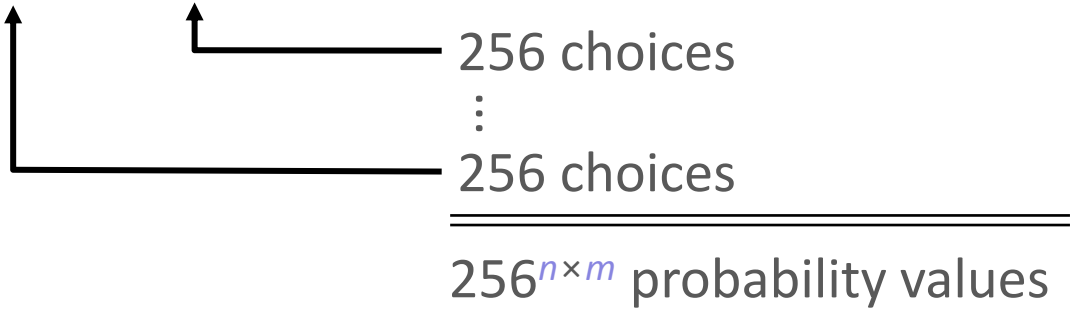
- Image statistics
- How pixels are colored depends on local neighborhood only (Markov Random Field)
- Predict color from neighborhood



A Little Bit of Theory...

Image statistics:

- An image of $n \times m$ pixels
- Random variable: $\mathbf{x} = [x_{11},\ldots,x_{nm}] \in [0, 1, \ldots, 255]^{n \times m}$
- Probability distribution:
  $p(\mathbf{x}) = p(x_{11}, \ldots, x_{nm})$



It is impossible to learn full images from examples!

Simplification

Problem:

- Statistical dependencies
- Simple modell can express dependencies on all kinds of combinations

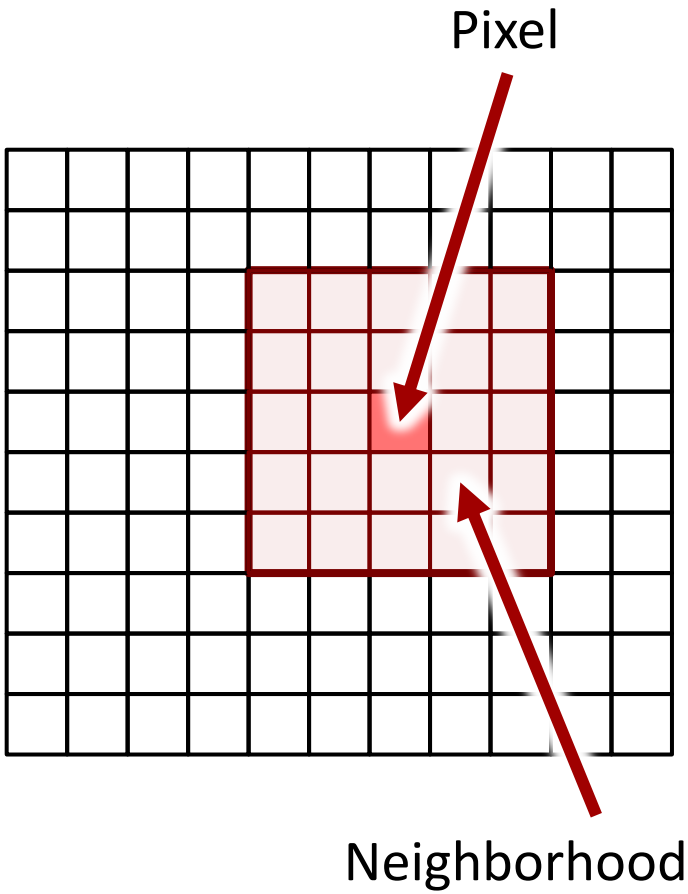
Markov Random Field:

- Each pixel is *conditionally independent* of the rest of the image given a small neighborhood
- In English: likelihood only depends on neighborhood, not rest of the image

Markov Random Field

Example:

- Red pixel depends on light red region
- Not on black region
- If region is known, probability is fixed and independent of the rest



However:

- Regions overlap
- Indirect global dependency

Texture Synthesis

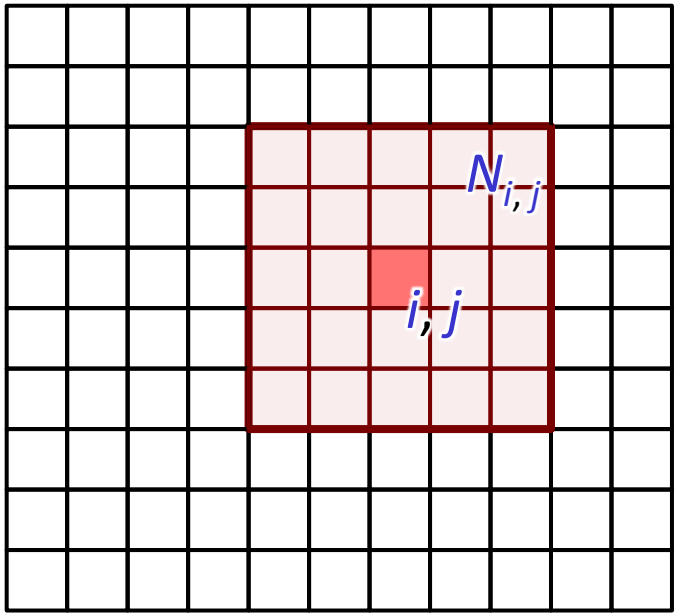
Use for Texture Synthesis

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{n}\prod_{j=1}^{m} p_{i,j}(N_{i,j})$$

$$p_{i,j} = p_{i,j}(N_{i,j})$$

$$= p_{i,j}(x_{i-k,j-k} \cdots, x_{i+k,j+k})$$

$$\sim \exp\left(\frac{-dist(N_{i,j}, data)^2}{2\sigma^2}\right)$$



Inference

Inference Problem

- Computing $p(\mathbf{x})$ is trivial for known $\mathbf{x}$.
- Finding the $\mathbf{x}$ that maximizes $p(\mathbf{x})$ is very complicated.
- In general: NP-hard
- No efficient solution known (not even for the image case)

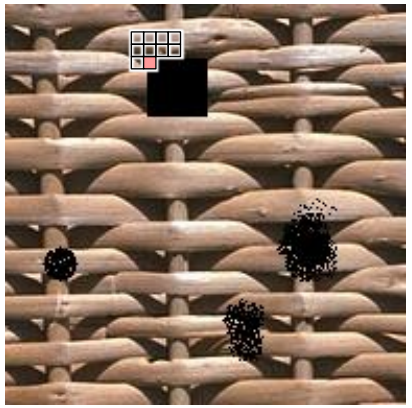
In practice

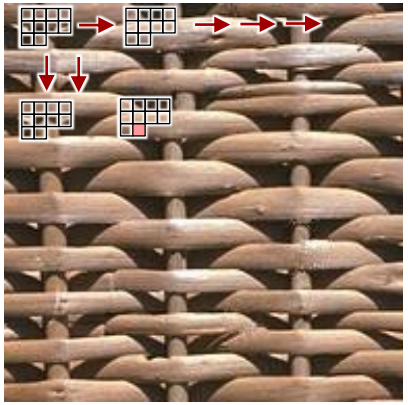
- Different approximation strategies
  ("heuristics", strict approximation is also NP-hard)

Simple Practical Algorithm

Here is the short story:

- Unknown pixels:
  consider known neighborhood
- Match to all of the known data
- Copy the pixel with the best matching neighborhood
- Region growing, outside in





Approximation only

- Can run into bad local minima

Learning Theory

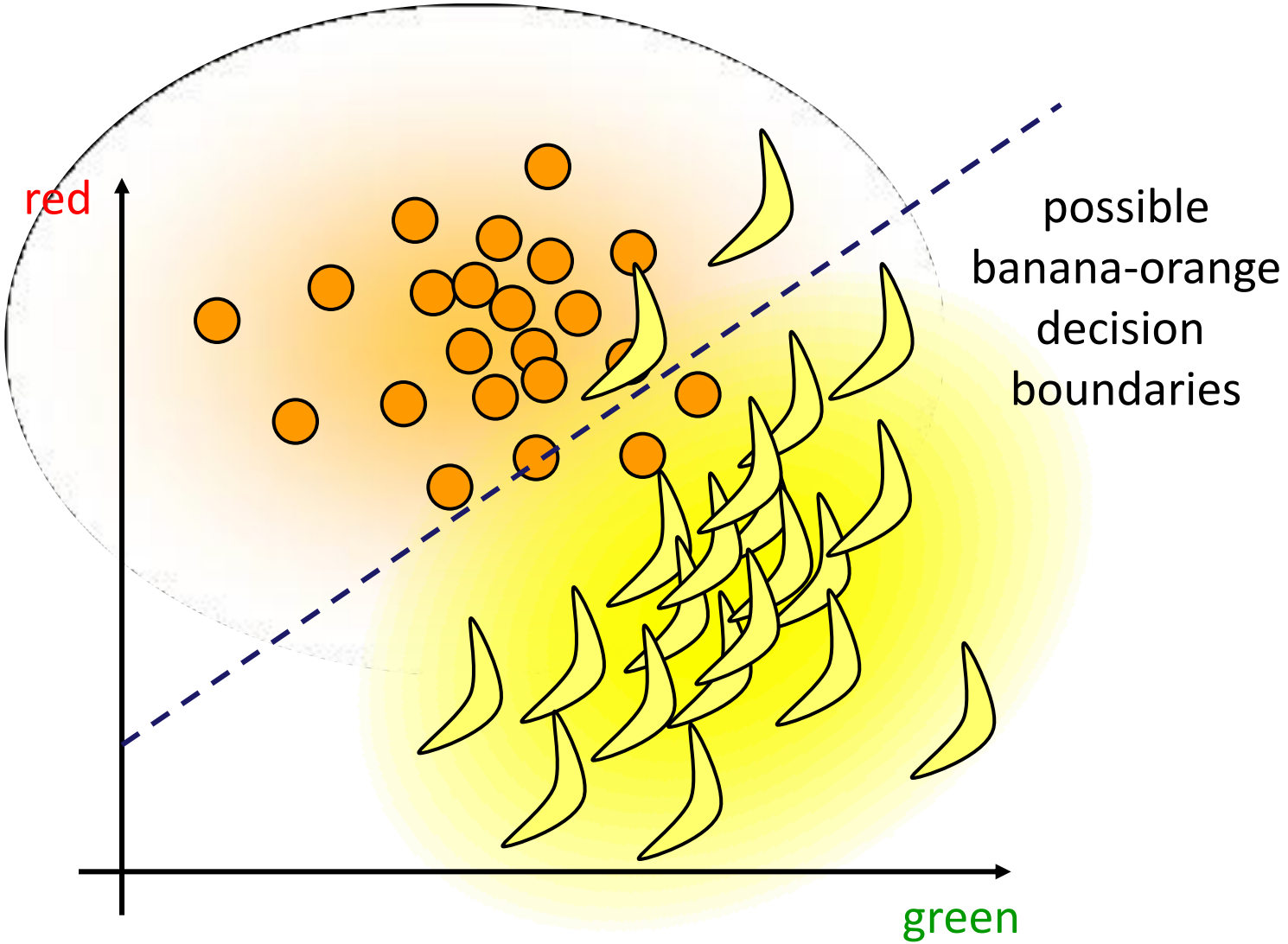
There is no such thing as a free lunch...

Overfitting

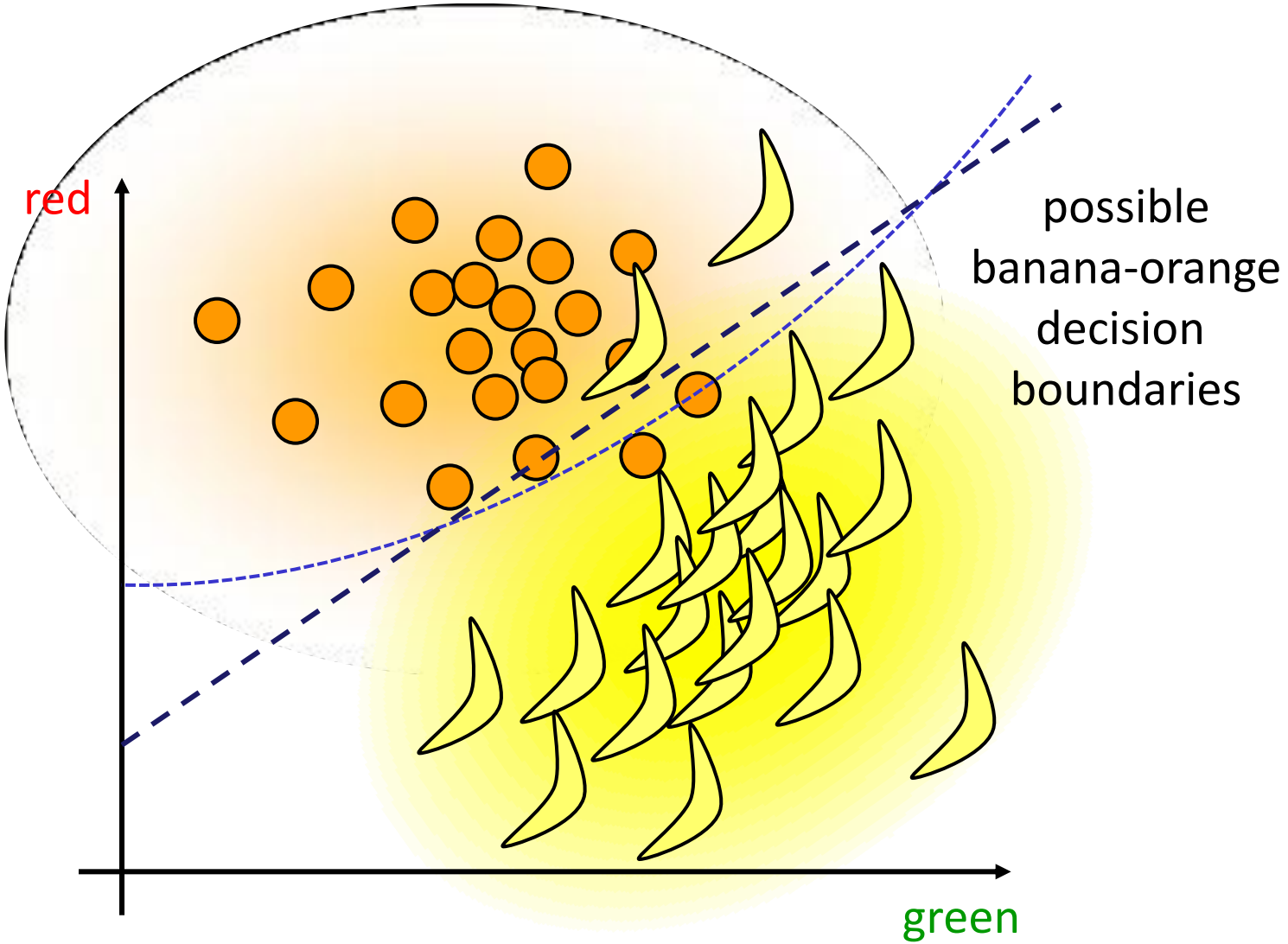
Problem: Overfitting

- Two steps:
  - Learn model on *training data*
  - Use model on more data ("*test data*")
- Overfitting
  - High accuracy in training is no guarantee for later performance

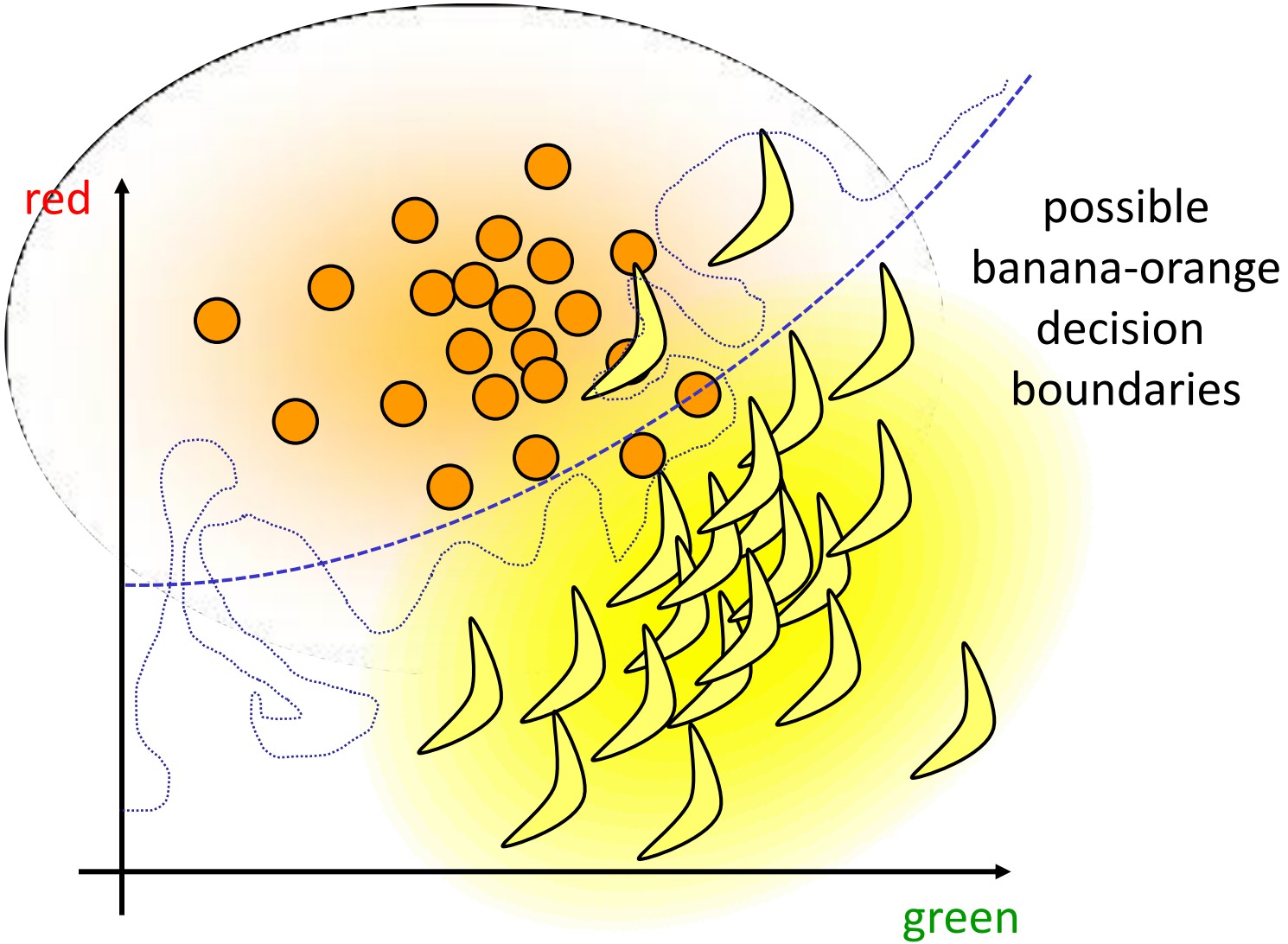
Learning Probabilities



Learning Probabilities

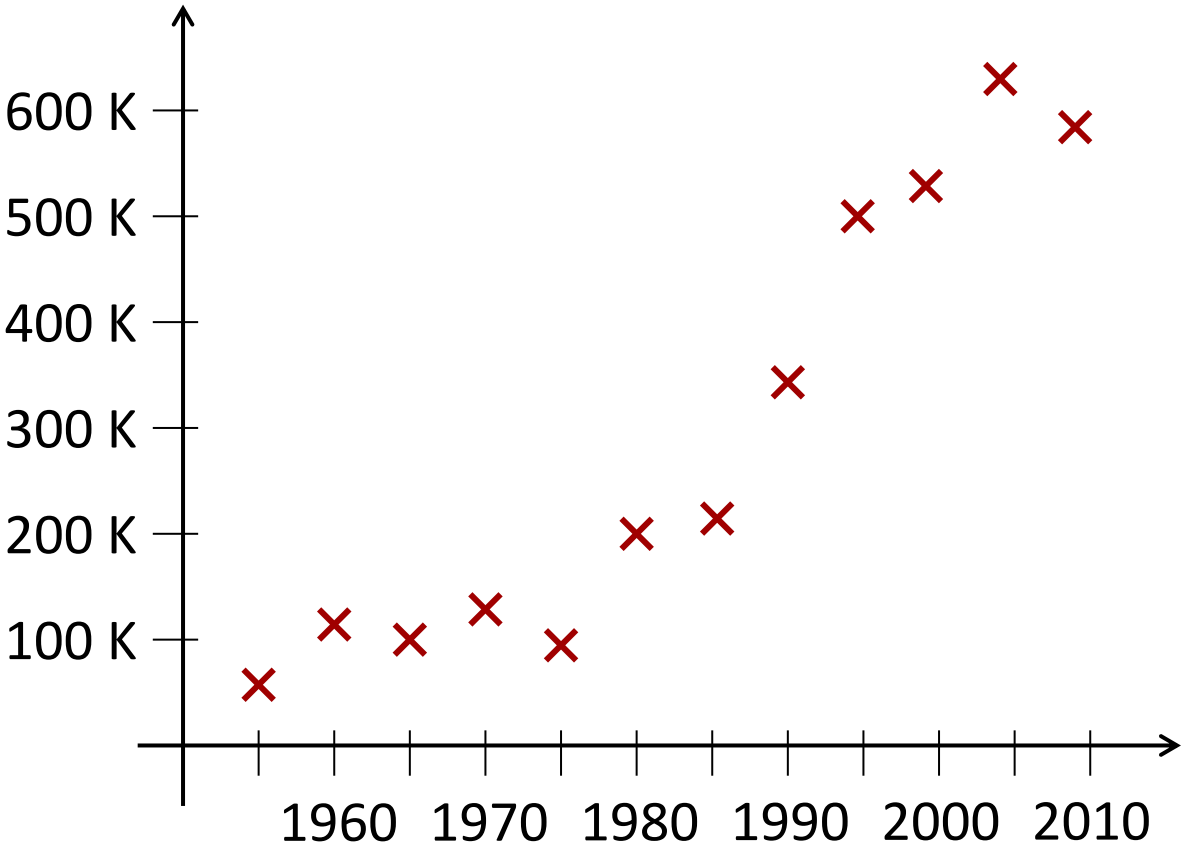


Learning Probabilities



Regression Example

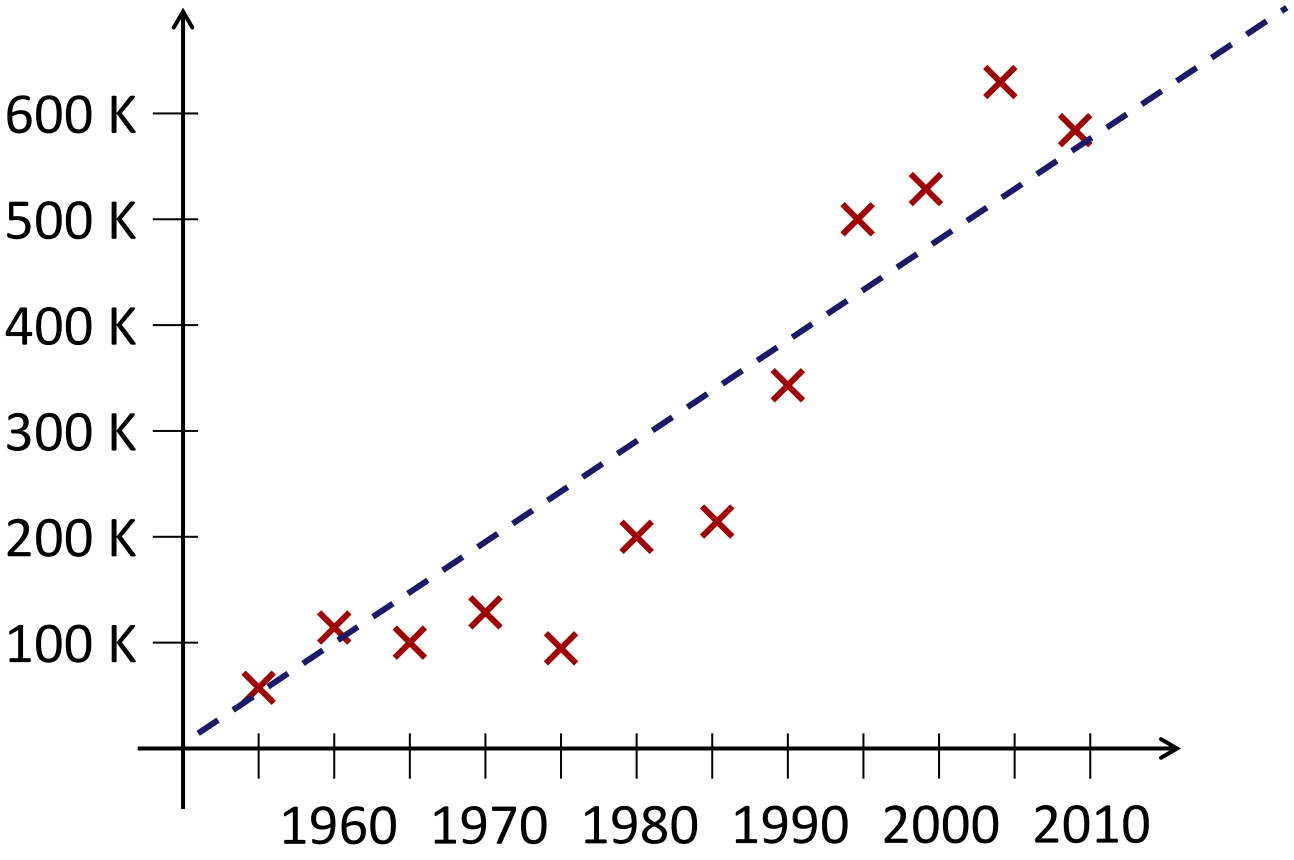
Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

Regression Example

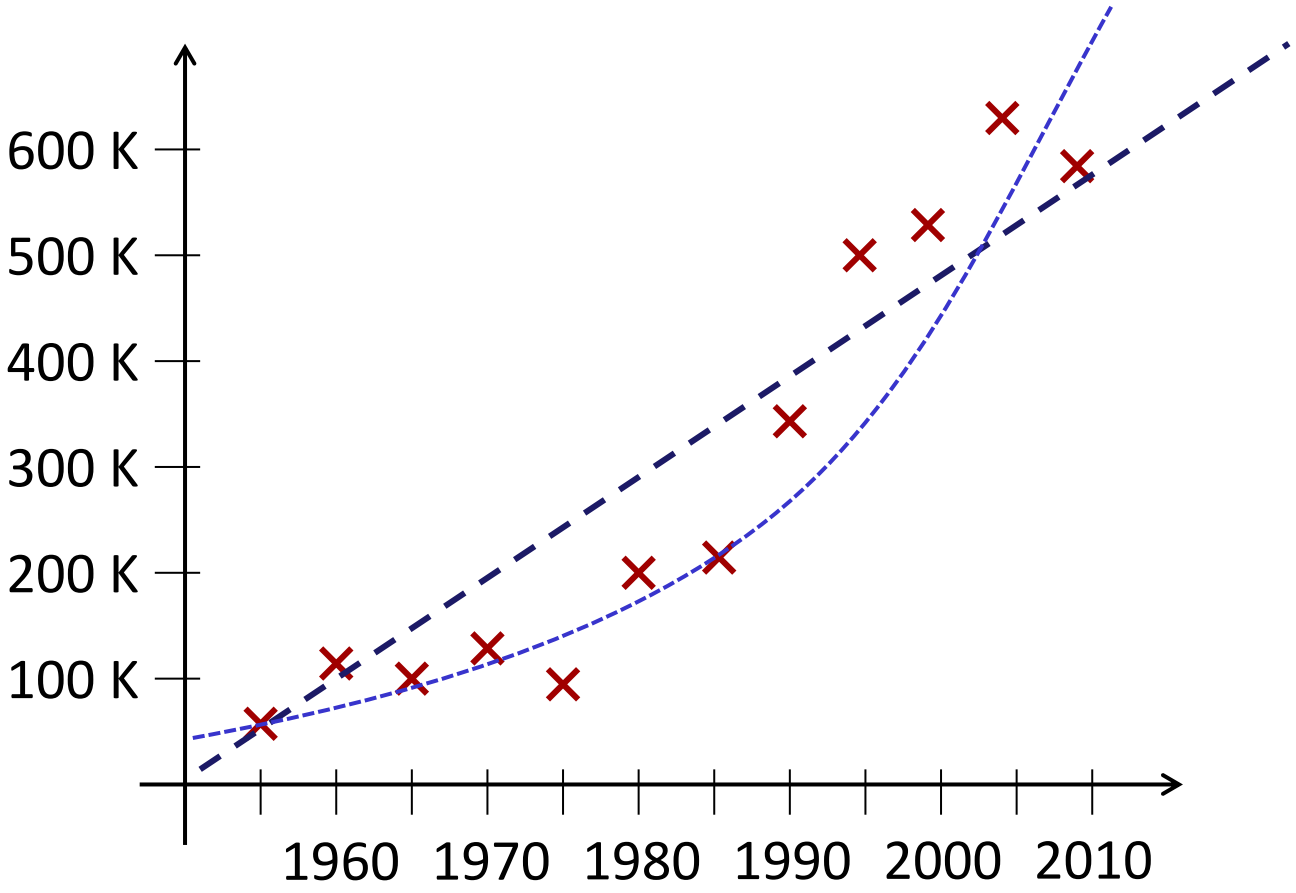
Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

Regression Example

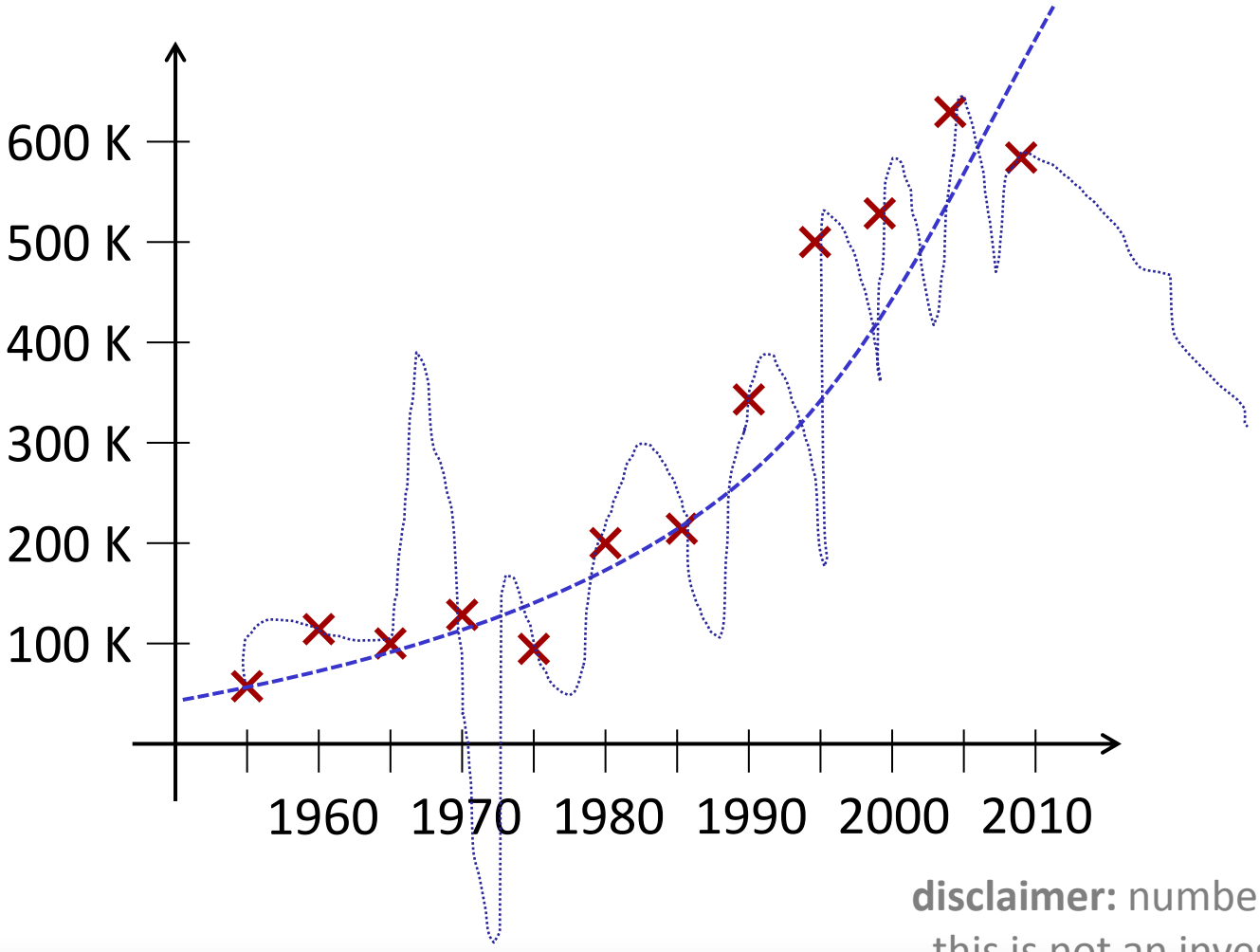
Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

Regression Example

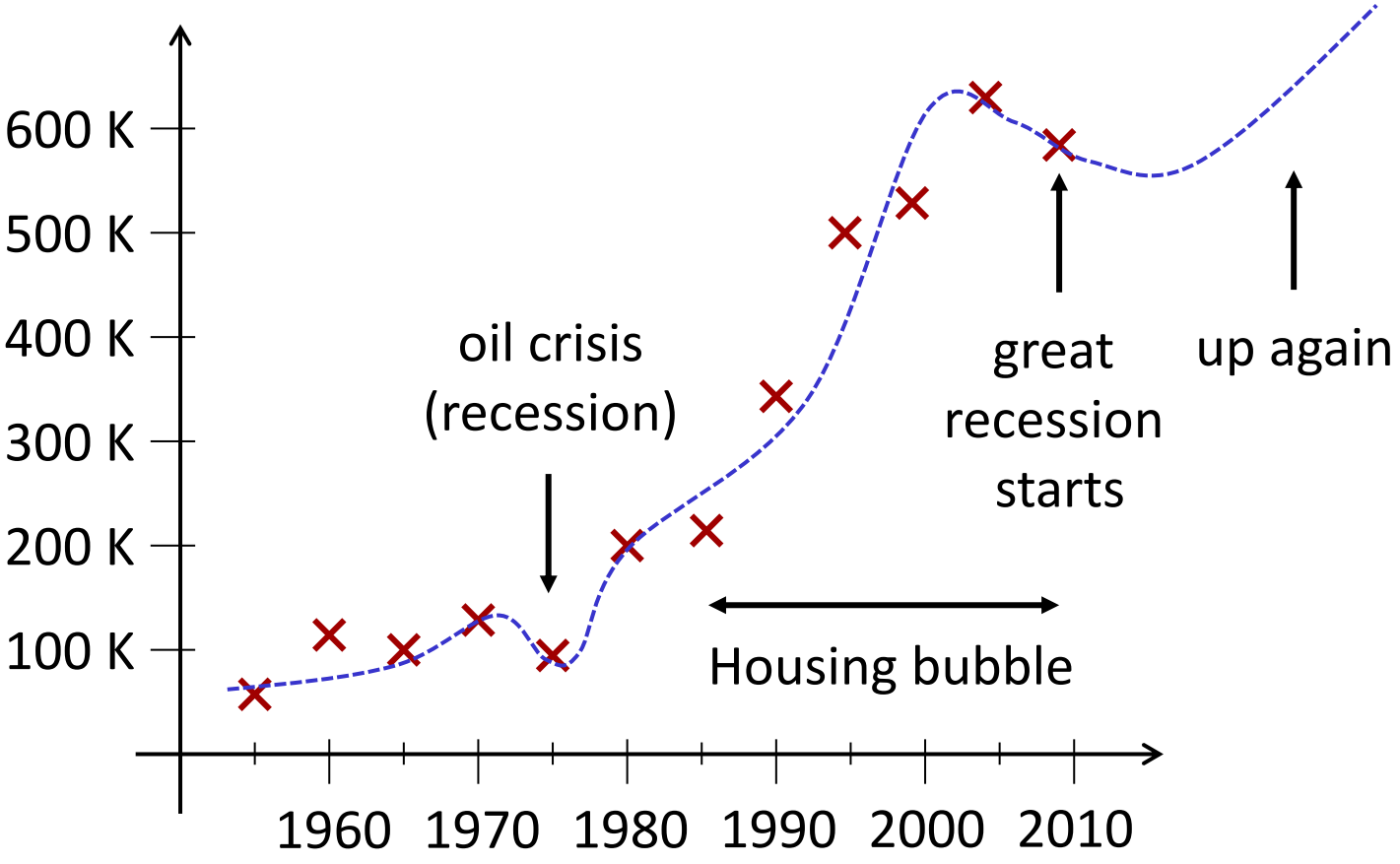
Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

Regression Example

Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

Bias – Variance Tradeoff

There is a trade off:

Bias:

- Coarse prior assumptions to regularize model

Variance:

- Bad generalization performance

Model Selection

How to choose the right model?

For example

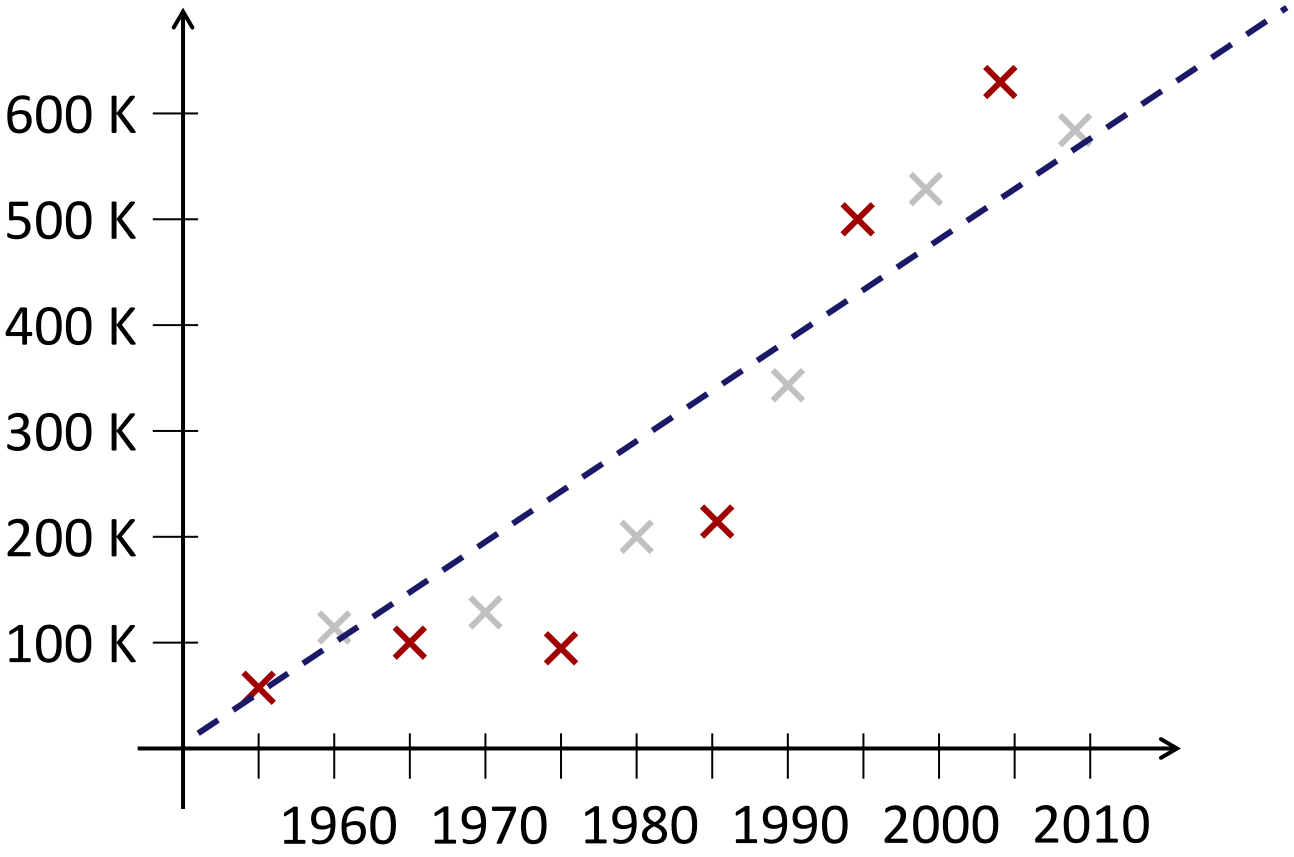
- Linear
- Quadratic
- Higher order

Standard heuristic: Cross validation

- Partition data in two parts (halfs, leave-one-out,...)
- Train on part 1, test on part 2
- Choose according to performance on part 2

Cross Validation

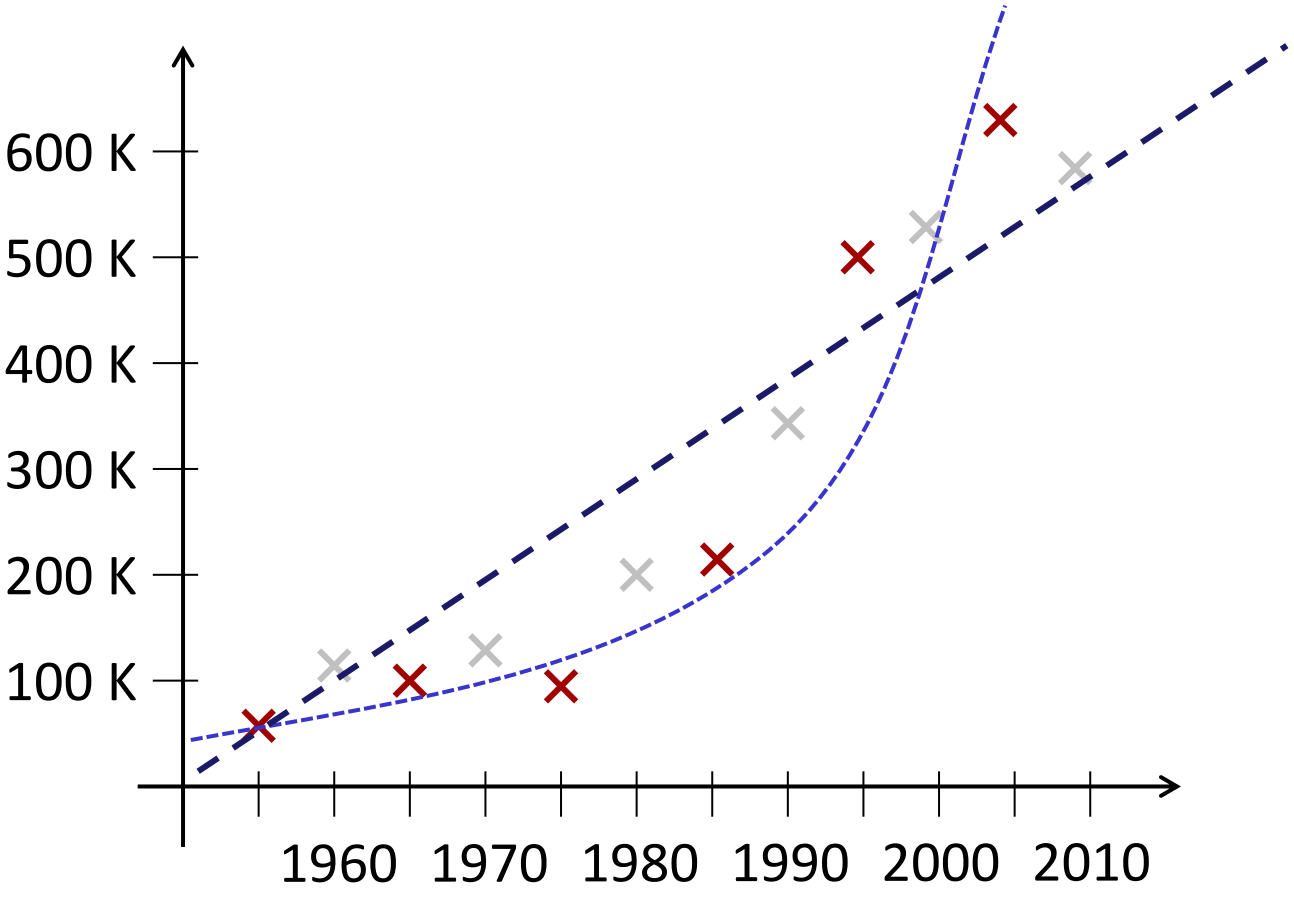
Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

Cross Validation

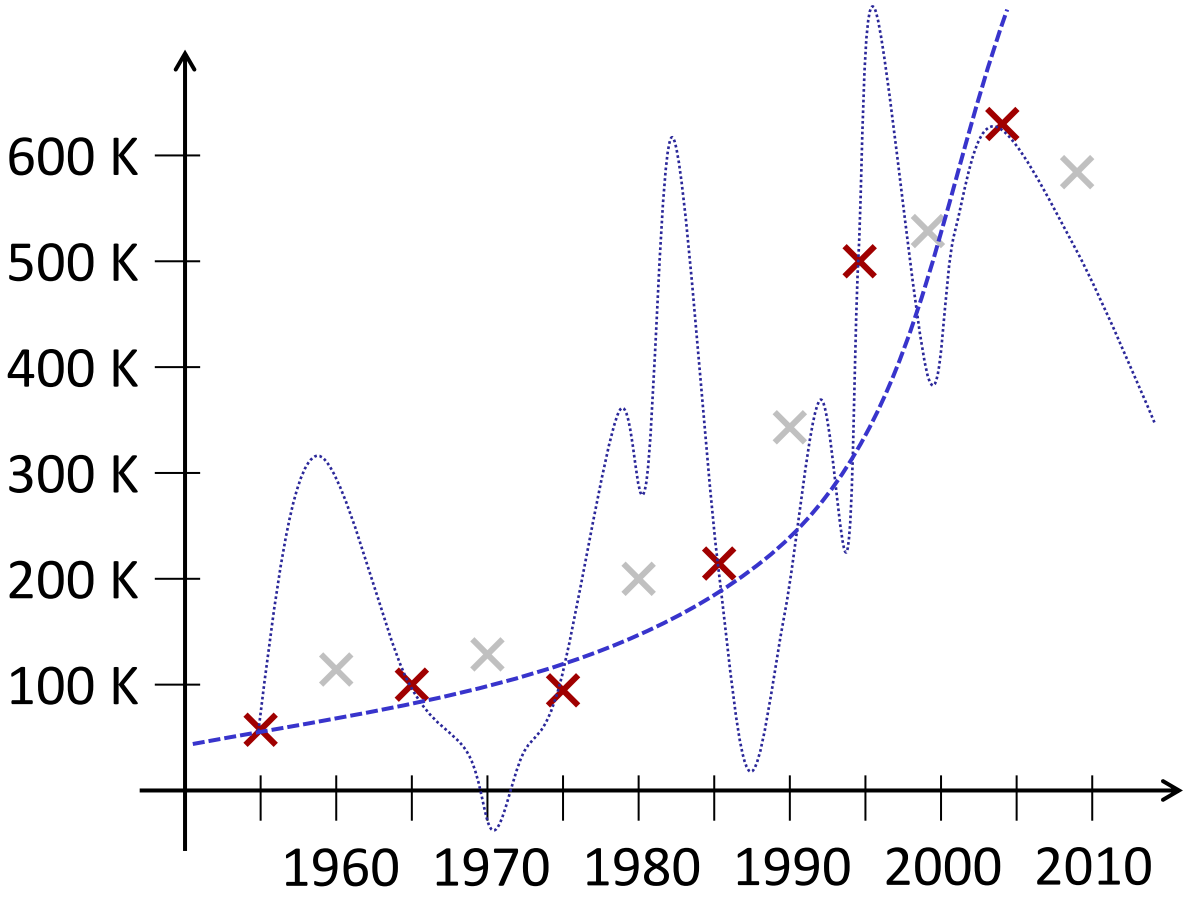
Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

Cross Validation

Housing Prices in Springfield



disclaimer: numbers are made up
this is not an investment advice

No Free Lunch Theorem

Given

- Labeling problem (holds in general as well)
  - Data $\mathbf{x}_i \in \Omega$ (for example: images of fruit)
  - Labels $l_i \in \{1,\ldots,k\}$ (for example: fruit type)
- Training data $\mathrm{D} = \{(\mathbf{x}_1 \equiv l_1), \ldots, (\mathbf{x}_n \equiv l_n)\}$

Looking for

- Hypothesis h that works everywhere on Ω
  - 1 MPixel photos: $256^{1\,000\,000}$ data items
  - Cannot cover everything with examples
- Off training error: Predictions on $\Omega \setminus \mathrm{D}$

No Free Lunch Theorem

Unknown:

- True labeling function $L: \Omega \rightarrow \{1, \ldots, k\}$

Assumption

- No prior information
- All true labeling functions are equally likely

Theorem ("no free lunch")

- Under these assumptions, all learning algorithms have the same expected performance (i.e.: averaged over all potential true L)

Consequences

Without prior knowledge:

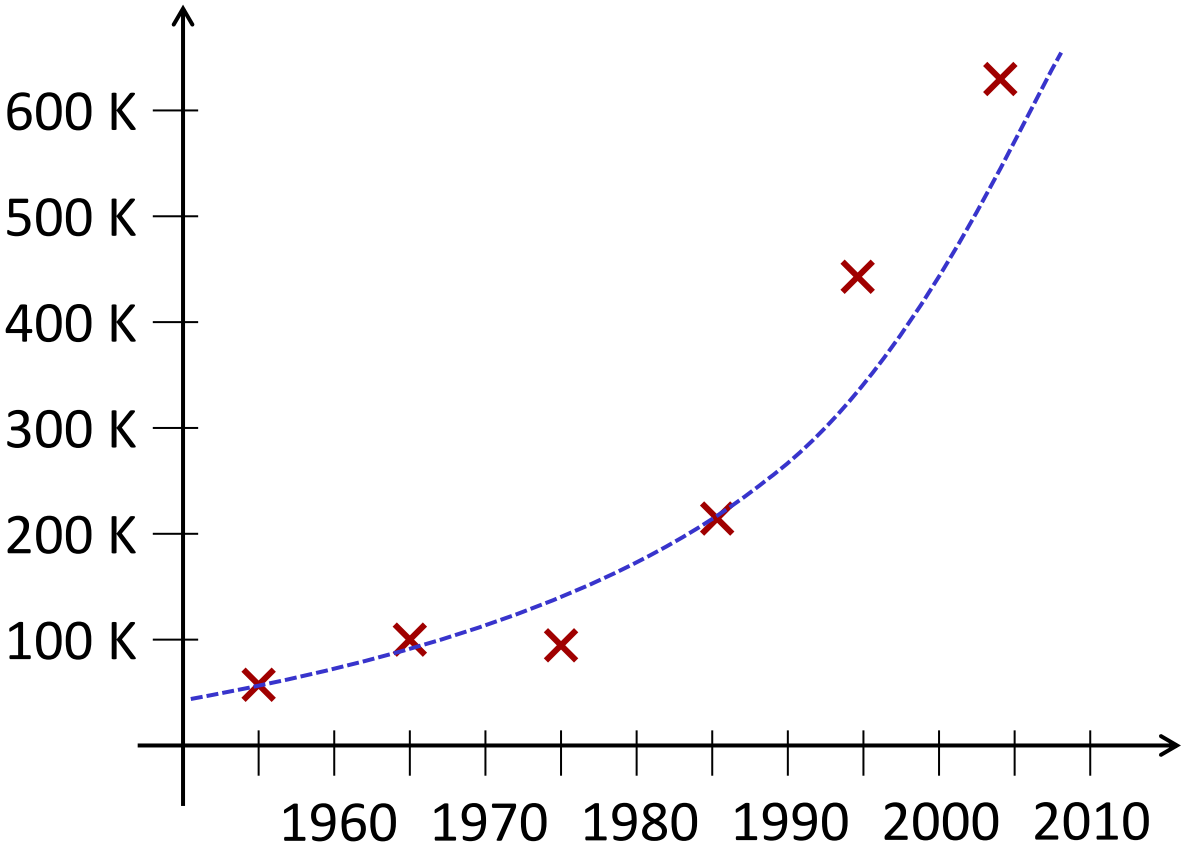
- The expected off-training error of the following algorithms is the same
  - Fancy Multi-Class Support Vector machine
  - Output random numbers
  - Output always 0
  - Learning with cross validation

There is no "ultimate learning algorithm"

- Learning from data needs further knowledge (structure assumptions)
- No truly "fully automatic" machine learning

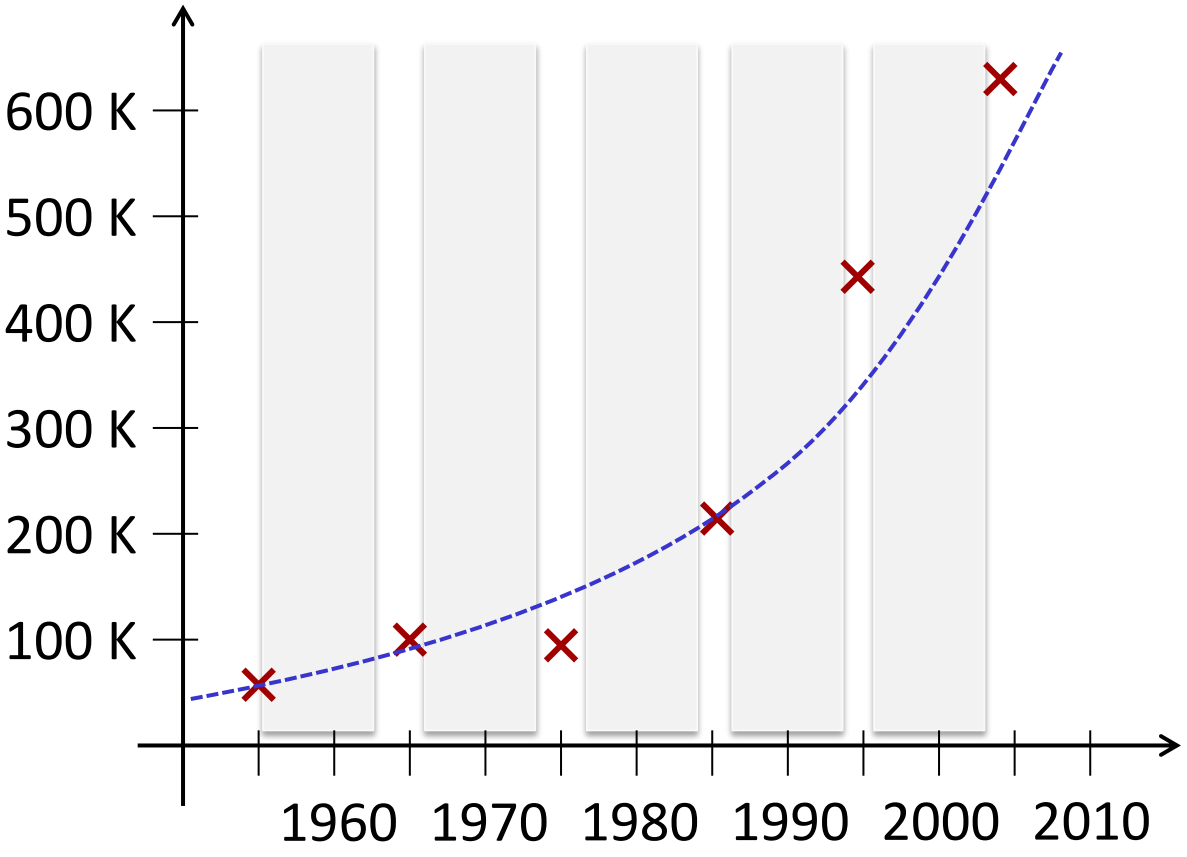
Example: Regression

Housing Prices in Springfield



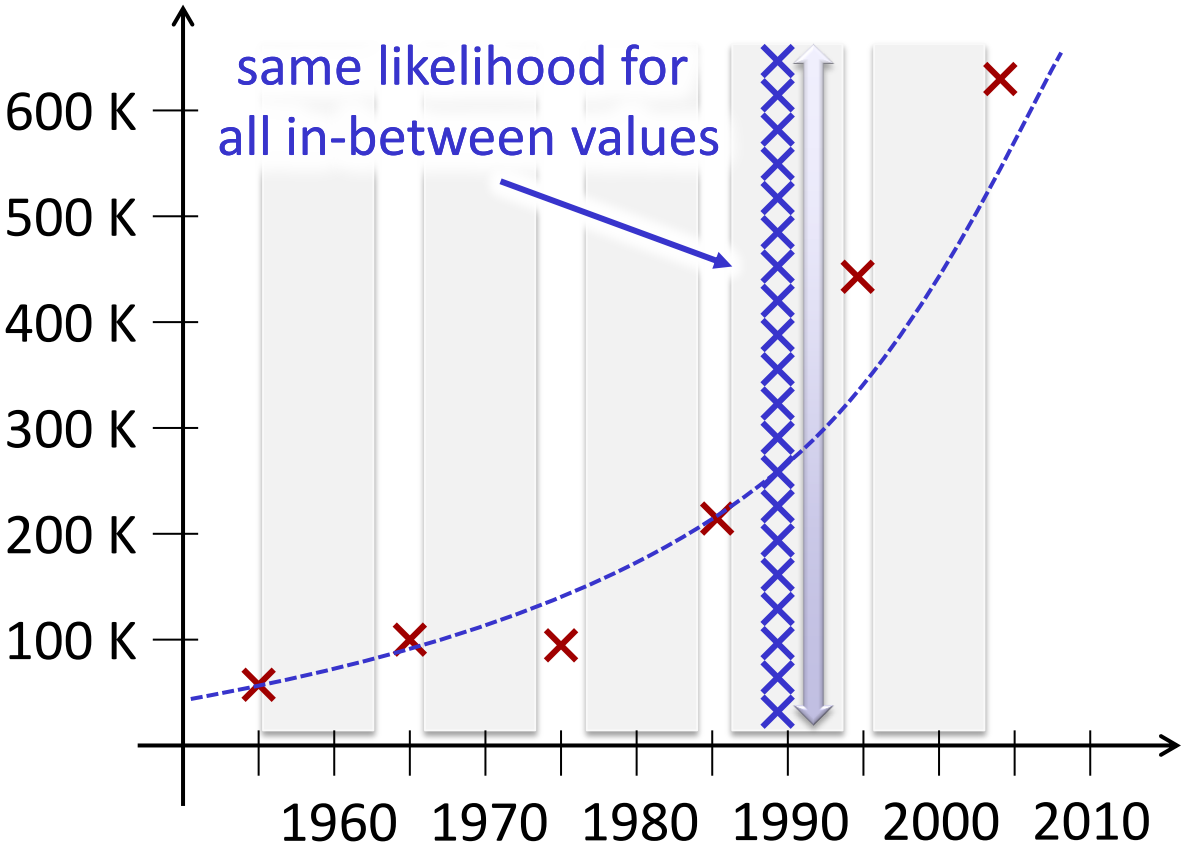
Example: Regression

Housing Prices in Springfield



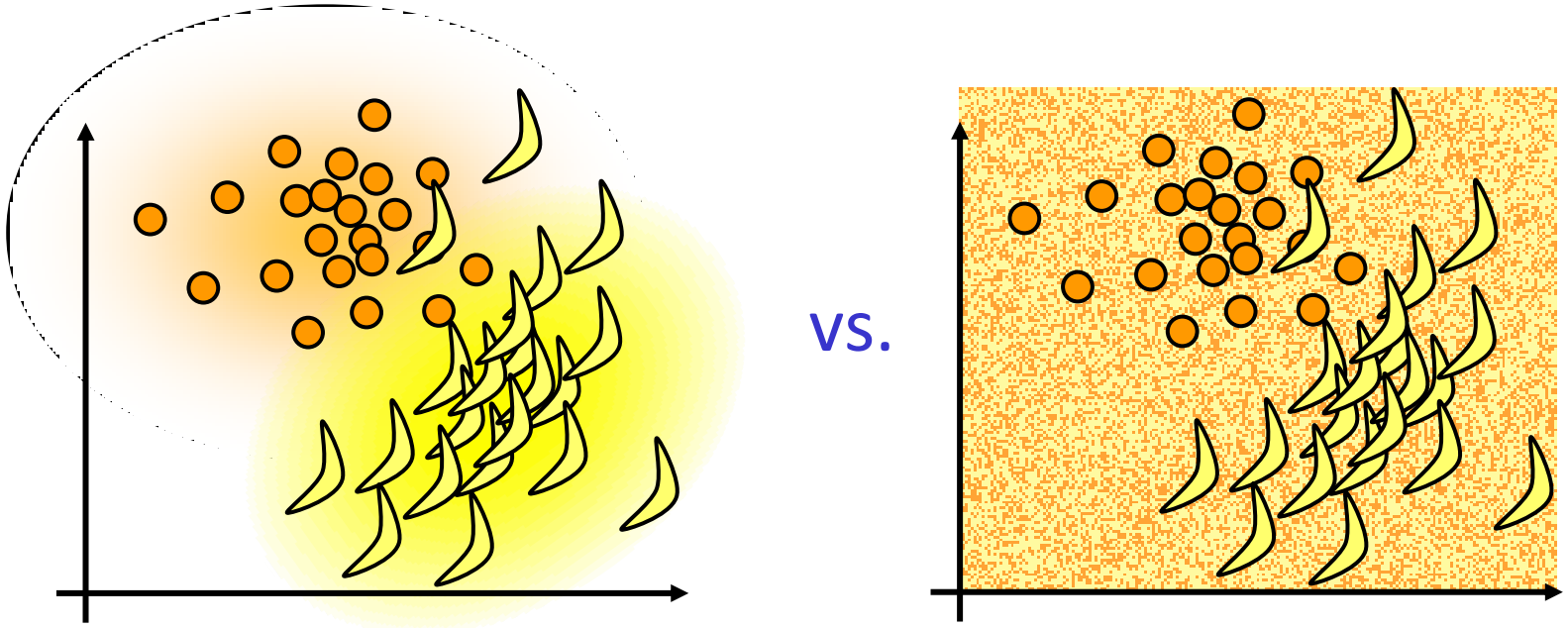
Example: Regression

Housing Prices in Springfield



Example: Density Estimation

Relativity of Orange-Banana Spaces



"smooth densities"
In this case: Gaussians

Significance and Capacity

Scenario

- We have a two hypothesis h_0, h_1
- One is correct

Solution

- Choose the one with higher likelihood

Significance test

- For example: Does new drug help?
- h_0: Just random outcome
- Show that $P(h_0)$ is small

Machine Learning: Capacity

We have:

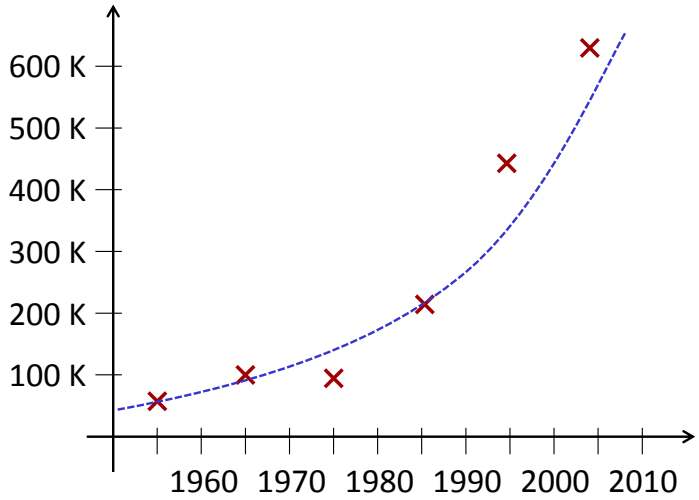
- Complex models

Example

- Polynomial fitting
- d continuous parameters a_i

$$p(x) = \sum_{i=0}^{d-1} a_i x^i$$

- "Capacity" grows with d



Significance?

Simple criterion

- Model must be able to predict training data
- Order d – 1 polynomial can always fit d points perfectly
  - Credit card numbers: 16 digits, 15-th order polynomial?
- Need O(d) training points at least
  - Random sampling: Overhead
  - d bins need O(d log d) random draws
- Rule of thumb “10 samples per parameter”

Simple Model

Single Hypothesis

- Hypothesis $h: \mathbb{R}^d \to \{0,1\}$, maps features to decisions
  Groud truth $g: \mathbb{R}^d \to \{0,1\}$, correct labeling
- Stream of data, drawn i.i.d.

$$(\mathbf{x}_i, y_i) \sim \mathcal{D}$$
$$g(\mathbf{x}_i) = y_i$$

  drawn from fixed distribution $\mathcal{D}$.
- Expected error:

$$\epsilon(h) = P_{\mathcal{D}}\big(h(\mathbf{x}) \neq g(\mathbf{x})\big)$$

Simple Model

Empirical vs. True Error

- Inifinte stream $(\mathbf{x}_i, y_i) \sim \mathcal{D}$, drawn i.i.d.
- Finite training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \sim \mathcal{D}$, drawn i.i.d.
- Expected error:

$$\epsilon(h) = P_{\mathcal{D}}\big(h(\mathbf{x}) \neq g(\mathbf{x})\big)$$

- Empirical error (training error):

$$\hat{\epsilon}(h) = \frac{1}{n}\sum_{i}^{n}\big(h(\mathbf{x}_i) - g(\mathbf{x}_i)\big)$$

- **Bernoulli experiment:** Chernoff bound

$$P(|\hat{\epsilon}(h) - \epsilon(h)| > \gamma) \leq 2\exp(-2\gamma^2 n)$$

Simple Model

Empirical vs. True Error

- Finite training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \sim \mathcal{D}$, drawn i.i.d.
- Training error bound:

$$P(|\hat{\epsilon}(h) - \epsilon(h)| > \gamma) \leq 2\exp(-2\gamma^2 n)$$

Result

- Reliability of *assessment* of *hypothesis quality* grows quickly with increasing number of trials
- We can bound generalization error

Machine Learning

We have multiple hypothesis

- Multiple hypothesis $\mathcal{H} = \{h_1, \dots, h_k\}$
- Need a bound on generalization error estimate for *all of them* after n training examples

$$P(\exists h_i \in \mathcal{H} : |\epsilon(h_i) - \hat{\epsilon}(h_i)| > \gamma)$$
$$= P([h_1 \text{ breaks}] \cup \cdots \cup [h_k \text{ breaks}])$$
$$\leq kP([h_i \text{ breaks}])$$
$$= 2k \exp(-2\gamma^2 n)$$

Machine Learning

Result

- After n training examples, we now training error up to γ *uniformly* for k hypothesis with probability of at least

$$1 - 2k \exp(-2\gamma^2 n)$$

- With probability of at least $1 - \delta$ sufficient to use

$$n \geq \frac{1}{2\gamma^2} \log \frac{2k}{\delta} \text{ (log in } k\text{)}$$

  training examples.

- With probability $1 - \delta$, error bounded by

$$\forall h_i \in \mathcal{H}: |\epsilon(h_i) - \hat{\epsilon}(h_i)| \leq \sqrt{\frac{1}{2n} \log \frac{2k}{\delta}}$$

Empirical Risk Minimization

ERM Learning Algorithm

- Evaluate all hypothesis $\mathcal{H} = \{h_1, \ldots, h_k\}$ on training set
- Choose $\hat{h}$ with lowest error,

$$\hat{h} = \arg\min_{i=1..k} \hat{\epsilon}(h_i)$$

- "Empirical Risk Minimization"

Empirical Risk Minimization

Guarantees

- When using empirical risk minimization
- With probability $\geq\ 1-\delta$, we get:
  - Not far from optimum:

$$\epsilon(\hat{h}) \leq \epsilon(h_{best}) + 2\gamma$$

- Trade off:

$$\epsilon(\hat{h}) \leq \underbrace{\epsilon(h_{best})}_{\text{Bias}} + \underbrace{2\sqrt{\frac{1}{2n}\log\frac{2k}{\delta}}}_{\text{Variance}}$$

Generalization

Can be generalized

- For multi-class learning, regression, etc.

Continuous set of hypothesis

- Simple: k bits encode hypothesis
- More sophisticated model:
  Vapnik-Chervonenkis (VC) dimension
  - „Capacity“ of classifier
  - Max. number of points that can be labled differently by hypothesis set
  - $\mathcal{O}\left(VC(\mathcal{H})\right)$ training examples needed

Conclusion

Two theoretical insights

- **No free lunch:**
  Without additional information,
  no prediction possible about off-training examples
- **Significance:** Yes, we can...
  - ...estimate expected generalization error with high probability
  - ...choose a good hypothesis from a set (with h.p. / error bounds)

Conclusion

Two theoretical insights

- There is no contradiction here
  - Still, some non-training points might be misclassified all the time
  - But they cannot show up frequently
  - Have to choose hypothesis set
    - Infinite capacity leads to unbounded error
    - Thus: We do need prior knowledge

Conclusions

Machine Learning

- Is basically density estimation
- Curse of dimensionality
  - High dimensionality makes things intractable
  - Model dependencies to fight the problem

Conclusions

Machine Learning

- No free lunch
  - You can only learn when you already know something
  - Math won't tell you were knowledge initially came from
- Significance
  - Beware of overfitting!
  - Need to adapt plasticity of model to available training data

Recommended Further Readings

Short intro:

- **Aaron Hertzman:** Siggraph 2004 Course
  "Introduction to Bayesian Learning"
  http://www.dgp.toronto.edu/~hertzman/ibl2004/

Bayesian learning, no free lunch:

- **R. Duda, P. Hart, D. Stork**: Pattern Classification, 2nd edition, Wiley.

Significance of multiple hypothesis:

- **Andrew Ng**, Stanford University
  "CS 229 – Machine Learning" Course notes (Lecture 4)
  http://cs229.stanford.edu/materials.html