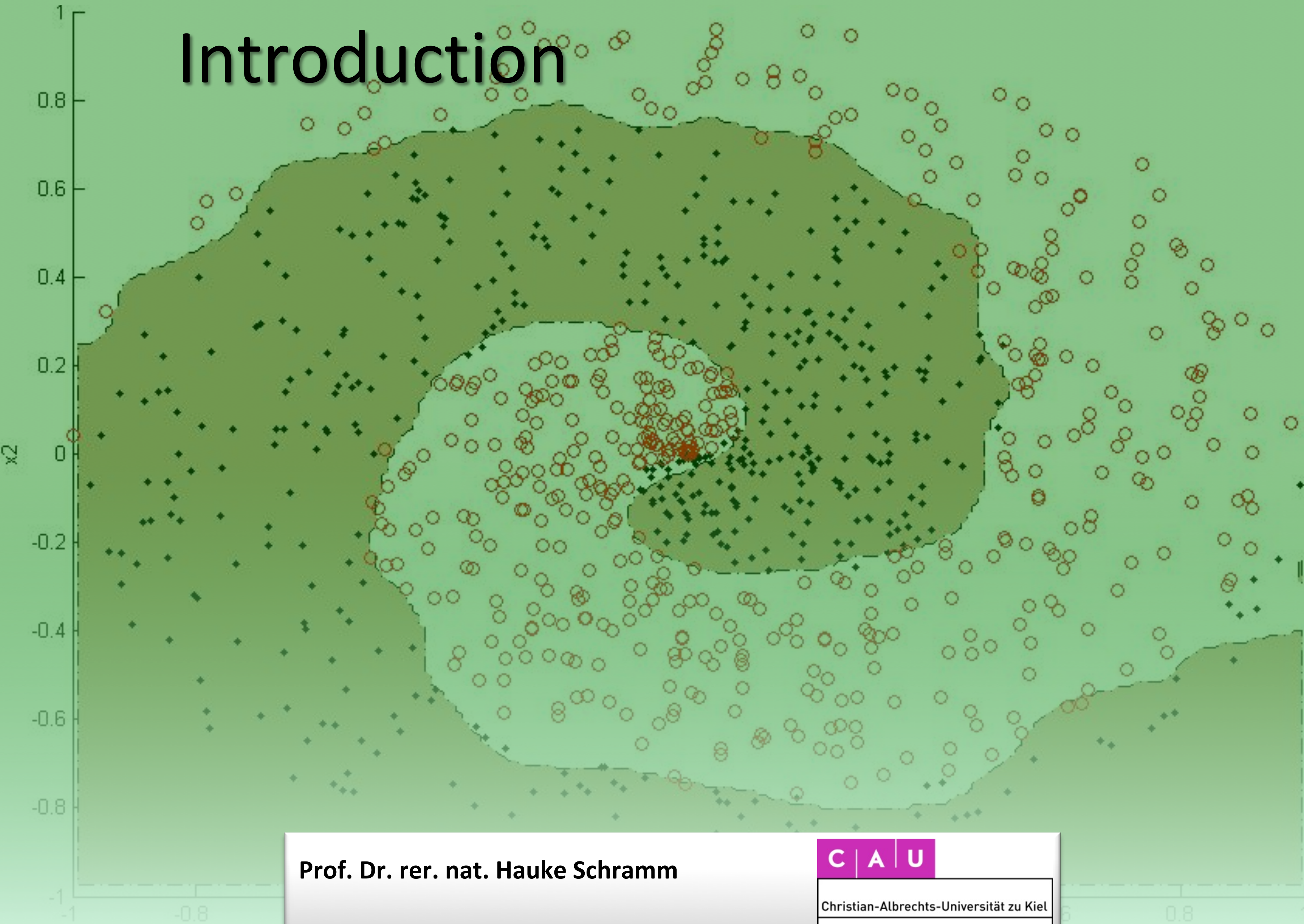
Introduction



Prof. Dr. rer. nat. Hauke Schramm

Christian-Albrechts-Universität zu Kiel

Institut für Informatik

An example-based introduction

Automation of fish sorting process in a fish packing plant:

- sea bass
- salmon

Camera-based process

An example

Physical differences:

- length
- width
- lightness
- mouth position
- ...

True differences between the two fish types.

→ ***Features*** for use in our classifier

Learn an individual description (model) for each fish population that captures statistics of the used features.

An example

Additional differences due to

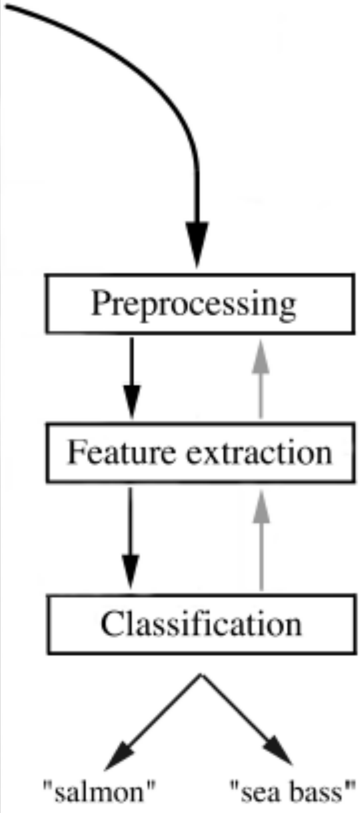
- lighting variations
- position of the fish
- camera electronics
- ...

Must be eliminated *as good as possible* in a preprocessing step.

An example

Fish classification

- Capture image and eliminate irrelevant information
- Compare observed features with both models
- Select the model that corresponds best as classification result.



An example

Let us go more into detail:

Assume, we knew (from somewhere) that sea bass are longer than salmon.

How could we use this information for classification?

→ Define a length threshold (How? Later!)

→ Measure length of each fish

→ If length exceeds the threshold: sea bass

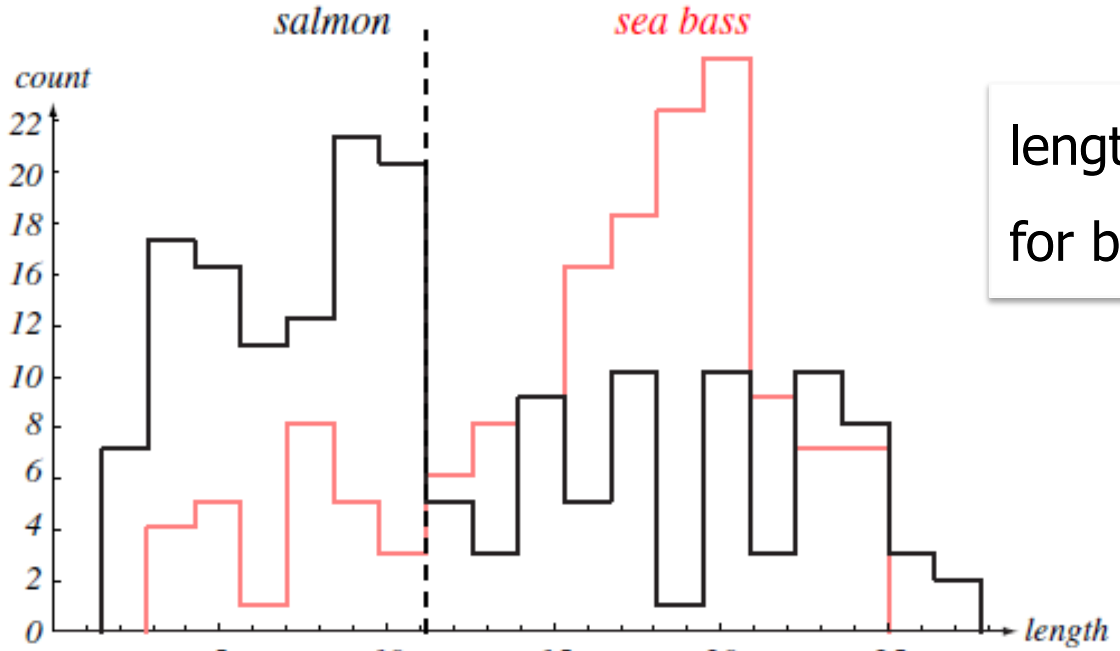
Otherwise: salmon

decision boundary

An example

How can we find an 'optimal' threshold?

- Collect (many) training images of the different types of fish
- Make length measurements and inspect results



length histogram for both classes

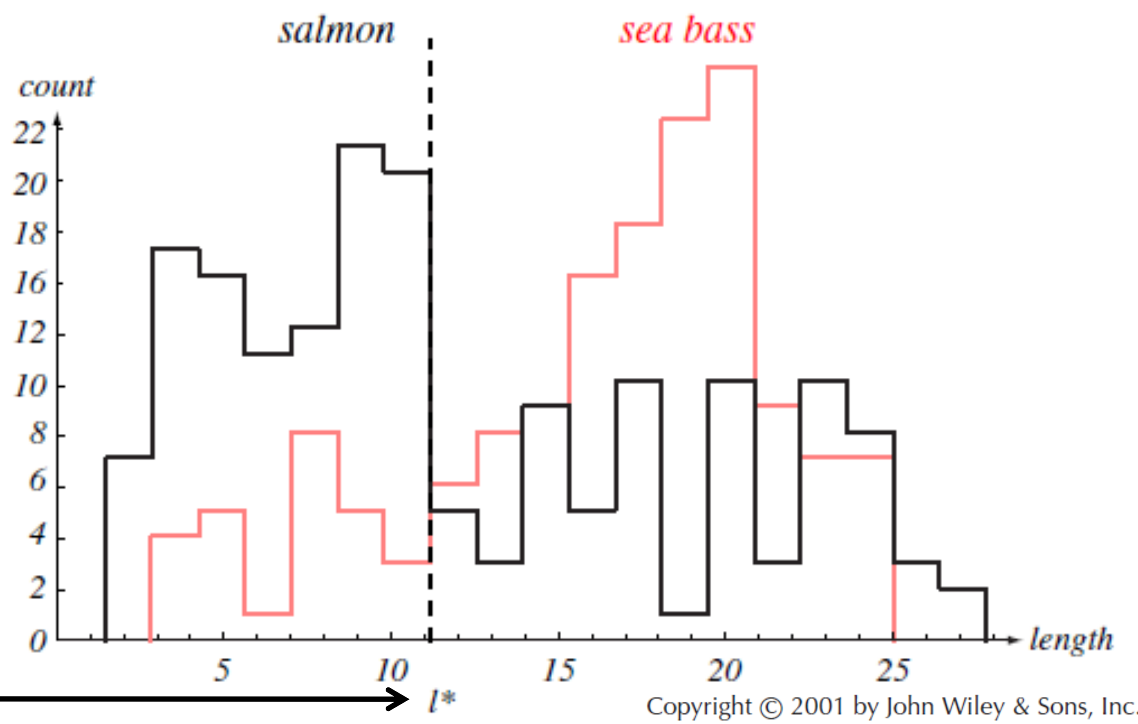
decision boundary

l^*

An example

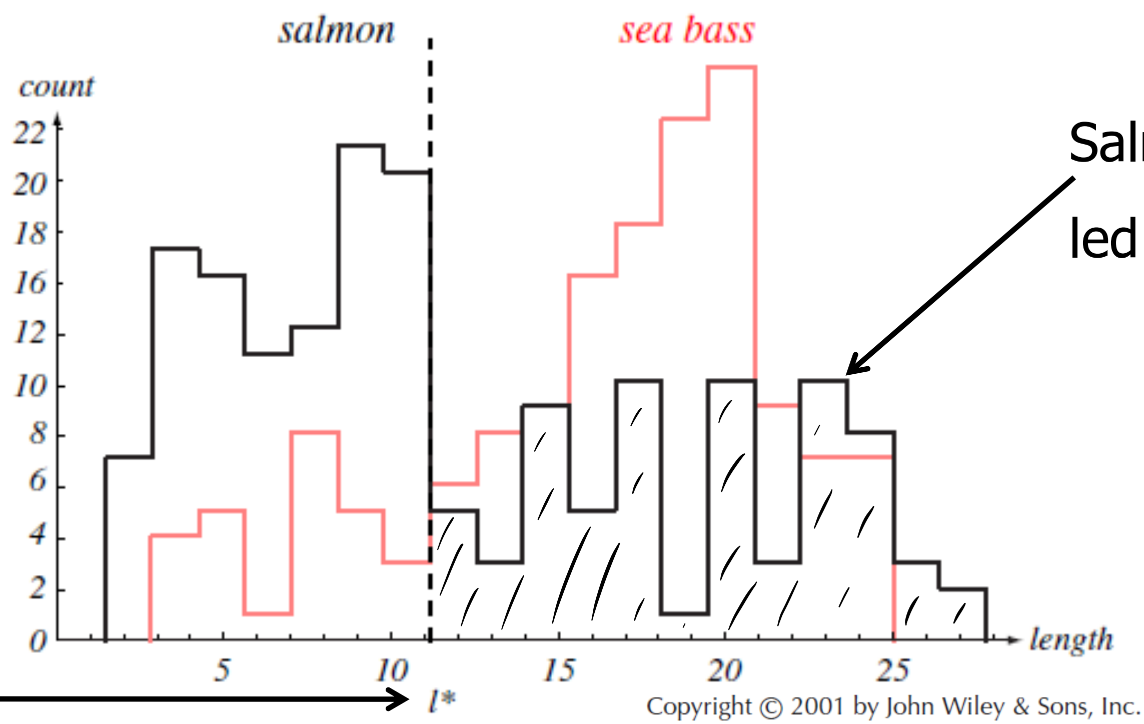
Value marked l* will lead to smallest number of errors on average

→ however, result is still unsatifying



decision boundary

An example



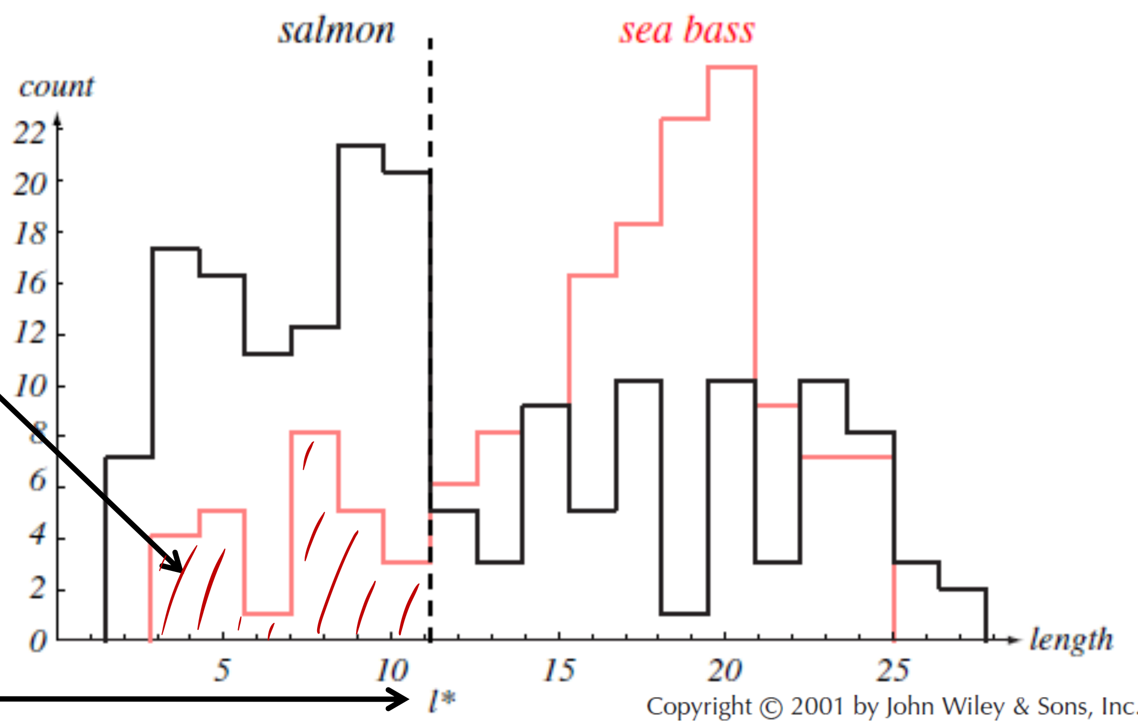
Salmons label-
led as sea bass

decision
boundary

An example

Sea bass label-led as salmon

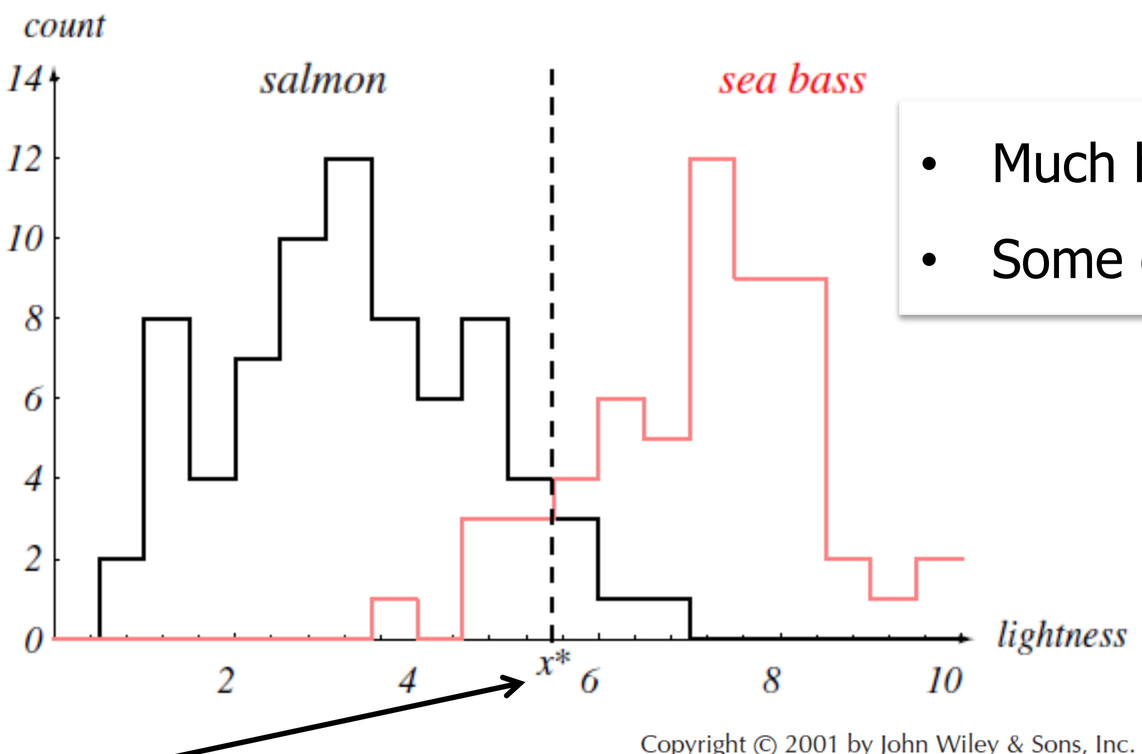
decision boundary



An example

Fish length is a poor feature for our classification task

→ try another one: lightness of the fish scales (German: Fischschuppen)

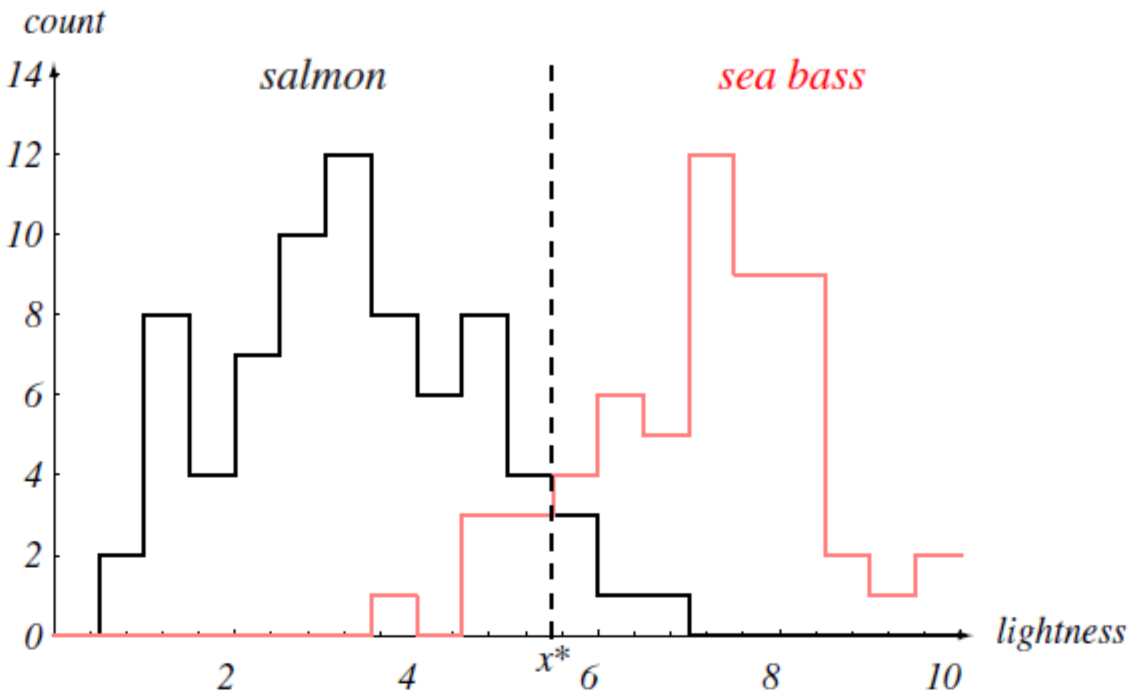


- Much better separation
- Some errors still remain

decision boundary

An example

Given decision boundary leads to minimum error on average if **costs** for both types of classification errors **are the same**.



An example

Example for unsymmetric cost distribution:

- Customers may accept pieces of tasty salmon in their sea bass cans

→ Assignment of low or zero costs to this type of classification error

- Customers will not accept pieces of sea bass in their cans labelled 'salmon'

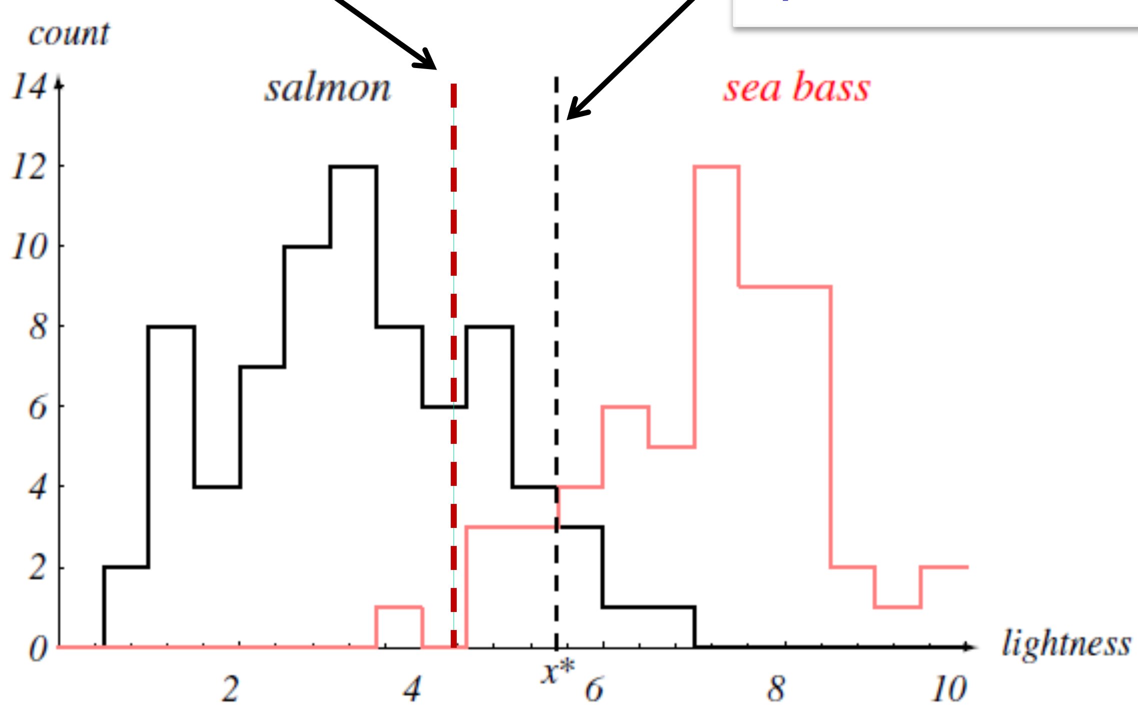
→ Assignment of high costs to this type of classification error

→ Decision boundary for this unsymmetric distribution of costs?

An example

Decision boundary with un-symmetric cost distribution

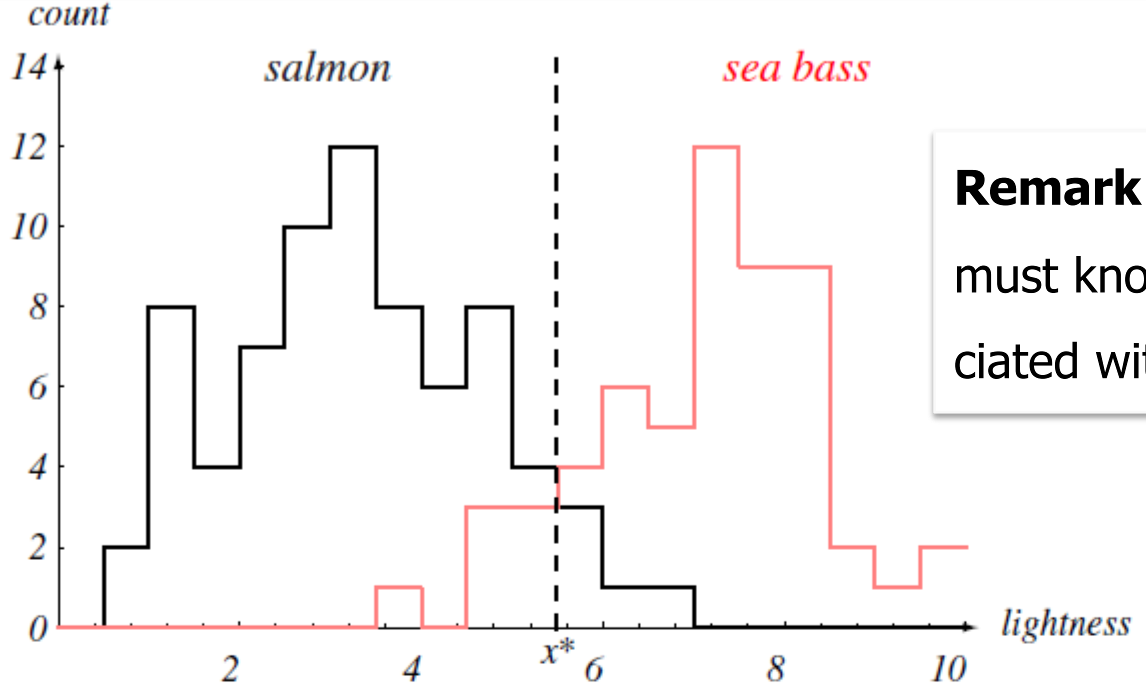
Decision boundary with symmetric cost distribution



An example

Task of **decision theory**:

Make a decision rule (i.e. set decision boundary) that minimizes costs associated with a specific classification task.



Remark: To this end we must know the costs asso-ciated with our decisions.

An example

Assume, we are still not satisfied with classification performance.

What could we do in order to improve the classifier?

Use more than one feature at a time!

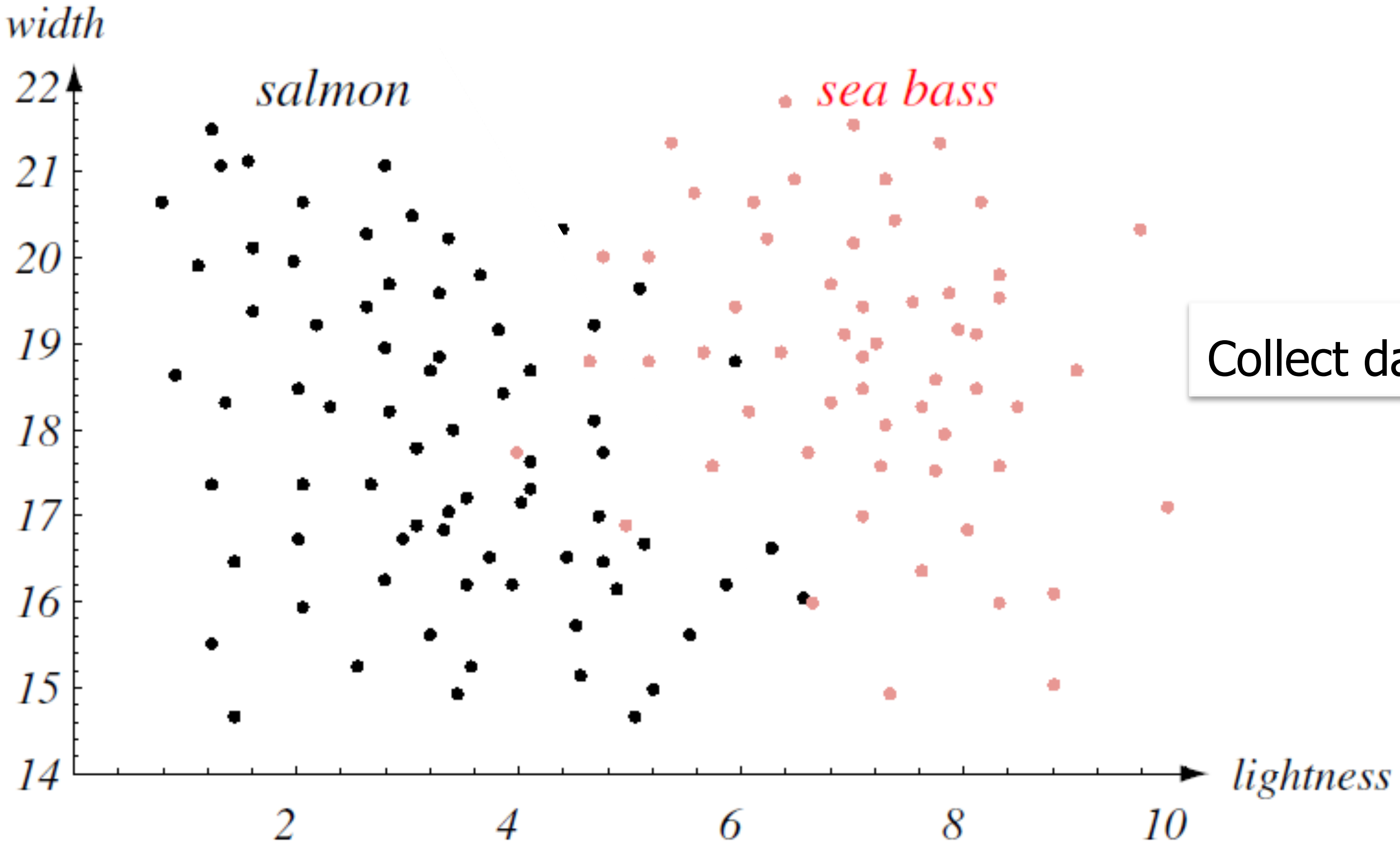
Length x_1

Lightness x_2

Feature **vector** $\vec{x} = \binom{x_1}{x_2}$

An example

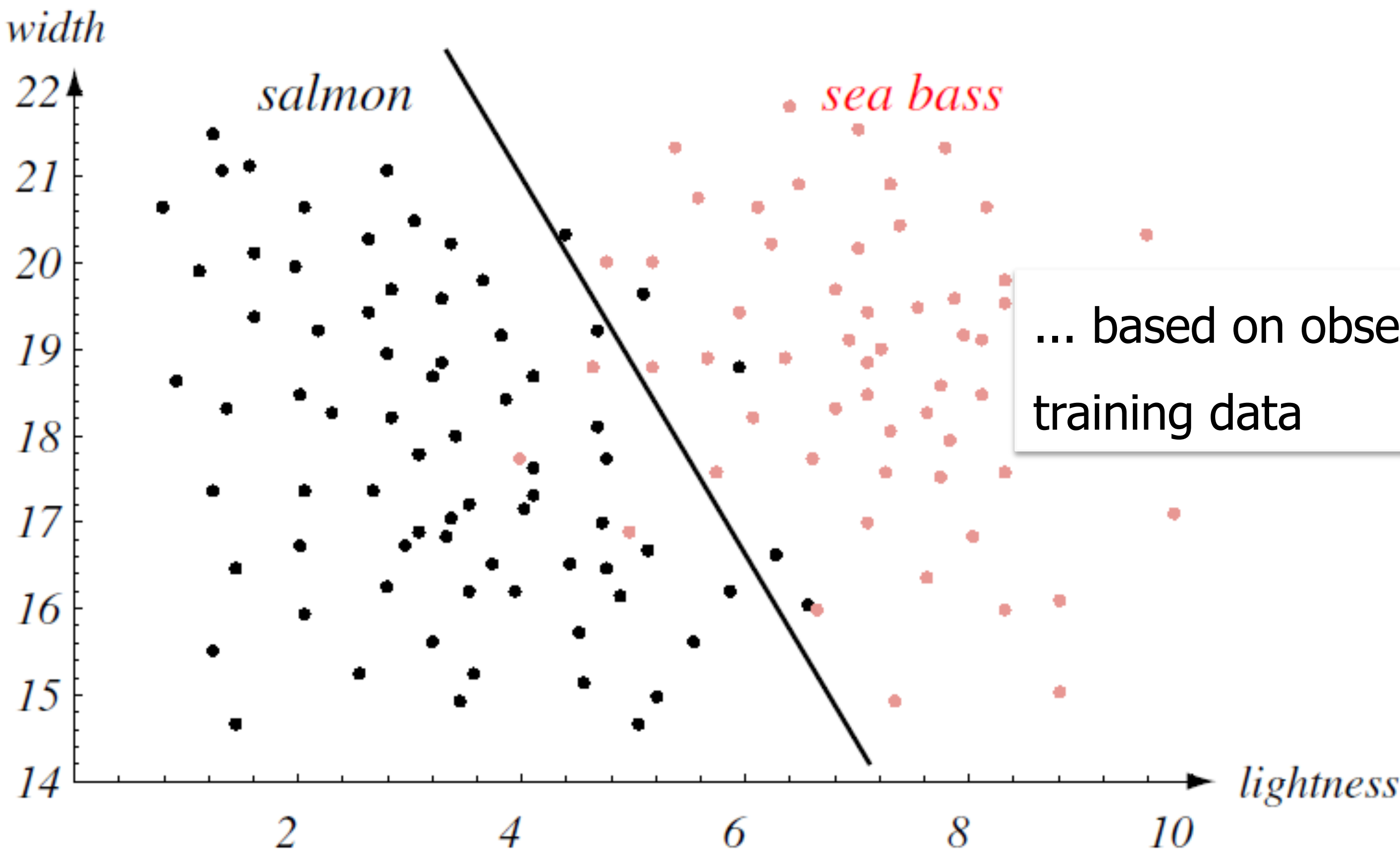
Each fish is then represented by a **2-D feature vector**



Collect data

An example

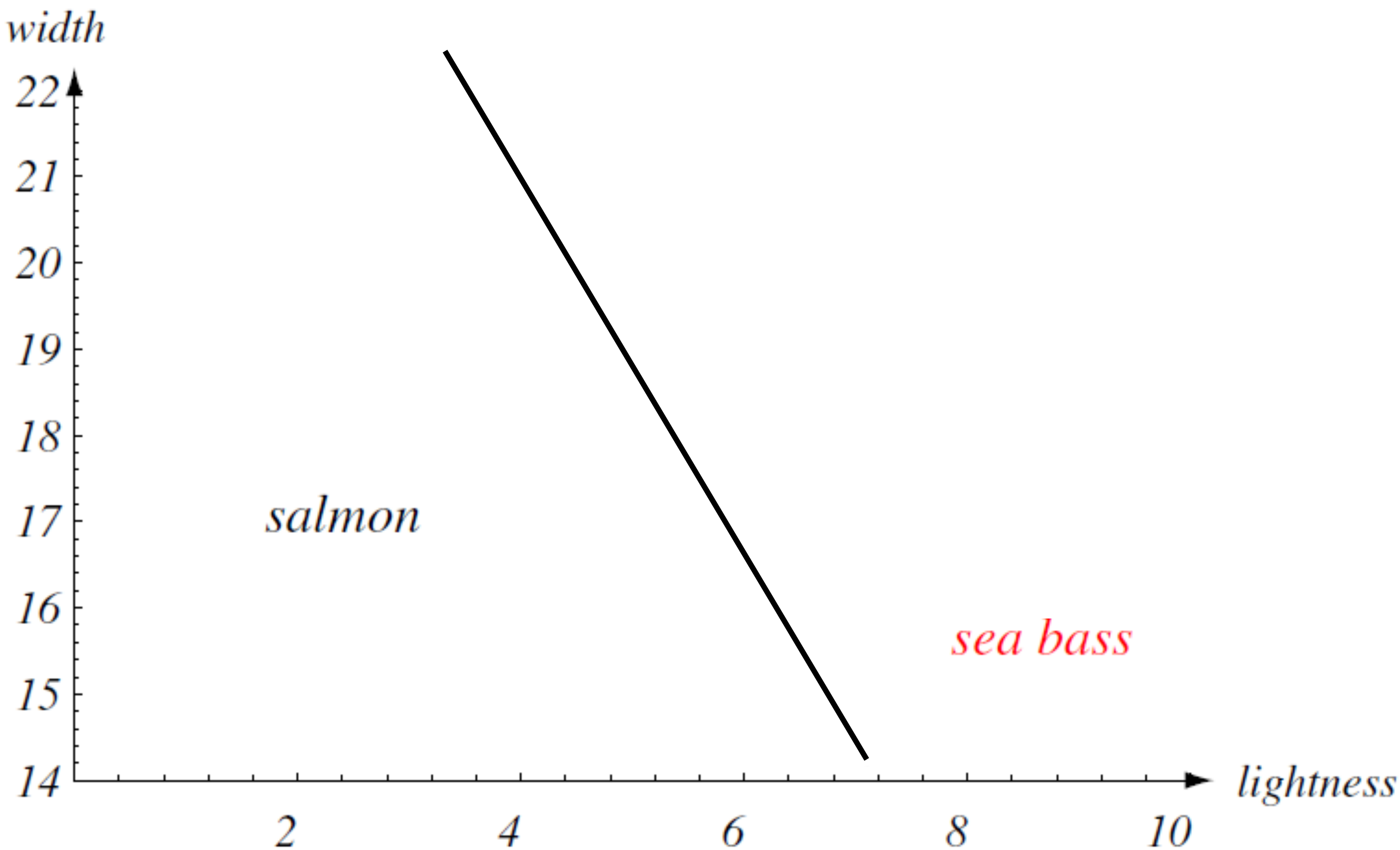
Classification now means to **partition the feature space into two regions**



... based on observed training data

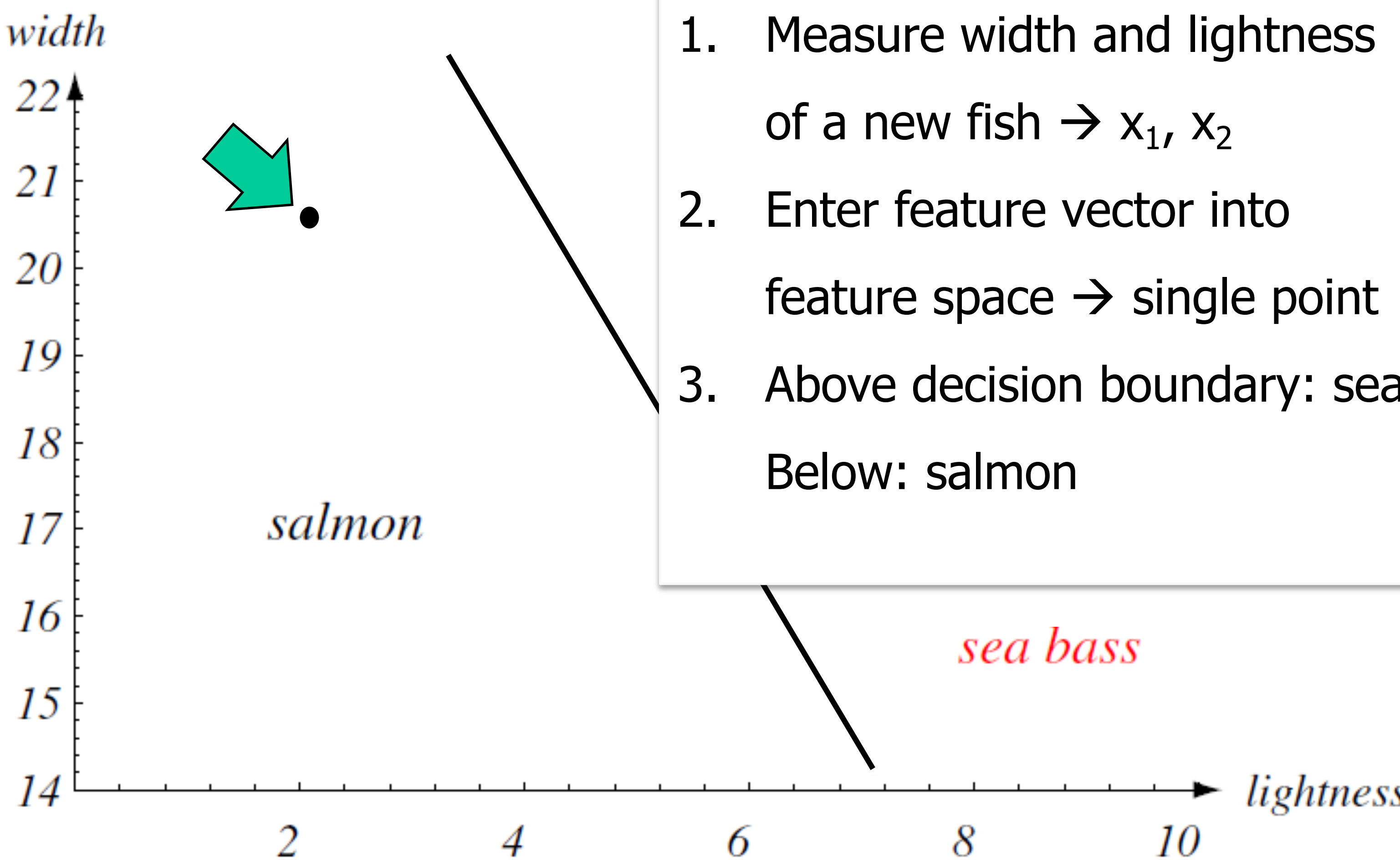
An example

... and assign class label 'sea bass' or 'salmon' to an unknown fish



An example

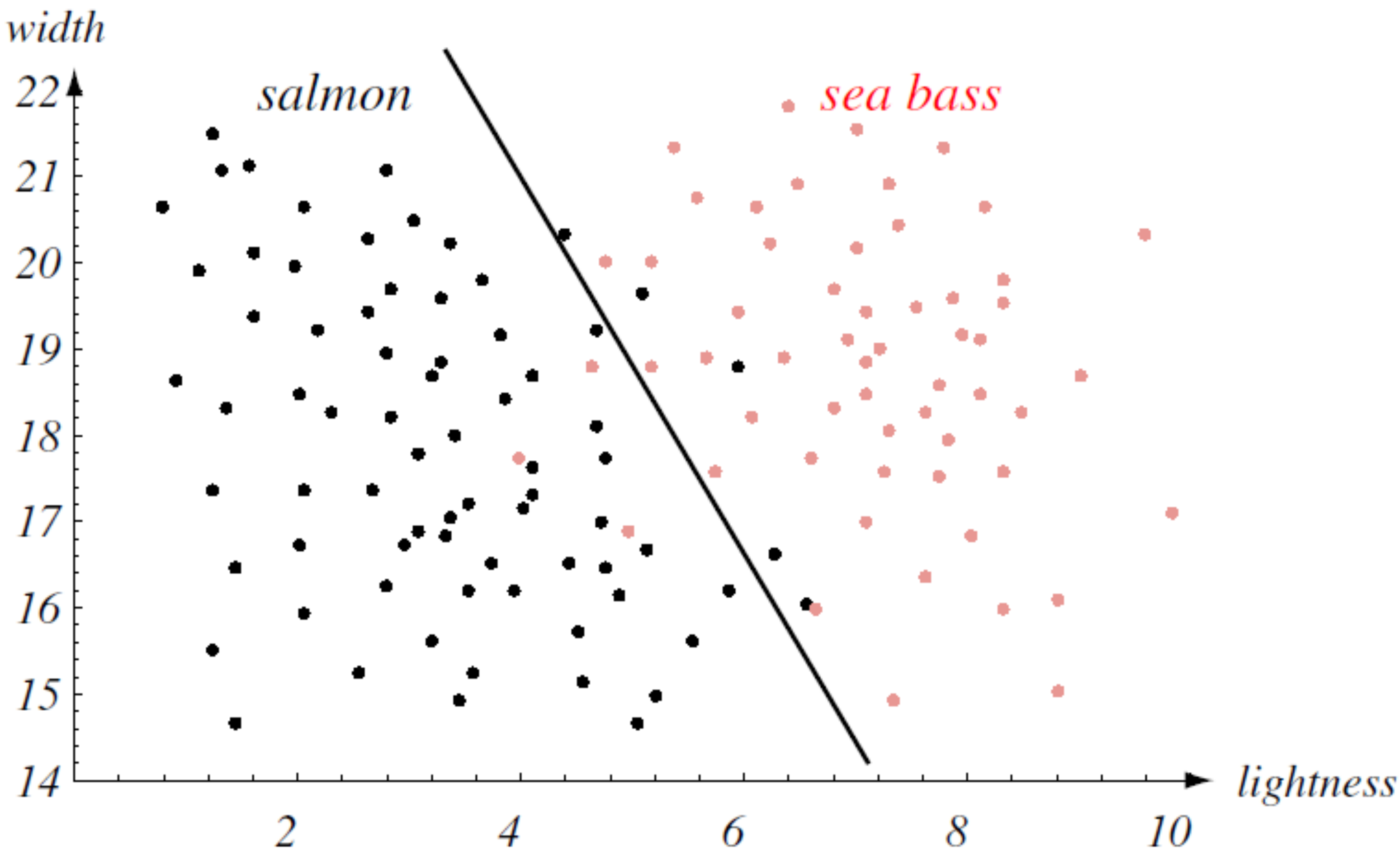
... and assign class label 'sea bass' or 'salmon' to an unknown fish



1. Measure width and lightness of a new fish → x_1, x_2
2. Enter feature vector into feature space → single point
3. Above decision boundary: sea bass
   Below: salmon

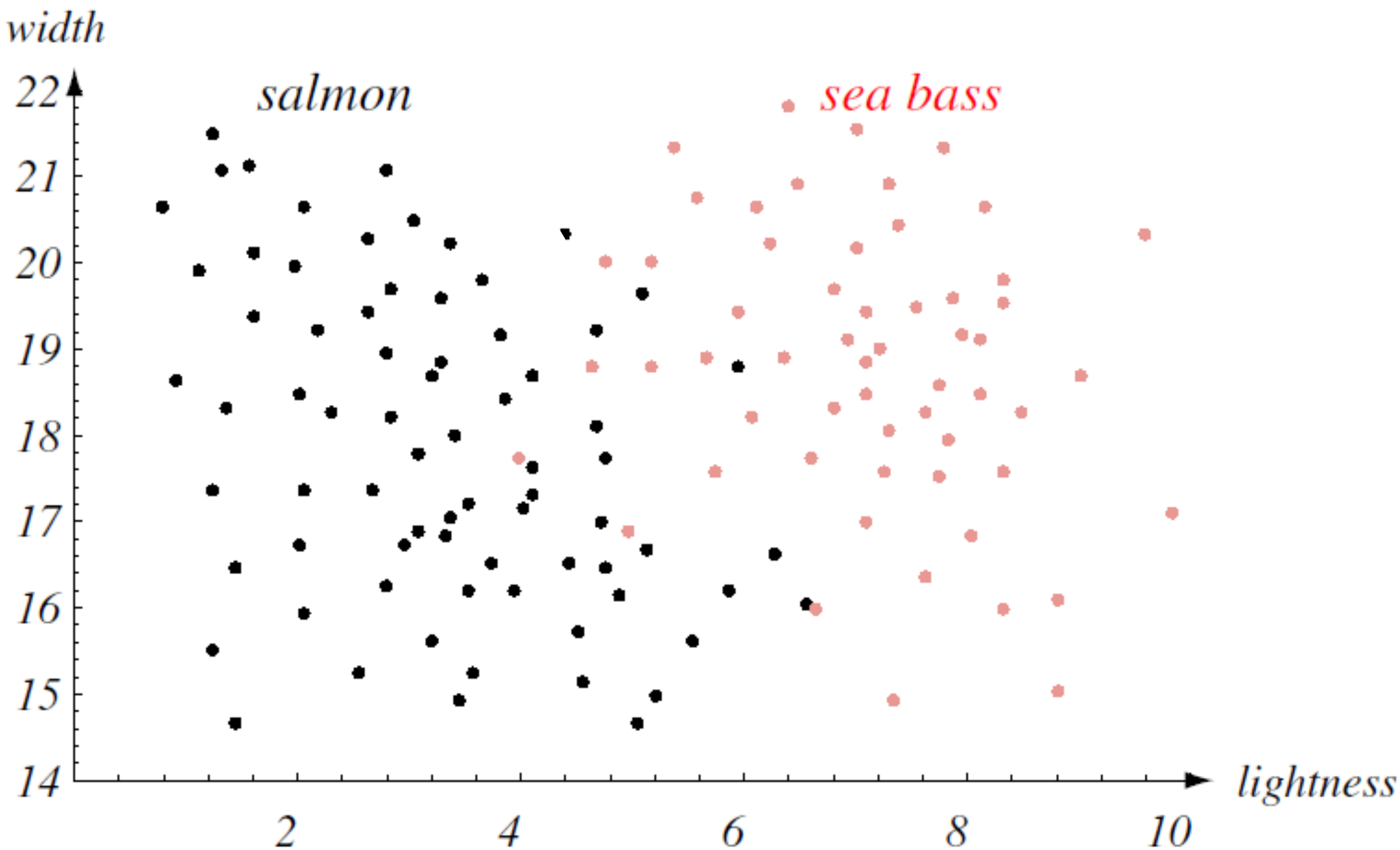
An example

Question: Good decision boundary?



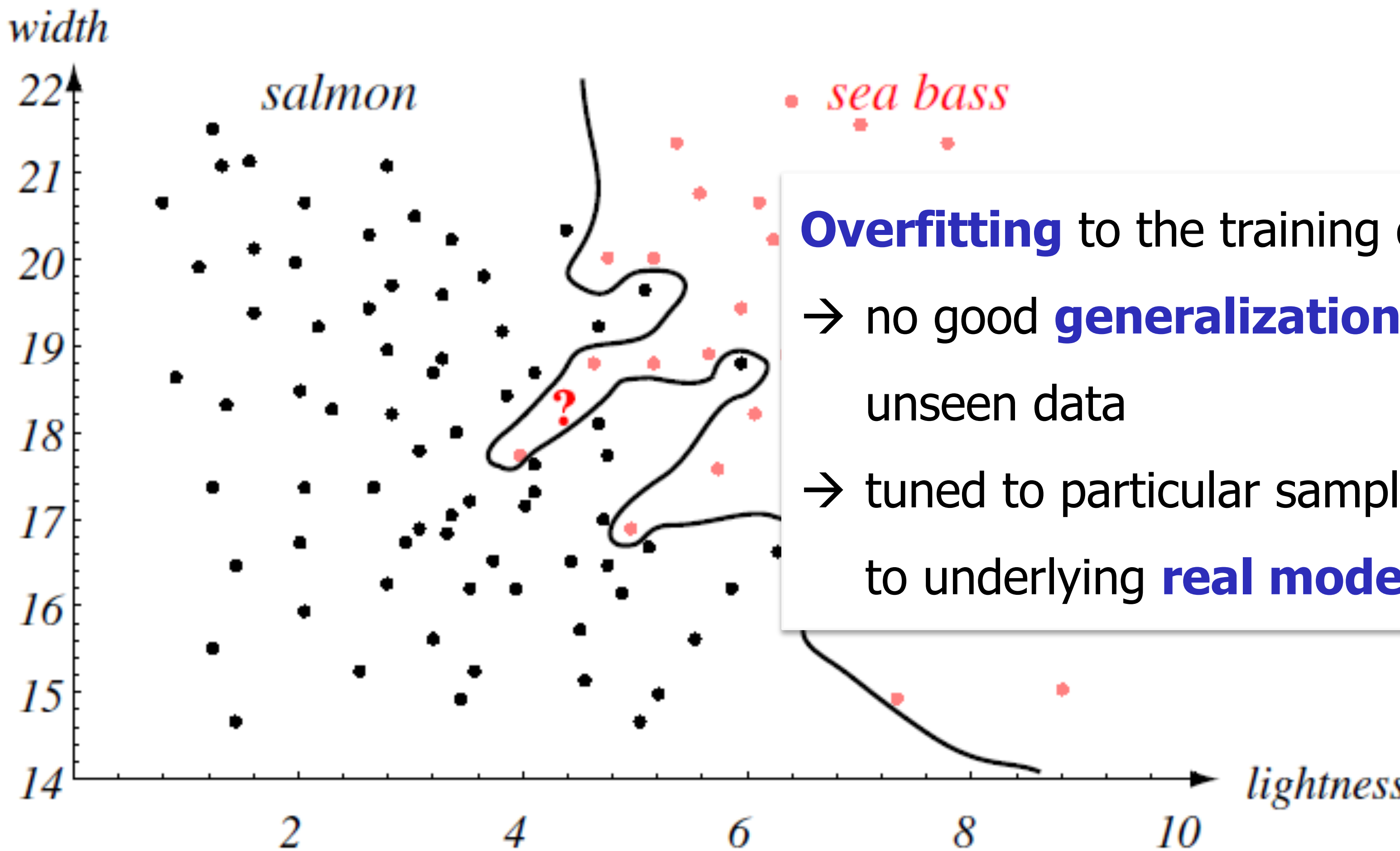
An example

Would that be a better decision boundary? Why / why not?



An example

Would that be a better decision boundary? Why / why not?



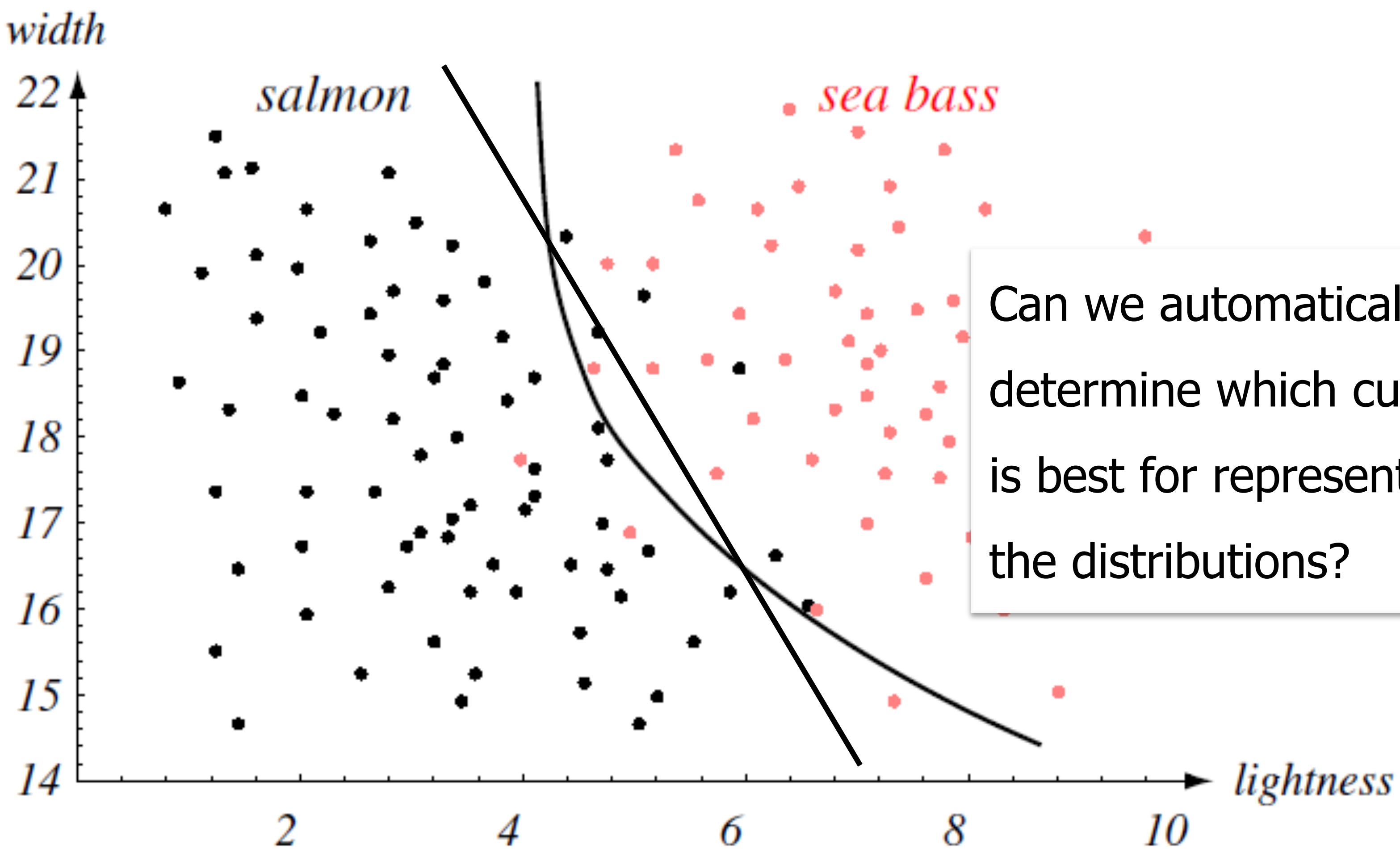
Overfitting to the training data

→ no good **generalization** to unseen data

→ tuned to particular samples, not to underlying **real models**

An example

Would that be a better decision boundary? Why / why not?



Can we automatically determine which curve is best for representing the distributions?

An example

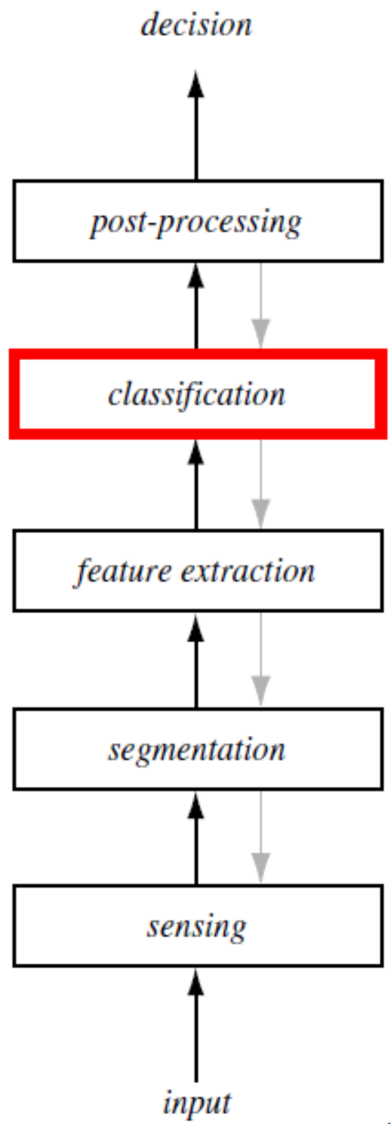
Leads us to some central problems in statistical pattern recognition:

- Sufficient amount of training data
- Accurate model design
- Generalization to unknown patterns
- ...

Other important aspects of classifier design apart from pure misclassification:

- robustness of features to noise
- processing time

Overview of a Pattern Recognition System



post-processing: recommend actions that minimize total expected costs (risk)

incorporation of background knowledge and costs into the decision, combination of multiple classifiers

classification: assign object to a category

fish classification: salmon, sea bass
speech recognition: word identity

feature extraction: measure a representation of the object that simplifies classification

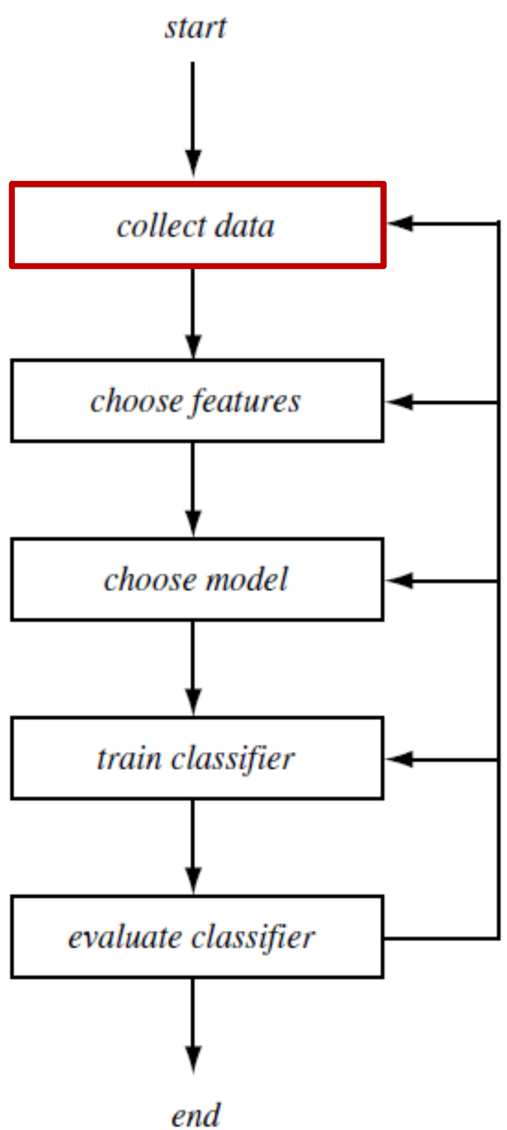
fish classification: length and lightness
speech recognition: spectral measurements

segmentation: separation of individual patterns

fish classification: begin and end of a fish
speech recognition: word or phoneme separation

sensing: camera, microphone
→ beyond scope of lecture

The Design Cycle of a Pattern Recognition System

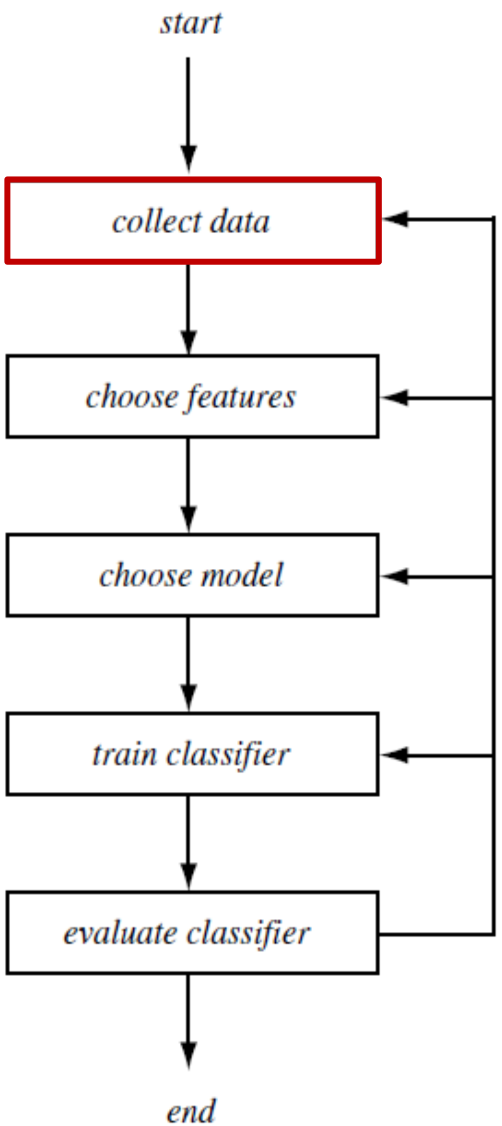


The Design Cycle of a Pattern Recognition System

Data collection:

- Data is required for
  - estimating model parameters
  - evaluating the trained system
- Very important and expensive development step
- 'There is no data like more data'

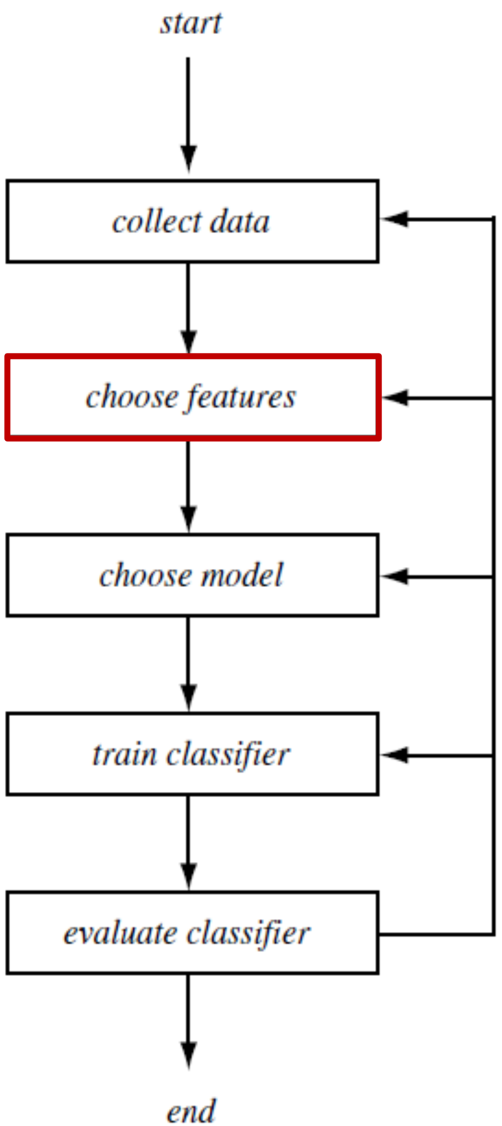
Question: When do we have 'enough' data?



The Design Cycle of a Pattern Recognition System

Feature selection:

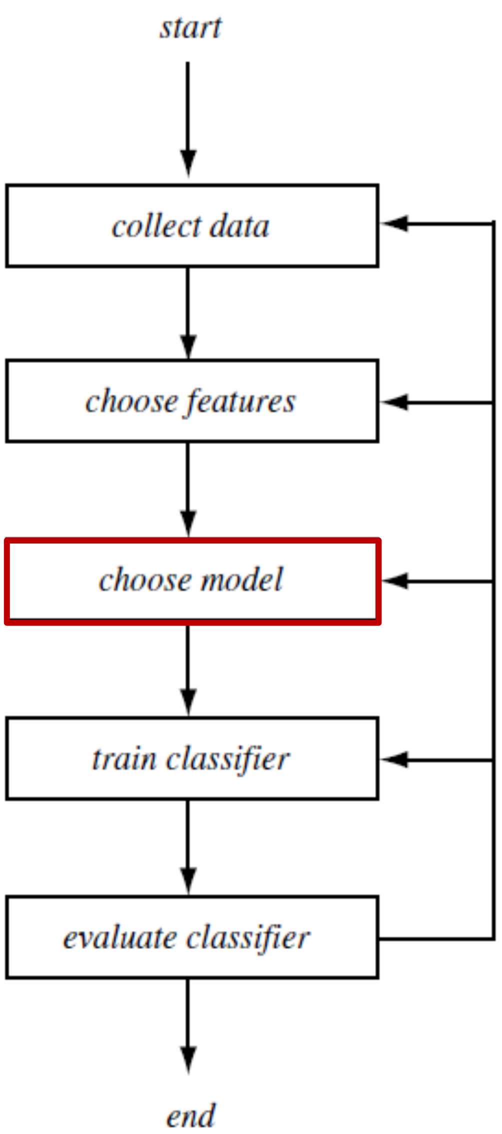
- Very critical step for system performance
- Decision typcially based on prior knowledge
  - taken from examples or expert knowledge
  - example: sea bass is longer than salmon
  - face recognition: face has 1 nose, 2 eyes, ...
- Optimal features:
  - easy to extract
  - invariant to irrelevant information
  - useful for *discriminating the categories*



The Design Cycle of a Pattern Recognition System

Model selection:

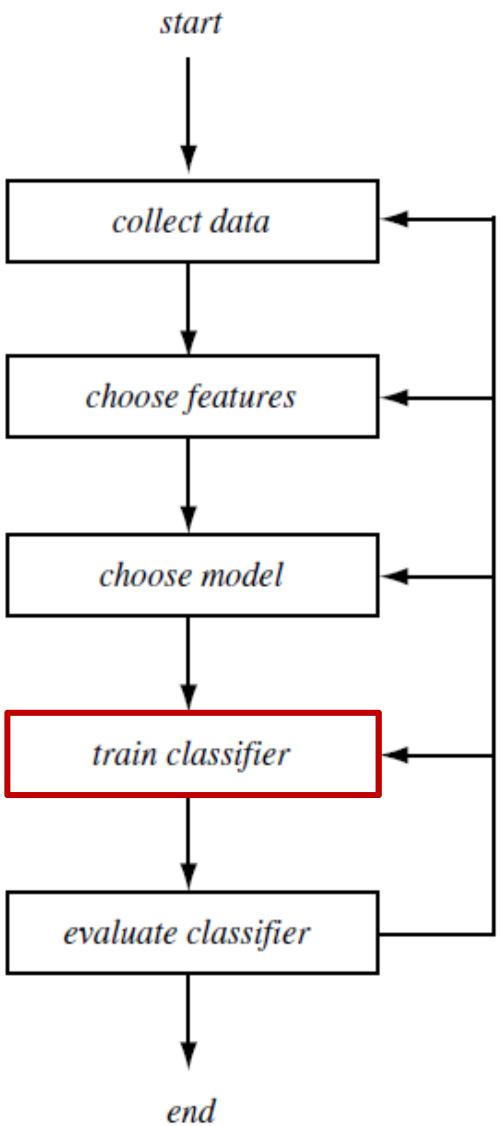
- Model: (mathematical) description of the adressed categories
- Example:
  - Gaussian or Laplacian distributions?
  - single or multi-modal?
  - with full or diagonal covariance matrix?
- Requires substantial experience of system designer
- In praxis: often based on trial and error



The Design Cycle of a Pattern Recognition System

Training:

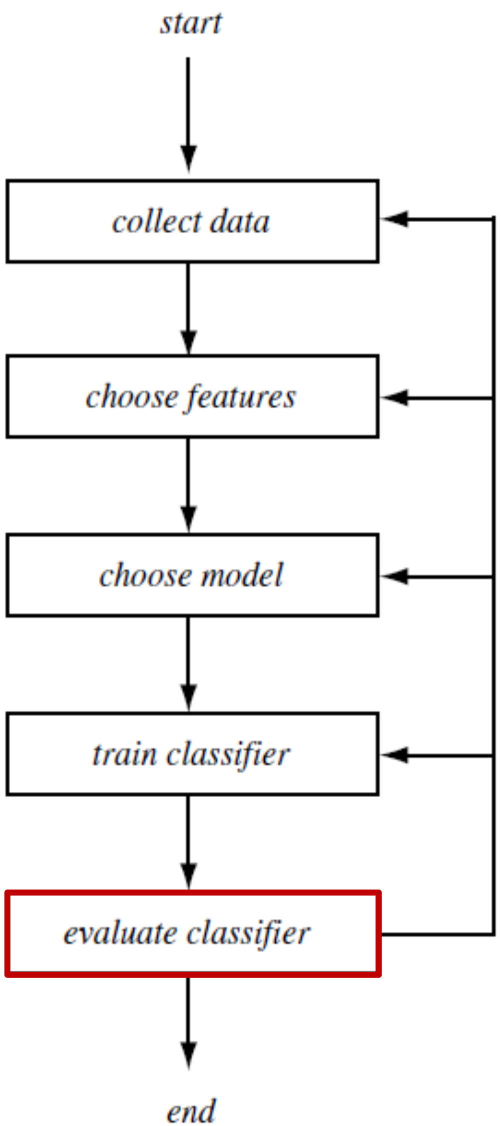
- Usage of data for determining model parameters
- Learning from example patterns
- Example: Collecting statistics of fish length and lighting from training images



The Design Cycle of a Pattern Recognition System

Evaluation:

- Measure system performance
- Evaluation on
  - (known) training data
  - (unknown) test data
- Error analysis unveils weak modeling components



Conclusion

- Automatic pattern classification is a complex process with a large number of complex subproblems
- Subproblems are highly interrelated
- Good news:
  - Problem can be solved (humans)
  - Powerful mathematical solutions to some of the problems exist
  - Much to improve!