Image Analysis (FMAN20) Lecture 5, 2021

MAGNUS OSKARSSON

Image Analysis - Motivation

Overview – Machine Learning 1

1. Machine Learning
2. Bayes Theorem
    1. Counting
    2. Binning
    3. Curse of dimensionality
    4. Adaptive binning (K-means)
3. Nearest Neighbour, K-NN

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

Machine Learning – Bayes rule

Assume that one feature vector **x** and class y are drawn from a joint probability distribution. If one can calculate the probability that the class is $y = j$ given the measurements **x**, i.e. the so called **posterior probability**.

$$P(y = j|\mathbf{x})$$

The **maximum a posteriori classifier** is obtained as selecting the class j that maximizes the posterior probability, i.e.

$$j = \text{argmax}_k \, P(y = k|\mathbf{x}).$$

It is often easier to model and estimate the **likelihood** $P(\mathbf{x}|y = j)$ and to model the **prior** $p(y = j)$. The a posteriori probabilites can then be calculated using the Bayes rule,

$$p(y = j|\mathbf{x}) = \frac{p(\mathbf{x}|y = j)p(y = j)}{p(\mathbf{x})}.$$

Example

- In a small town, there are two bicycle brands **albatross** and **butterfly**. Albatross sell mostly **green** bikes (80 percent) are green and the rest are **yellow**. Butterfly sell 50 percent green bikes and 50 percent yellow. The Albatross brand is more popular. They have 90 percent of the market in the town. **In another nearby town**, from a distance you see a yellow bike. What is the probability that the bike is an Butterfly bike?

Joint probabilites P(x,y)

- Joint probability and the prior $P(\mathbf{x}) = \sum_{\mathbf{y}} P(\mathbf{x}, \mathbf{y})$

- Joint probability and the total probability $P(\mathbf{y}) = \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{y})$

$$P(\mathbf{y}) = \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{y})$$

P(x,y)	x=Green	x=Yellow	
y=Albatross	0,72	0,18	0,9 p(y=Albatross)
y=Butterfly	0,05	0,05	0,1 p(y=Butterfly)
	0,77	0,23	1
	p(x=Green)	p(x=Yellow)	

P(x|y)	x=Green	x=Yellow	
y=Albatross	0,80	0,20	1,00
y=Butterfly	0,50	0,50	1,00
	1,30	0,70	2,00

$$P(\mathbf{x}) = \sum_{\mathbf{y}} P(\mathbf{x}, \mathbf{y})$$

A posteriori probabilites

- A posteriori probabilites

$$P(\mathbf{y}|\mathbf{x}) = P(\mathbf{x}, \mathbf{y})/P(\mathbf{x})$$

P(x,y)	x=Green	x=Yellow	
y=Albatross	0,72	0,18	0,9 p(y=Albatross)
y=Butterfly	0,05	0,05	0,1 p(y=Butterfly)
	0,77	0,23	1
	p(x=Green)	p(x=Yellow)	

P(x|y)	x=Green	x=Yellow	
y=Albatross	0,80	0,20	1,00
y=Butterfly	0,50	0,50	1,00
	1,30	0,70	2,00

P(y|x)	x=Green	x=Yellow	
y=Albatross	0,94	0,78	1,72
y=Butterfly	0,06	0,22	0,28
	1	1	2

Usual workflow

- Model prior $P(\mathbf{y})$ and measurement probabilites $P(\mathbf{x}|\mathbf{y})$
- Joint probability $P(\mathbf{x},\mathbf{y}) = P(\mathbf{x}|\mathbf{y})P(\mathbf{y})$
- Total probability $P(\mathbf{x}) = \sum_{\mathbf{y}} P(\mathbf{x},\mathbf{y})$
- A posteriori probability $P(\mathbf{y}|\mathbf{x}) = P(\mathbf{x},\mathbf{y})/P(\mathbf{x})$

- Classify according to maximum a posteriori probability

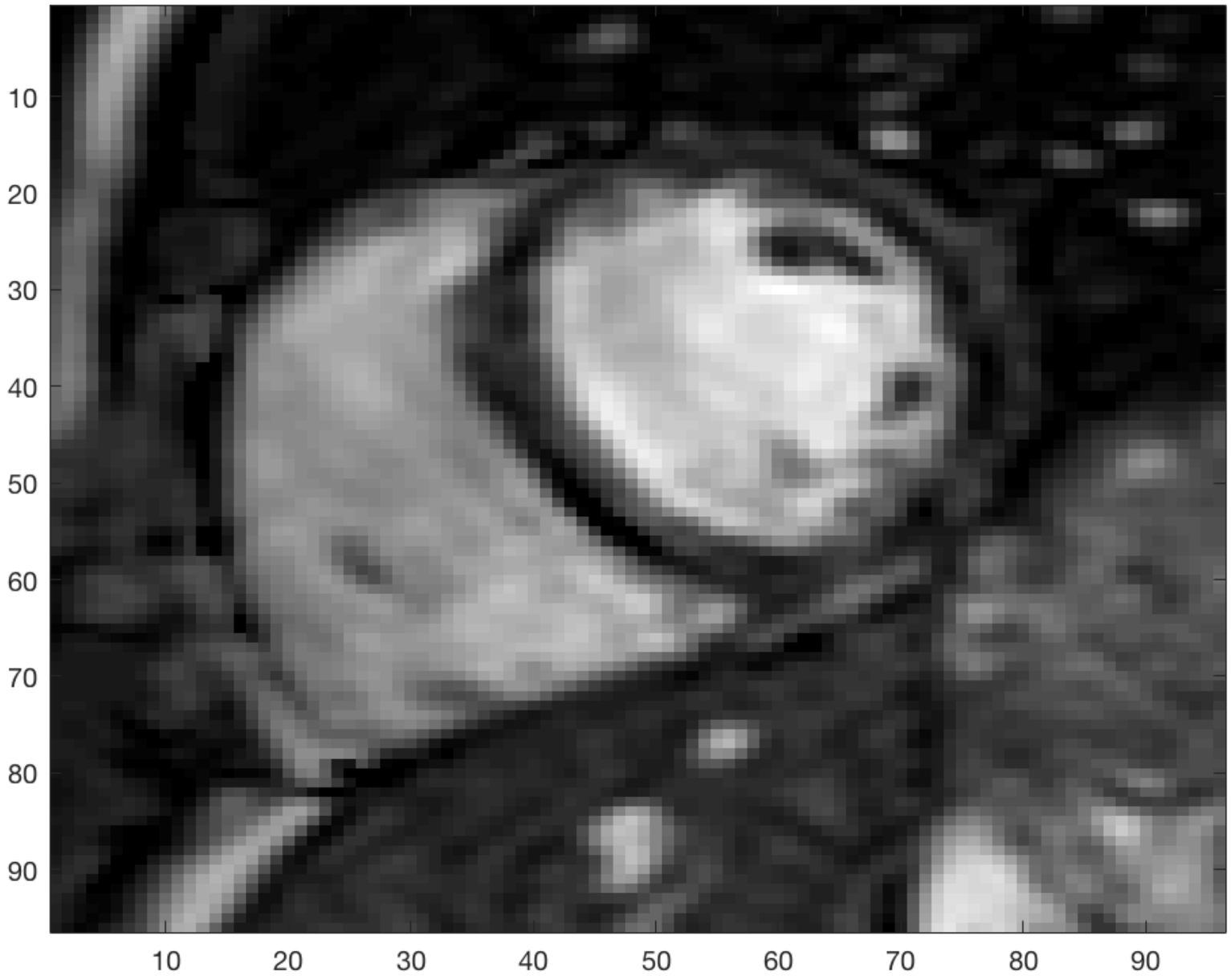
P(x,y)	x=Green	x=Yellow	
y=Albatross	0,72	0,18	0,9 p(y=Albatross)
y=Butterfly	0,05	0,05	0,1 p(y=Butterfly)
	0,77	0,23	1
	p(x=Green)	p(x=Yellow)	

P(x|y)	x=Green	x=Yellow	
y=Albatross	0,80	0,20	1,00
y=Butterfly	0,50	0,50	1,00
	1,30	0,70	2,00

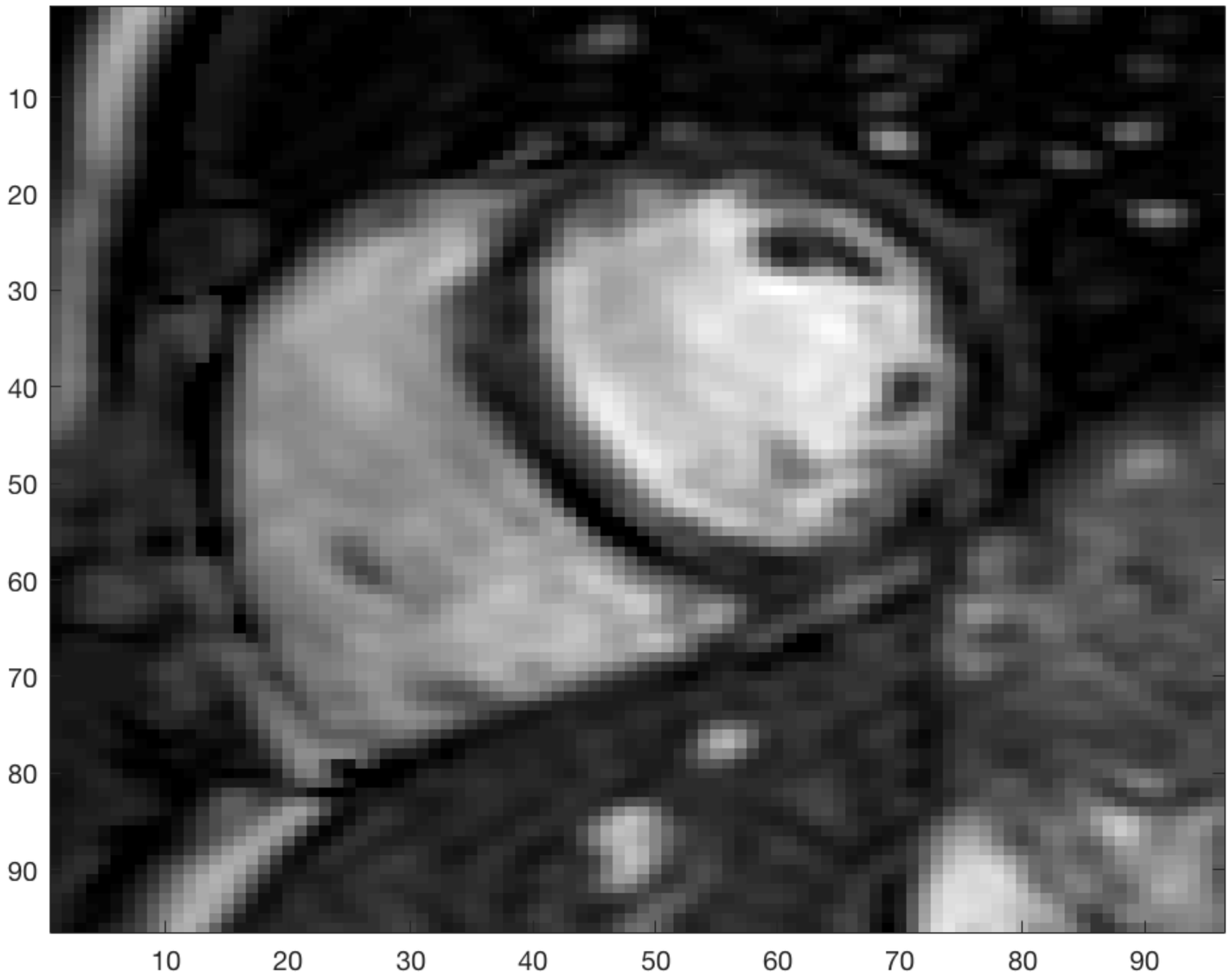
P(y|x)	x=Green	x=Yellow	
y=Albatross	0,94	0,78	1,72
y=Butterfly	0,06	0,22	0,28
	1	1	2

Another example – heart pixels

(We assume that we have segmented and annotated a number of heart images)



Use binning (quantization) to get fewer gray levels. Here 6 levels



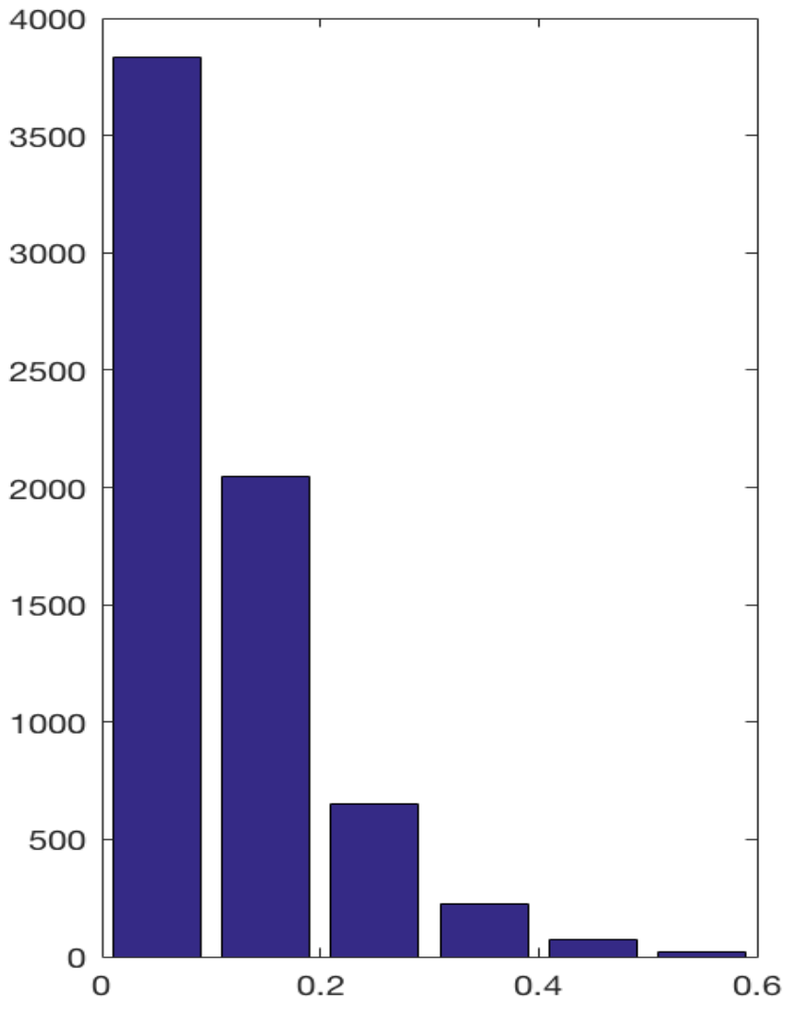
Discretize pixel brightness using 6 bins. Estimate probabilites from training data

P(x | background) P(x | heart)

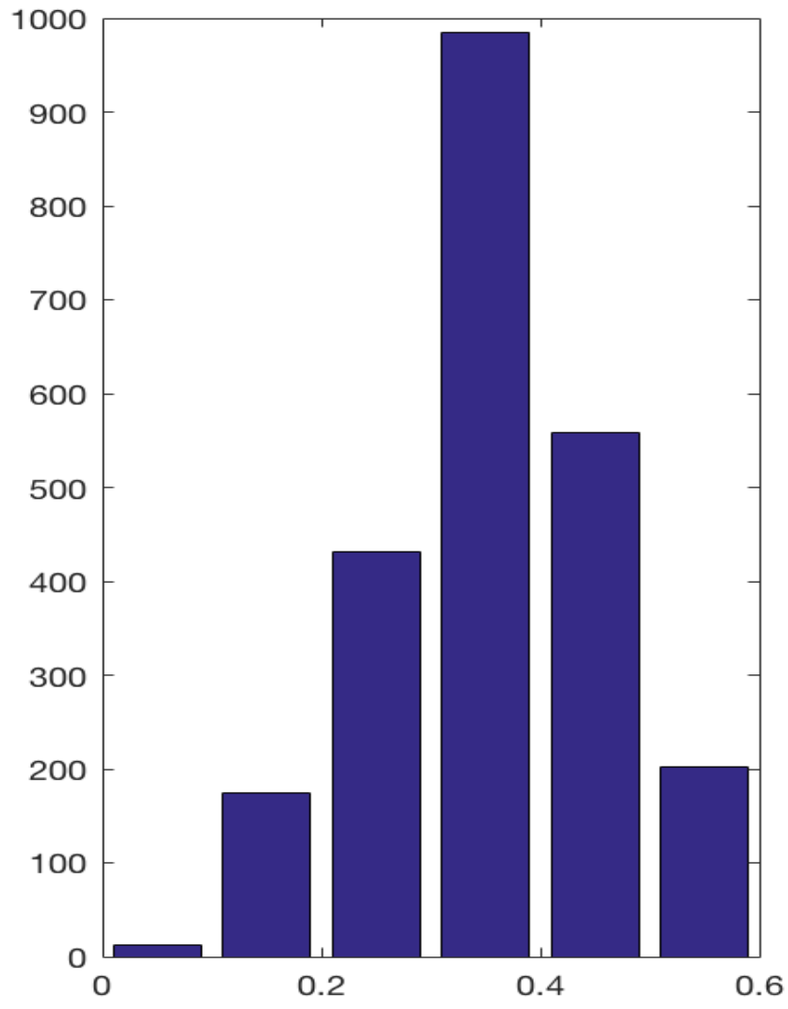
P(x|y)	x=1	x=2	x=3	x=4	x=5	x=6	
y=heart	0,0051	0,0737	0,1825	0,417	0,2363	0,0855	1,00
y=backgroun	0,56	0,30	0,095	0,03	0,01	0,00	1,00
	0,56	0,37	0,28	0,45	0,25	0,09	2,00

Discretize pixel brightness using 6 bins. Estimate probabilites from training data

P(x | background)



P(x | heart)



Estimate a posteriori probabilites.

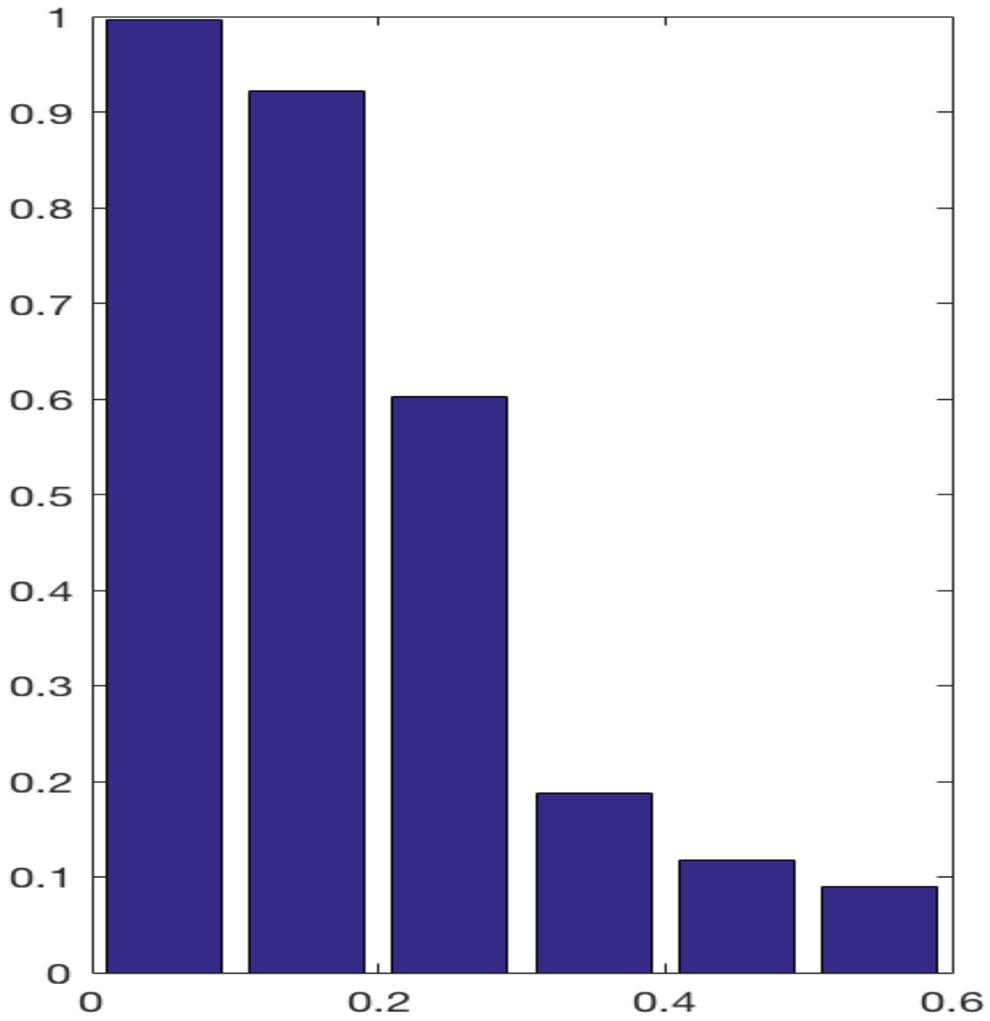
P(x|y)	x=1	x=2	x=3	x=4	x=5	x=6	
y=heart	0,0051	0,0737	0,1825	0,417	0,2363	0,0855	1,00
y=backgroun	0,56	0,30	0,095	0,03	0,01	0,00	1,00
	0,56	0,37	0,28	0,45	0,25	0,09	2,00

P(x,y)	x=1	x=2	x=3	x=4	x=5	x=6		
y=heart	0,001326	0,019162	0,04745	0,10842	0,061438	0,02223	0,26	p(y=heart)
y=backgroun	0,413956	0,221112	0,0703	0,024494	0,007992	0,002146	0,74	p(y=background)
	0,415282	0,240274	0,11775	0,132914	0,06943	0,024376	1	
	p(x=1)	p(x=2)	p(x=3)	p(x=4)	p(x=5)	p(x=6)		

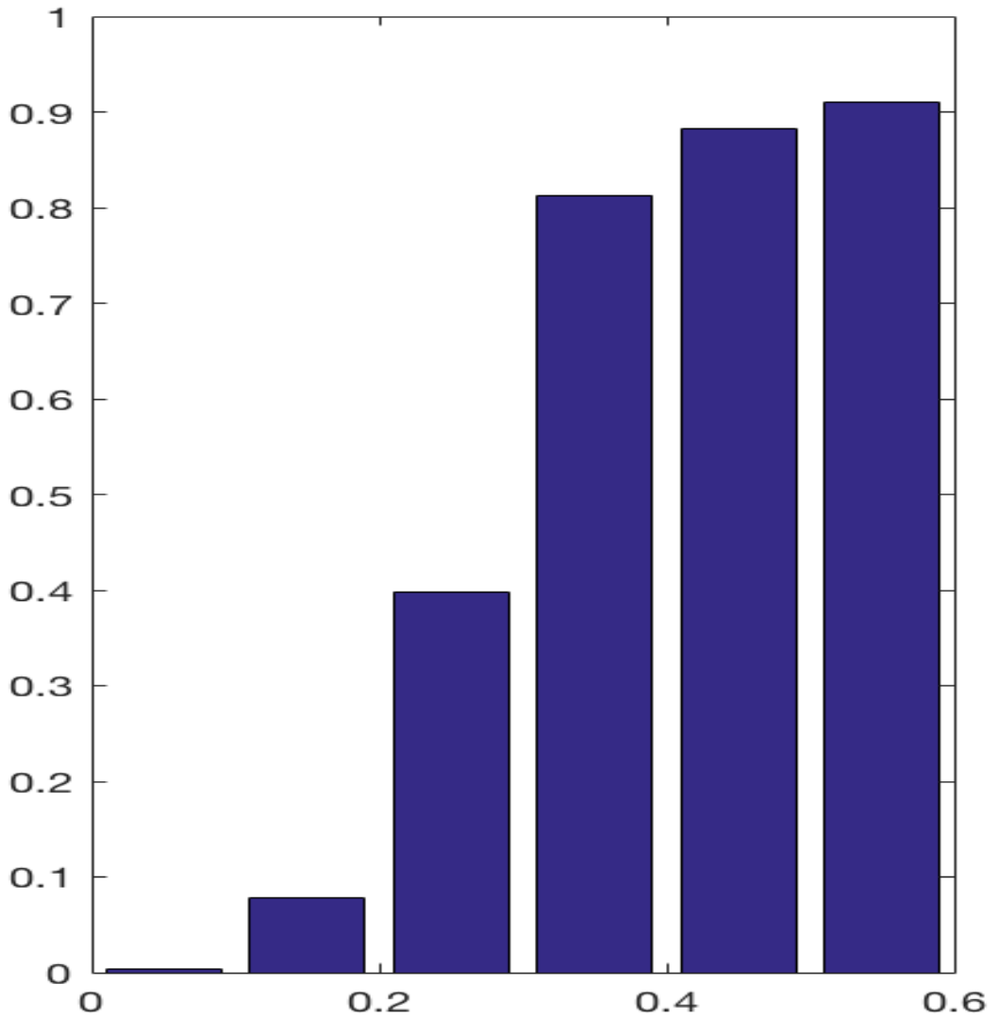
P(y|x)	x=1	x=2	x=3	x=4	x=5	x=6	
y=heart	0,00	0,08	0,40	0,82	0,88	0,91	0,92
y=backgroun	1,00	0,92	0,60	0,18	0,12	0,09	1,08
	1	1	1	1	1	1	2

Estimate a posteriori probabilites. Use these as gray level transforms

P(background | x)

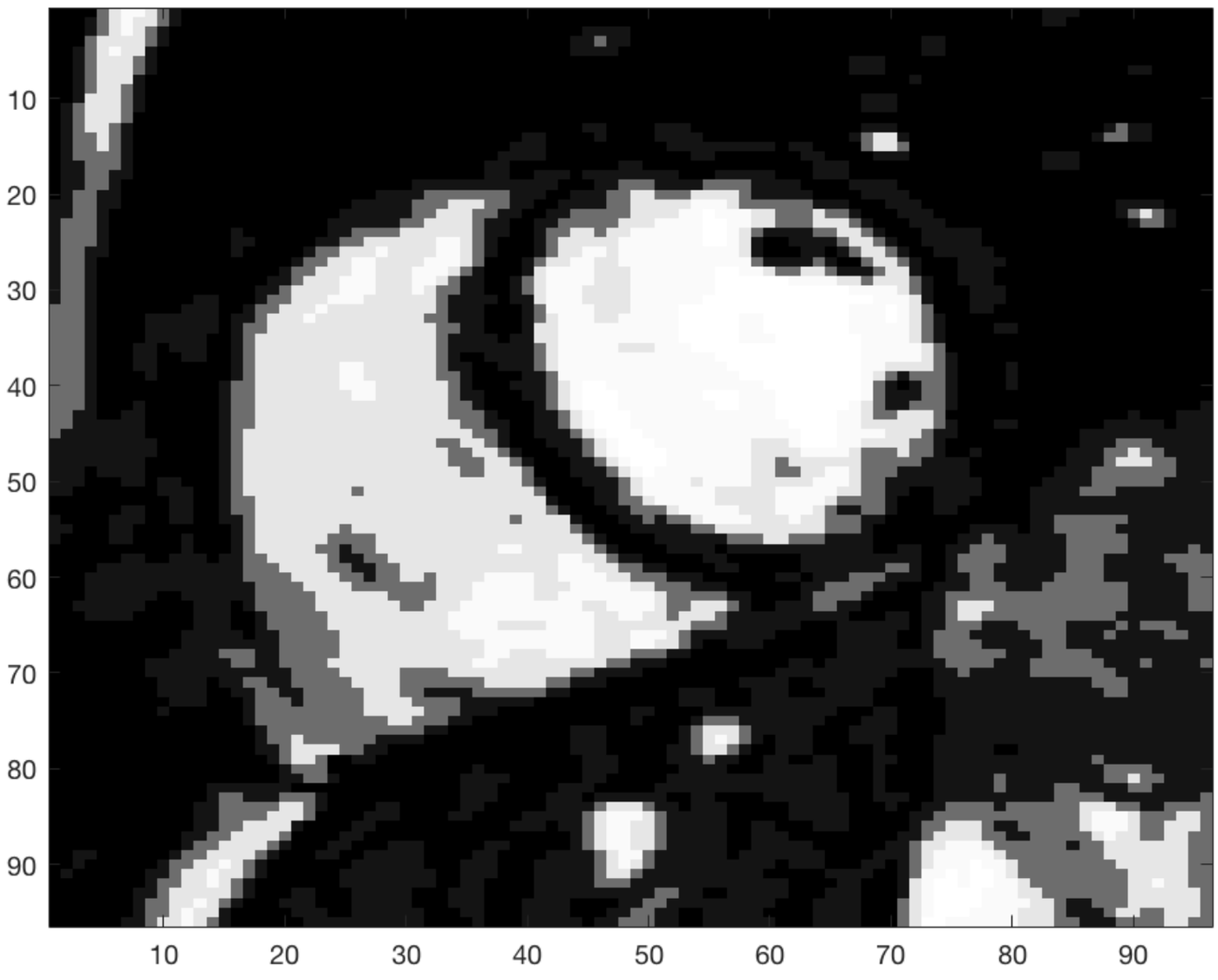


P(heart | x)



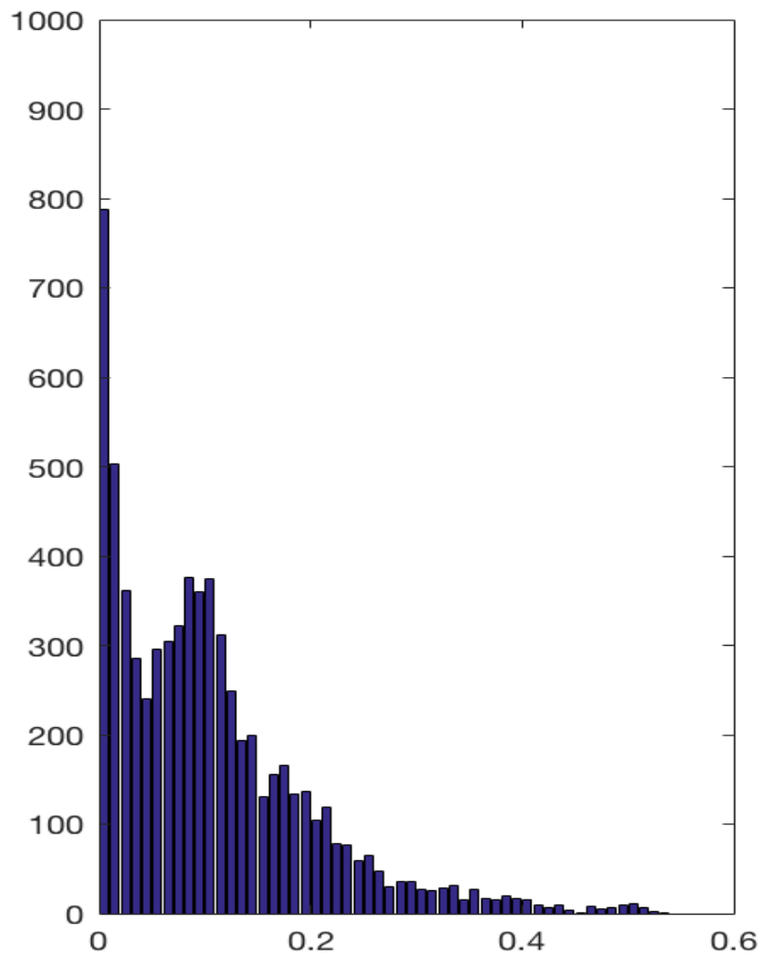
Discretize pixel brightness using 6 bins. Estimate probabilites

P(heart | x)

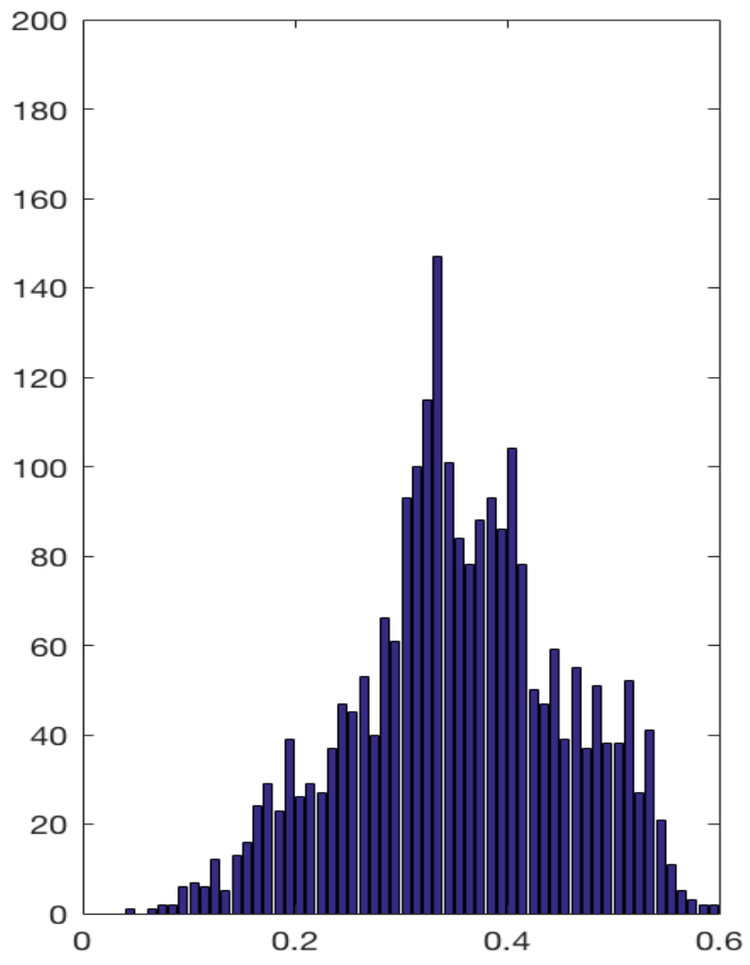


Discretize pixel brightness using 60 bins. Estimate probabilites

P(x | background)

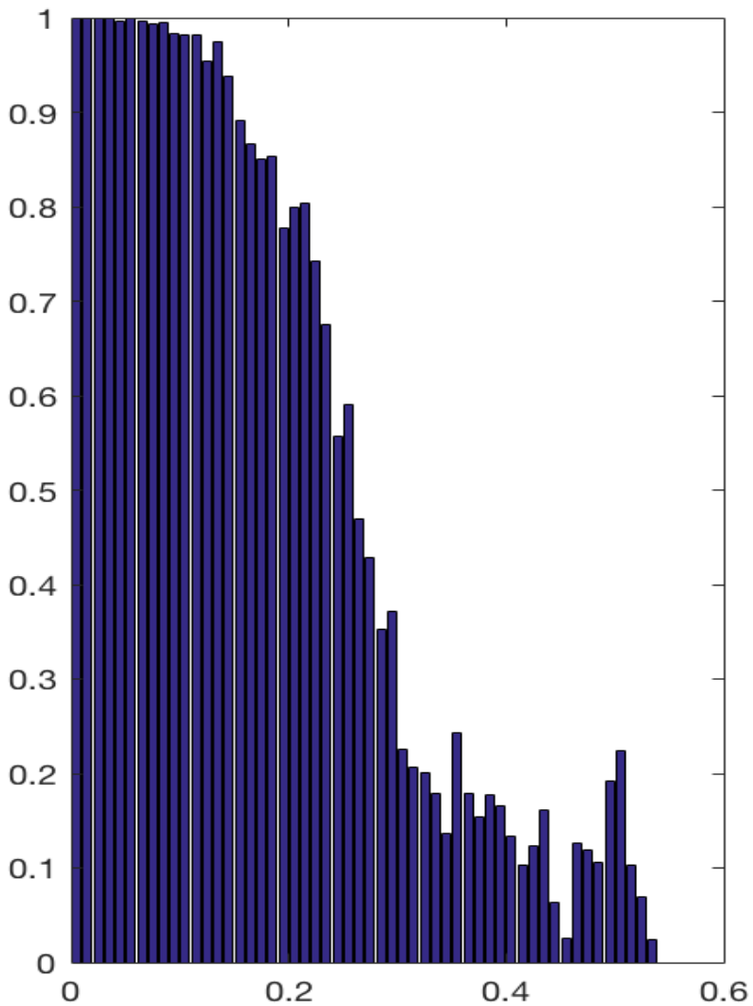


P(x | heart)

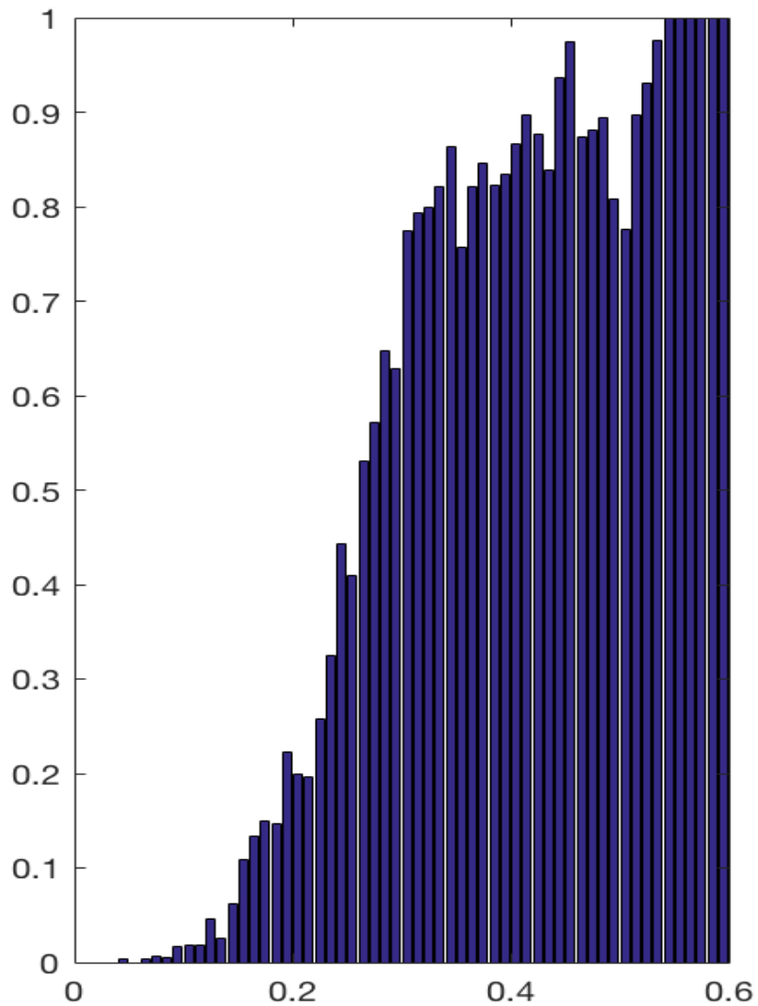


Estimate a posteriori probabilites.
Use these as gray level transforms

P(background | x)

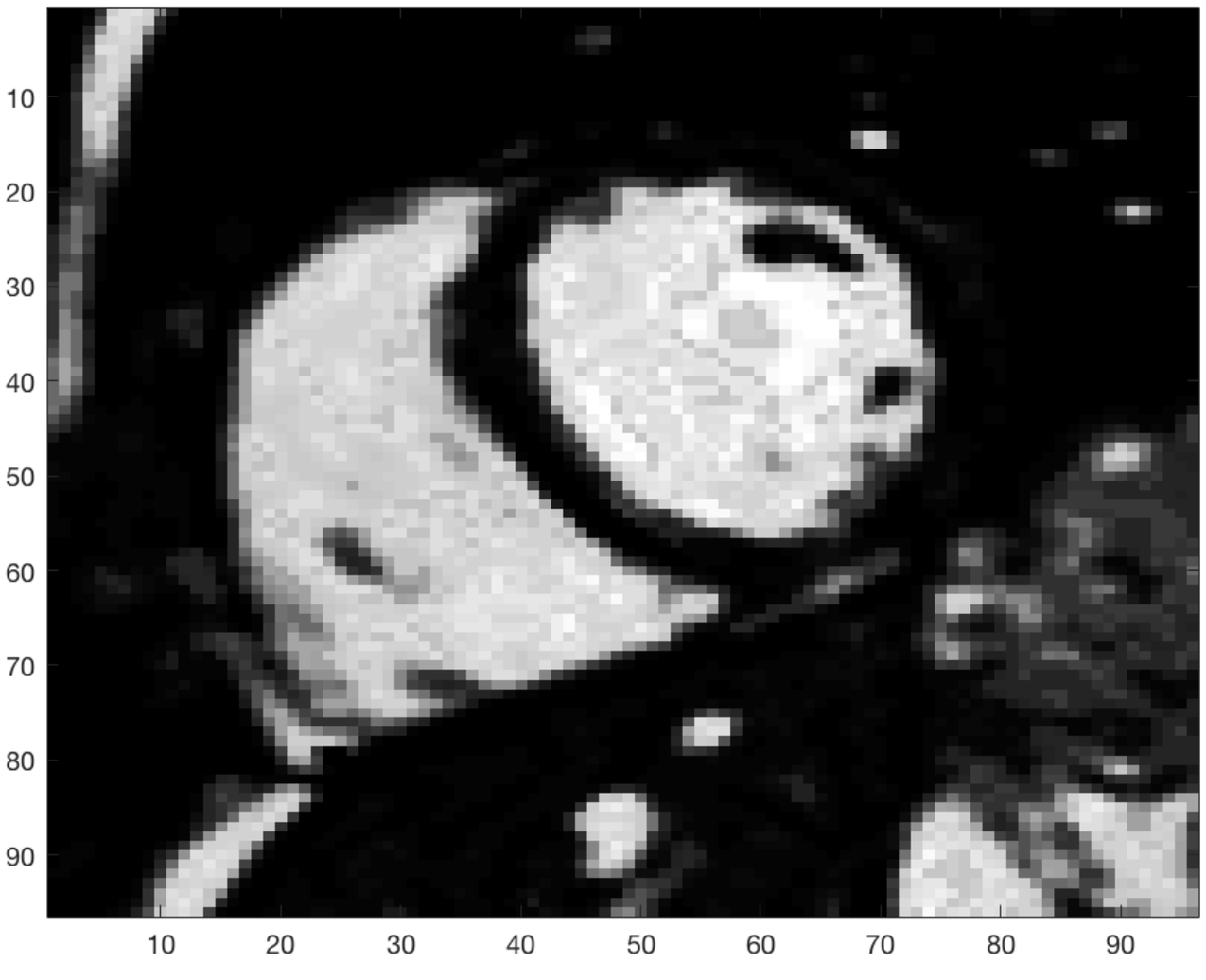


P(heart | x)



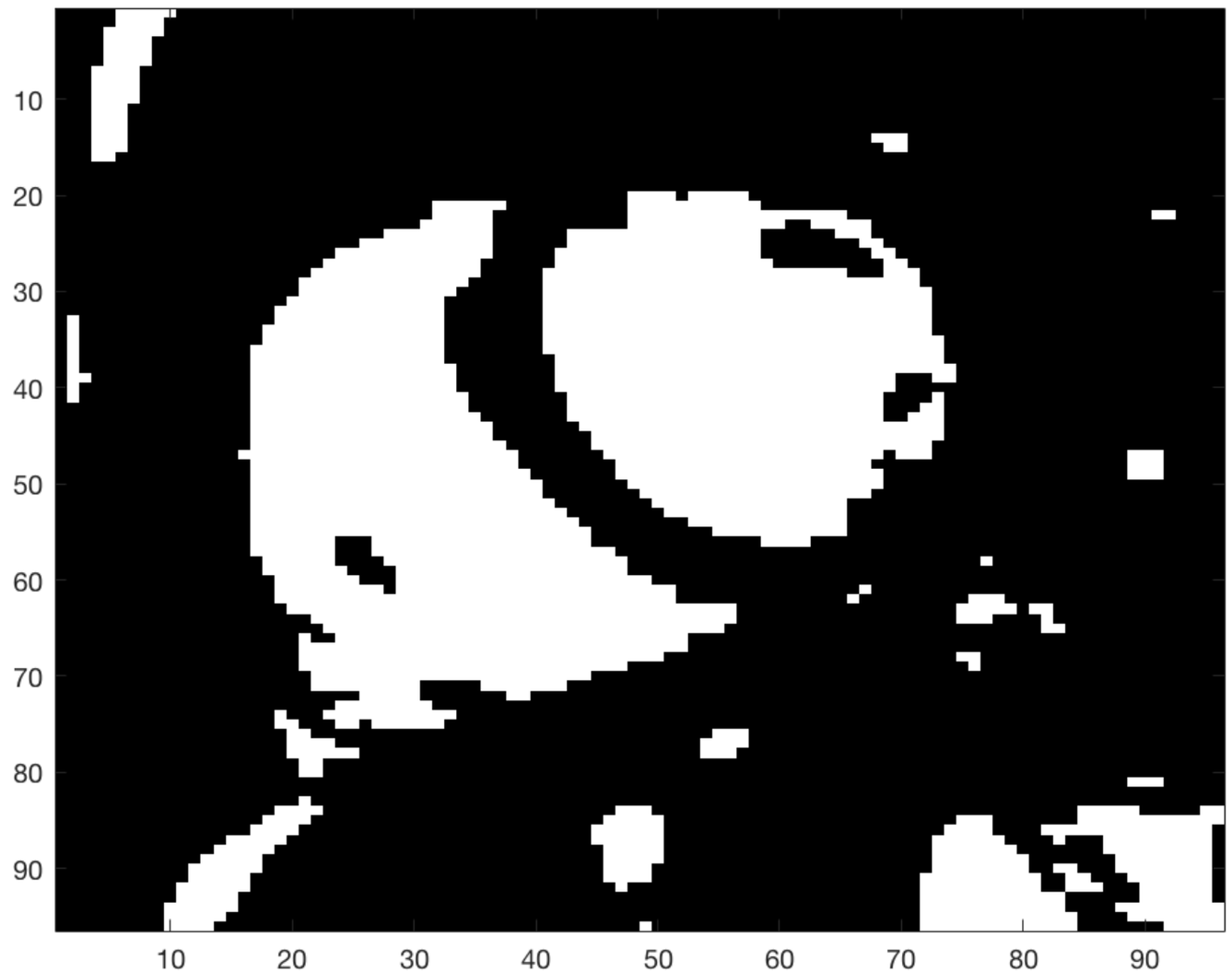
Discretize pixel brightness using 60 bins. Estimate probabilites

P(heart | x)



Discretize pixel brightness using 60 bins. Estimate probabilites

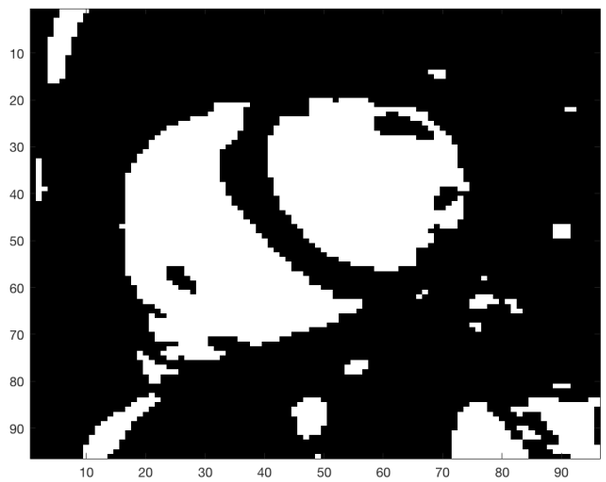
$$P(\text{heart} \mid x) > 0.5$$

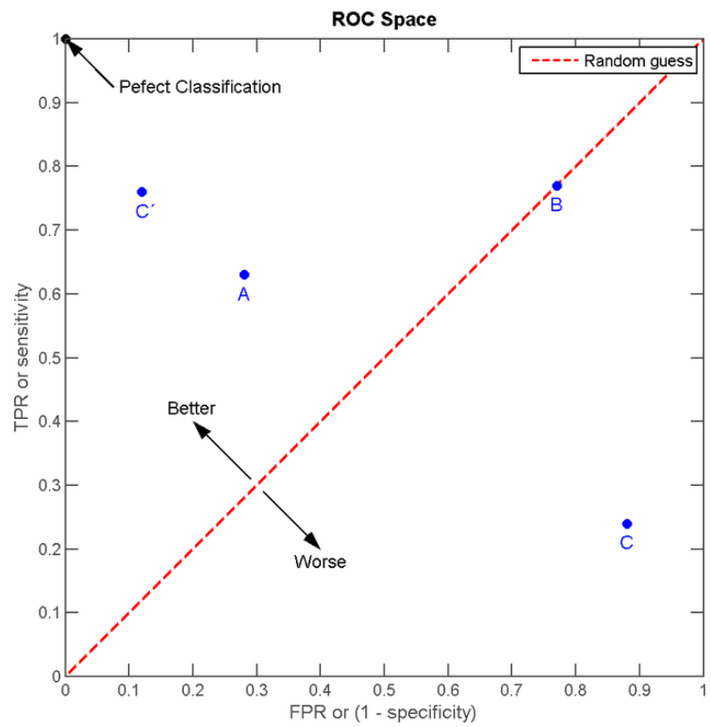


False Positives, False Negatives ROC - Curve

- For two class problems - Negatives and Positives
- Negatives that are classified as negatives – True Negatives (TN)
- Positives that are classified as positives – True Positives (TP)
- Negatives that are classified as positives – False Positives (FP)
- Positives that are classified as negatives – False Negatives (FN)

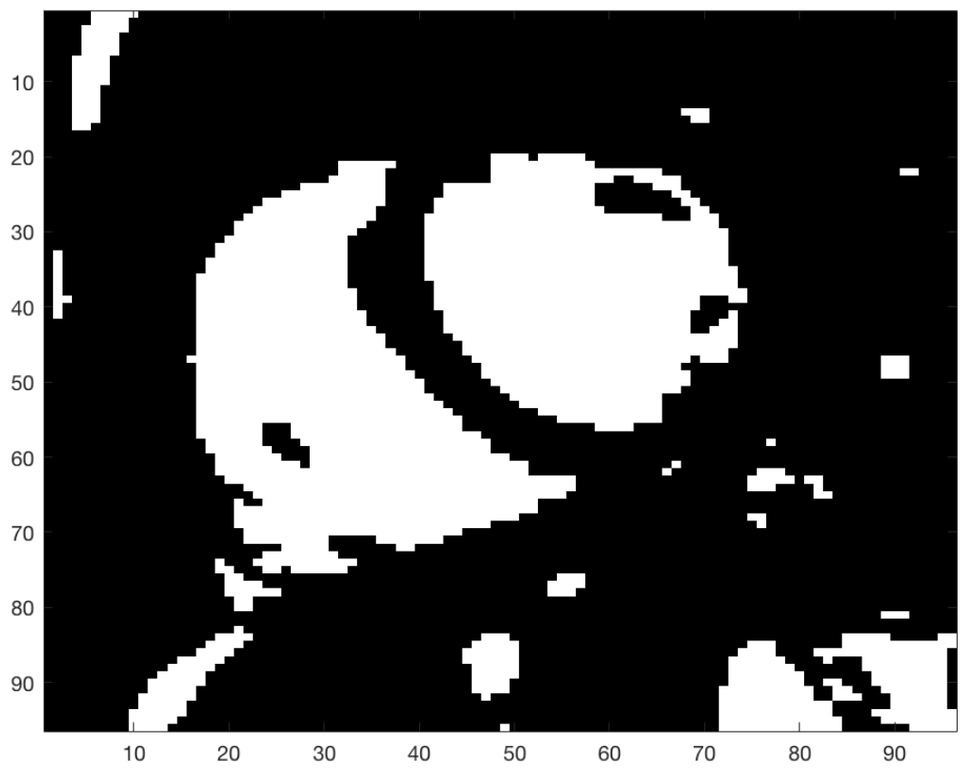
- False Positive Rate FPR = FP/(FP+TN) -> x-axis
- True Positive Rate TPR = TP/(TP+FN) -> y-axis

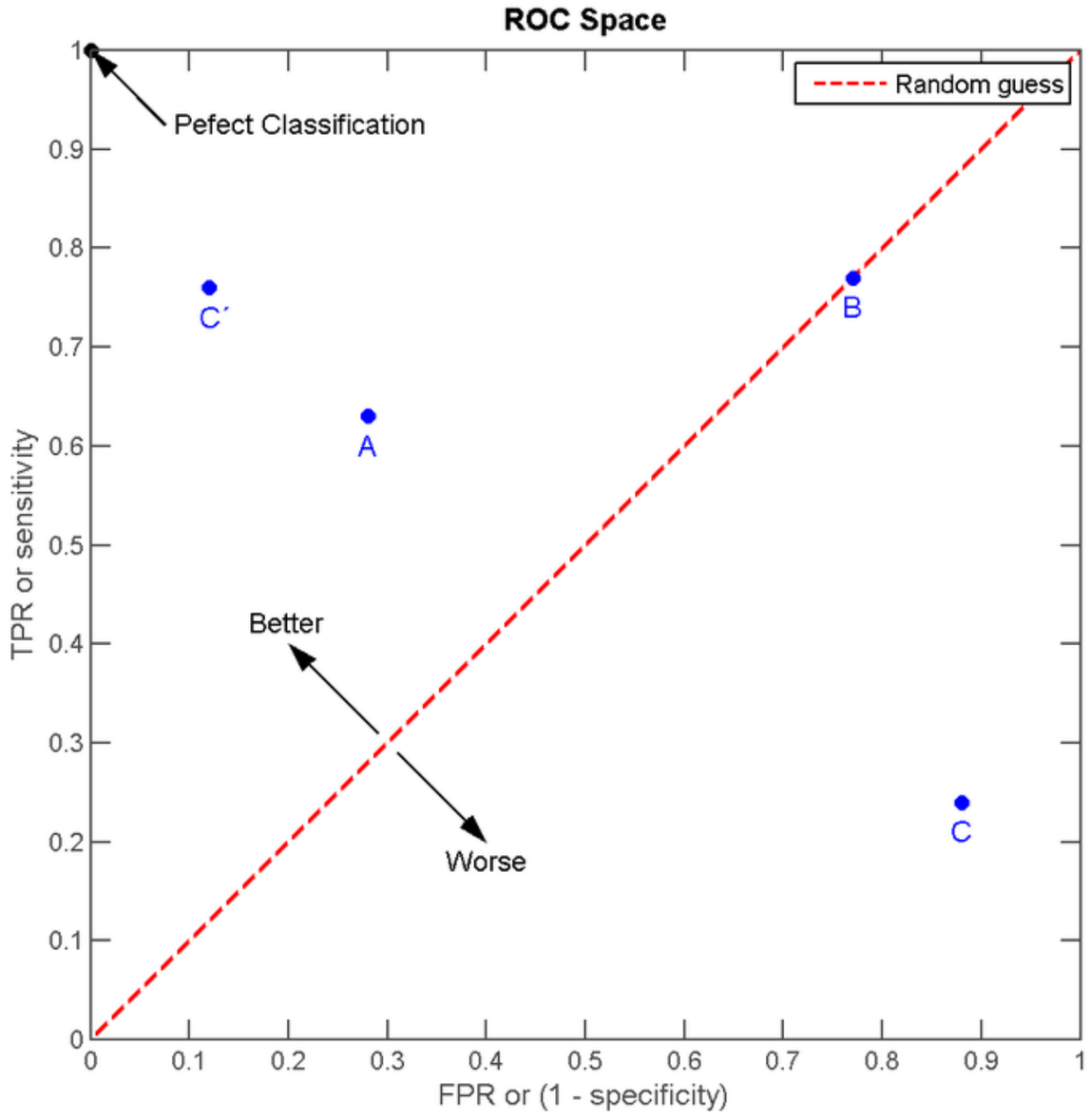




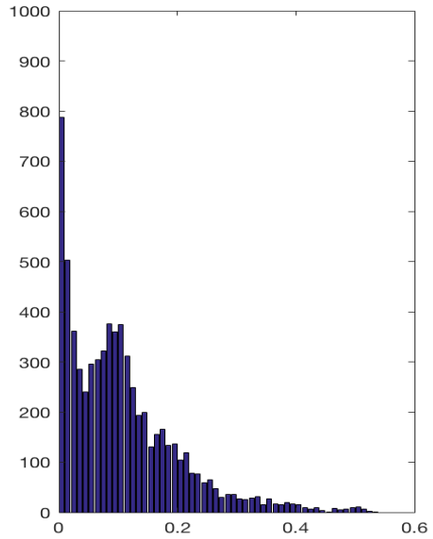
False Positives, False Negatives ROC - Curve

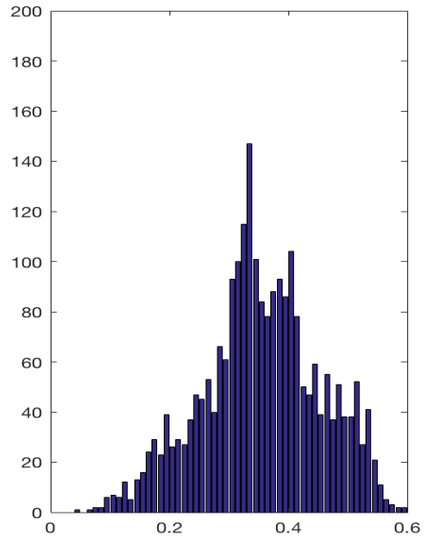
- FPR = FP/(FP+TN) -> x-axis
- TPR = TP/(TP+FN) -> y-axis





Bayes Theorem





- Bayes Theorem

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

- Interpret P as probabilites, e.g. If X and Y are discrete
- Interpret P as probability density functions, e.g. If X and/or Y are continuous stochastic variable,

$$P(Y = y|X = x) = \frac{f_X(x|Y = y)P(Y = y)}{f_X(x)}$$

$$f_Y(y|X = x) = \frac{f_X(x|Y = y)f_Y(y)}{f_X(x)}$$

From 1 channel to 3
Colour images

Binning in higher dimensions
The curse of dimensionality

- 10 bins in one dimension -> 10 bins
- 10 bins in two dimensions -> 100 bins
- 10 bins in three dimensions -> 1000 bins
- 10 bins in 30 dimensions -> 1 000 000 000 000 000 000 000 000 000 000 bins
- 10 bins in 128 dimensions -> 10^(128) bins

- ”*Many algorithms that work fine in low dimensions become intractable when the input is high-dimensional*.”, Bellman, 1961.

Richard Bellman

LUND UNIVERSITY

Clustering – adaptive binning

- Colour images. Pixels are RGB with 8 bits each. 2^24 = 16777216 types of pixels.
- Too many.
- Try to bin in a more clever way?
- Clustering

Clustering: group together similar points and represent them with a single token

Key Challenges:

1) What makes two points/images/patches similar?

2) How do we compute an overall grouping from pairwise similarities?

Why do we cluster?

- **Summarizing data**
  - Look at large amounts of data
  - Patch-based compression or denoising
  - Represent a large continuous vector with the cluster number
- **Counting**
  - Histograms of texture, color, SIFT vectors
- **Segmentation**
  - Separate the image into different regions
- **Prediction**
  - Images in the same cluster may have the same labels

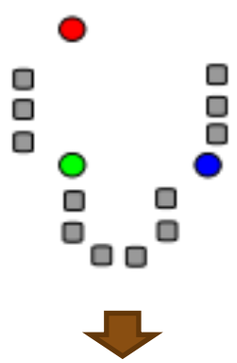
Slide: Derek Hoiem

How do we cluster?

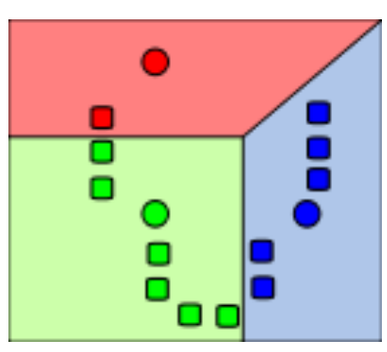
- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

K-means algorithm

1. Randomly select K centers



2. Assign each point to nearest center



3. Compute new center (mean) for each cluster

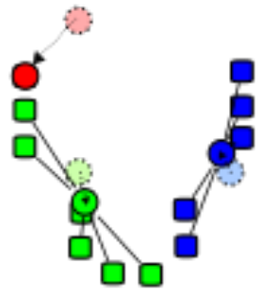
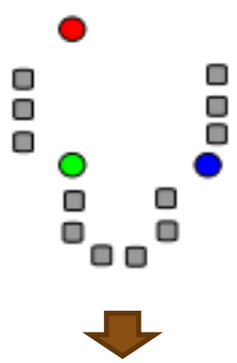


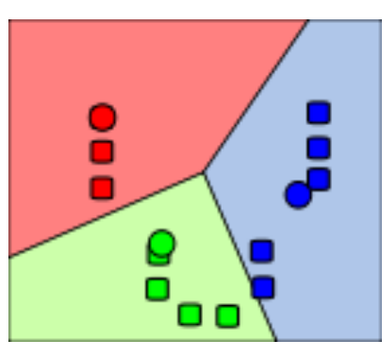
Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means algorithm

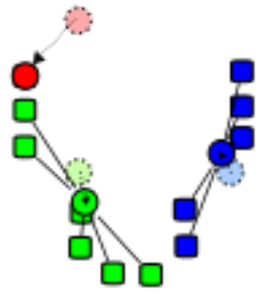
1. Randomly select K centers



2. Assign each point to nearest center



3. Compute new center (mean) for each cluster



Back to 2

Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means

1. Initialize cluster centers: $\mathbf{c}^0$; t=0

2. Assign each point to the closest center

$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

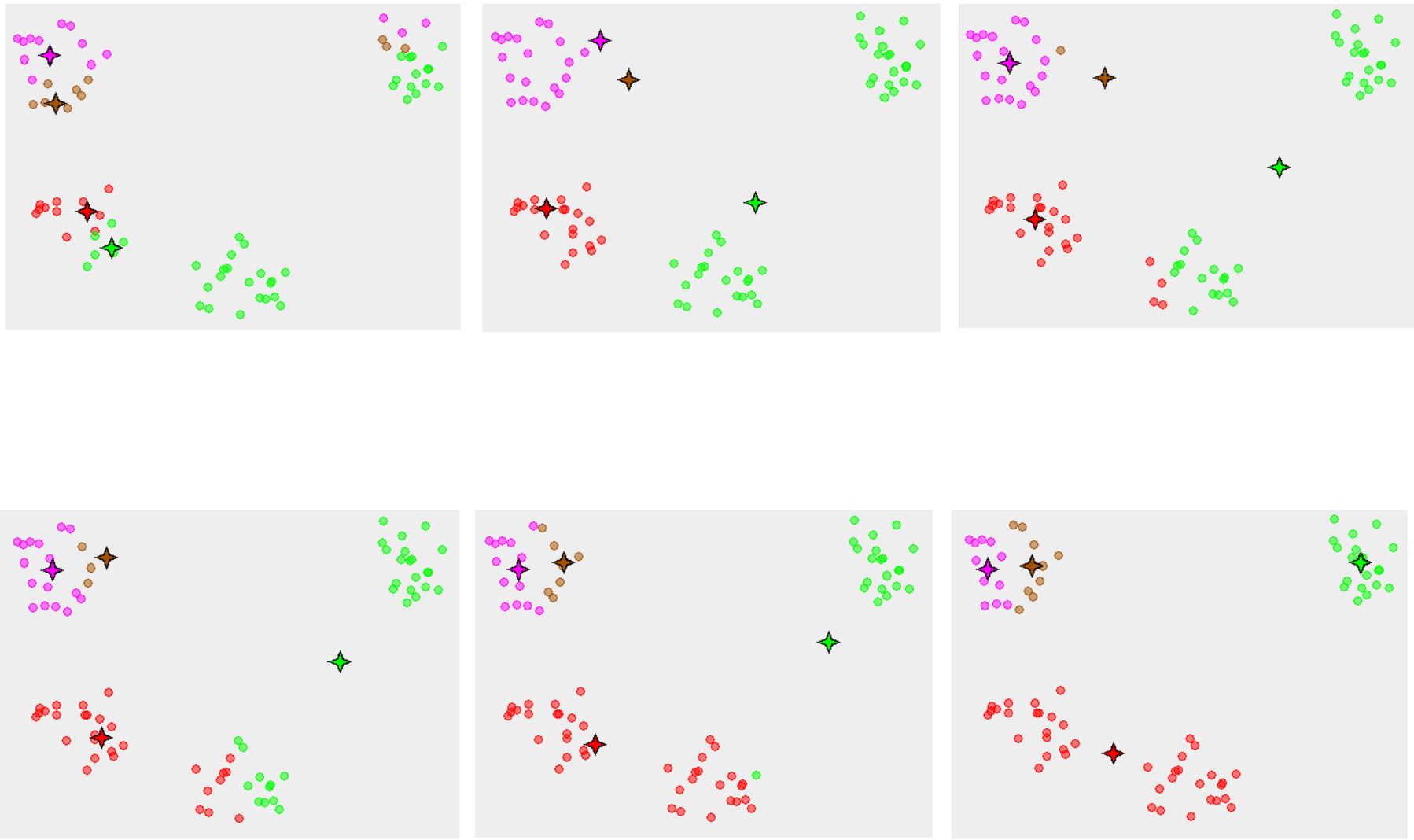
3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4. Repeat 2-3 until no points are re-assigned $(t=t+1)$

Slide: Derek Hoiem

K-means converges to a local minimum



K-means: design choices

- Initialization
  - Randomly select K points as initial cluster center
  - Or greedily choose K points to minimize residual

- Distance measures
  - Traditionally Euclidean, could be others

- Optimization
  - Will converge to a *local minimum*
  - May want to perform multiple restarts

How to evaluate clusters?

- Generative
  - How well are points reconstructed from the clusters?
- Discriminative
  - How well do the clusters correspond to labels?
  - Note: unsupervised clustering does not aim to be discriminative

Slide: Derek Hoiem

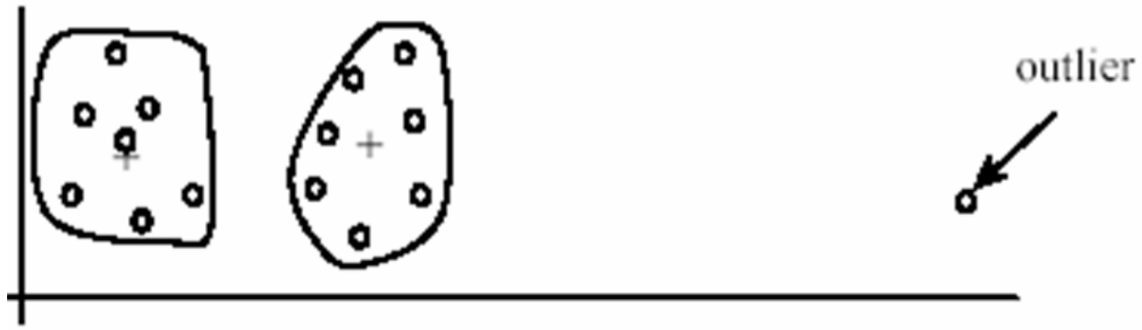
How to choose the number of clusters?

- Validation set
  - Try different numbers of clusters and look at performance
    - When building dictionaries (discussed later), more clusters typically work better

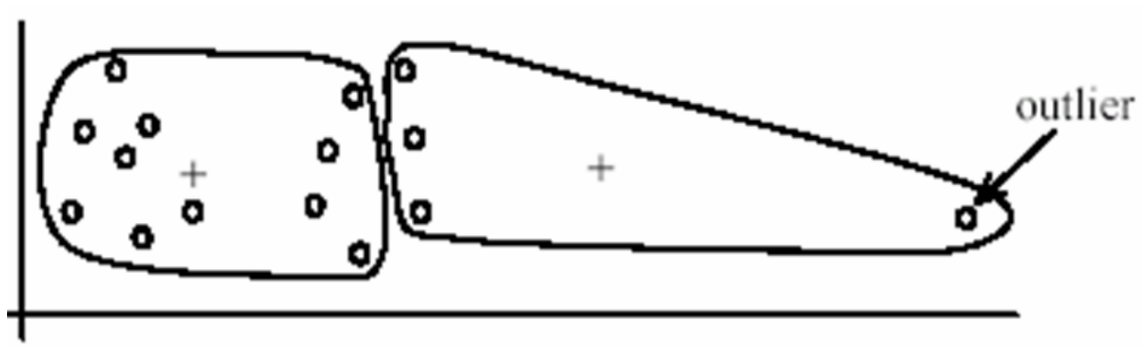
Slide: Derek Hoiem

K-Means pros and cons

- Pros
  - Finds cluster centers that minimize conditional variance (good representation of data)
  - Simple and fast*
  - Easy to implement
- Cons
  - Need to choose K
  - Sensitive to outliers
  - Prone to local minima
  - All clusters have the same parameters (e.g., distance measure is non-adaptive)
  - *Can be slow: each iteration is O(KNd) for N d-dimensional points
- Usage
  - Rarely used for pixel segmentation

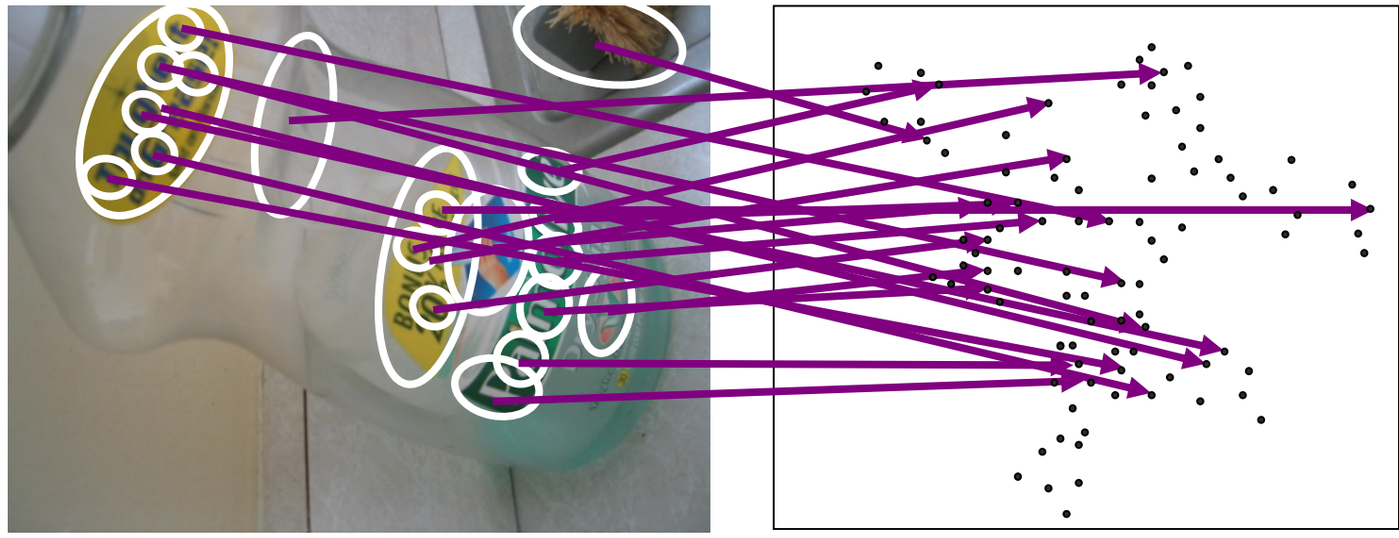


(B): Ideal clusters

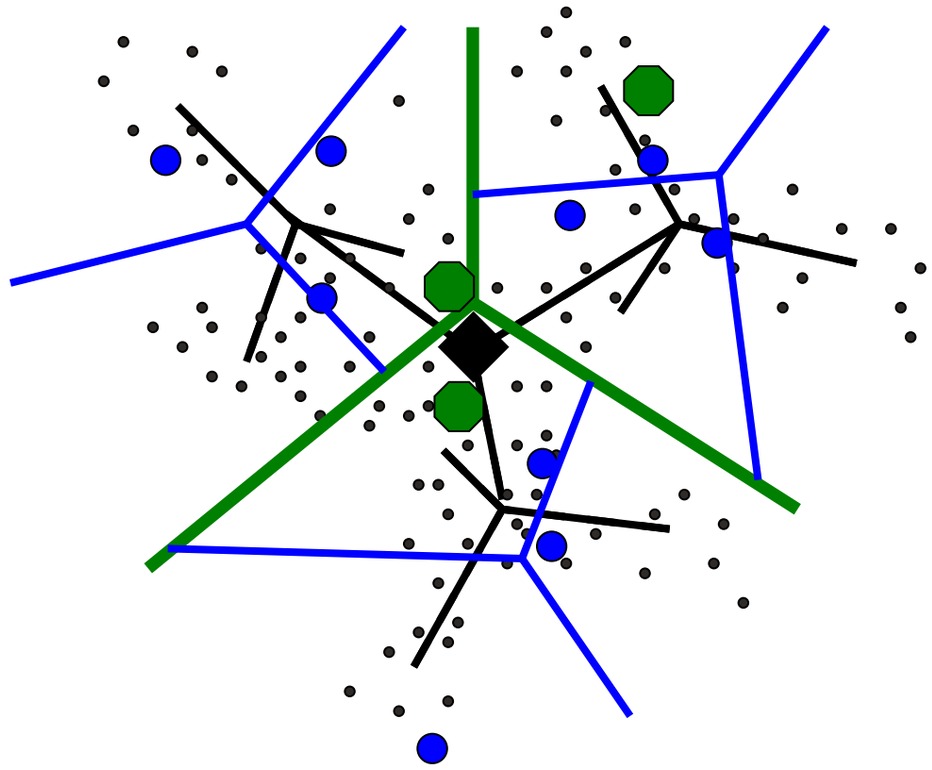


Building Visual Dictionaries

1. Sample patches from a database
   - E.g., 128 dimensional SIFT vectors

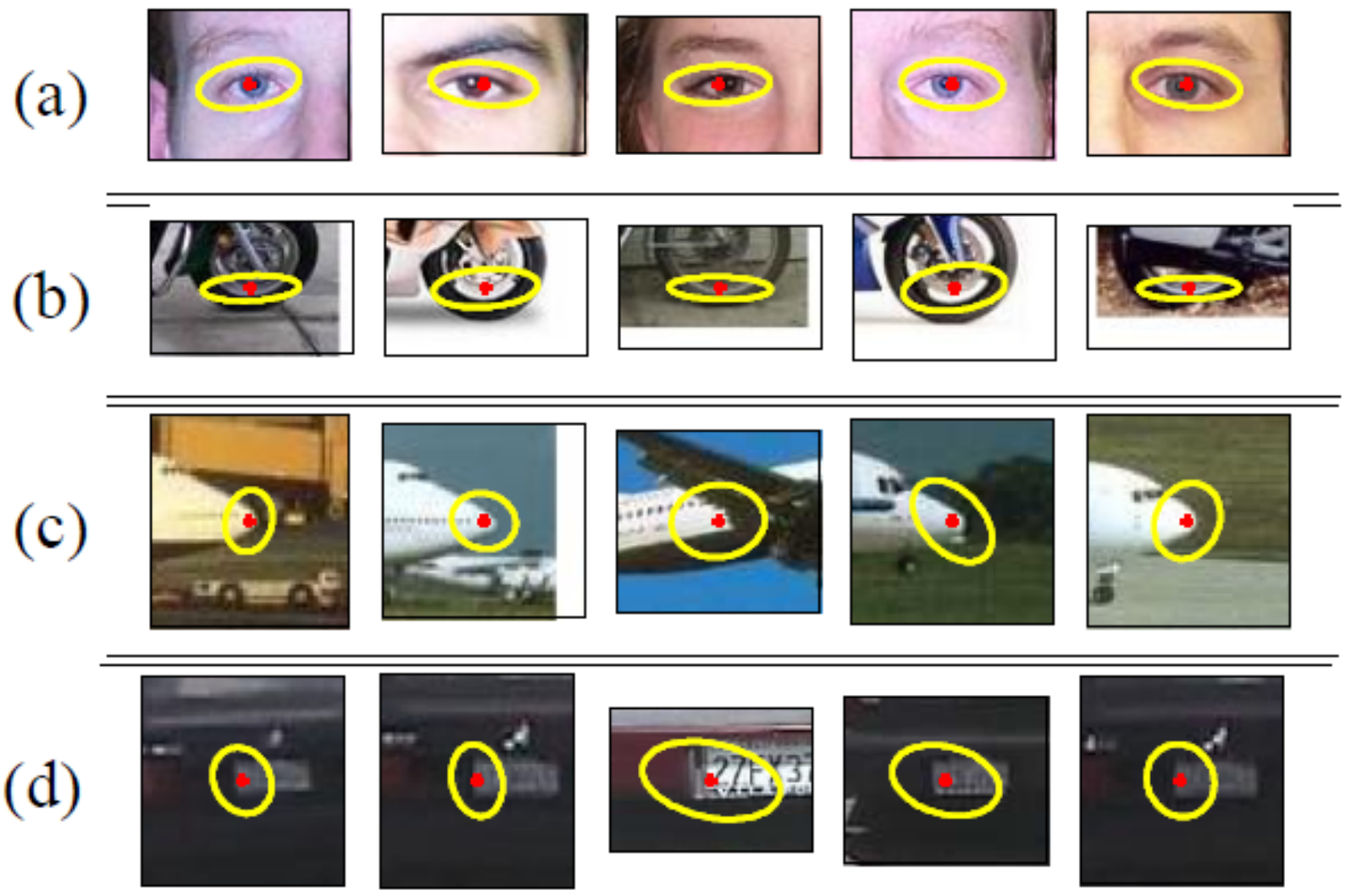


2. Cluster the patches
   - Cluster centers are the dictionary



3. Assign a codeword (number) to each new patch, according to the nearest cluster

Examples of learned codewords



Most likely codewords for 4 learned “topics”
EM with multinomial (problem 3) to get topics

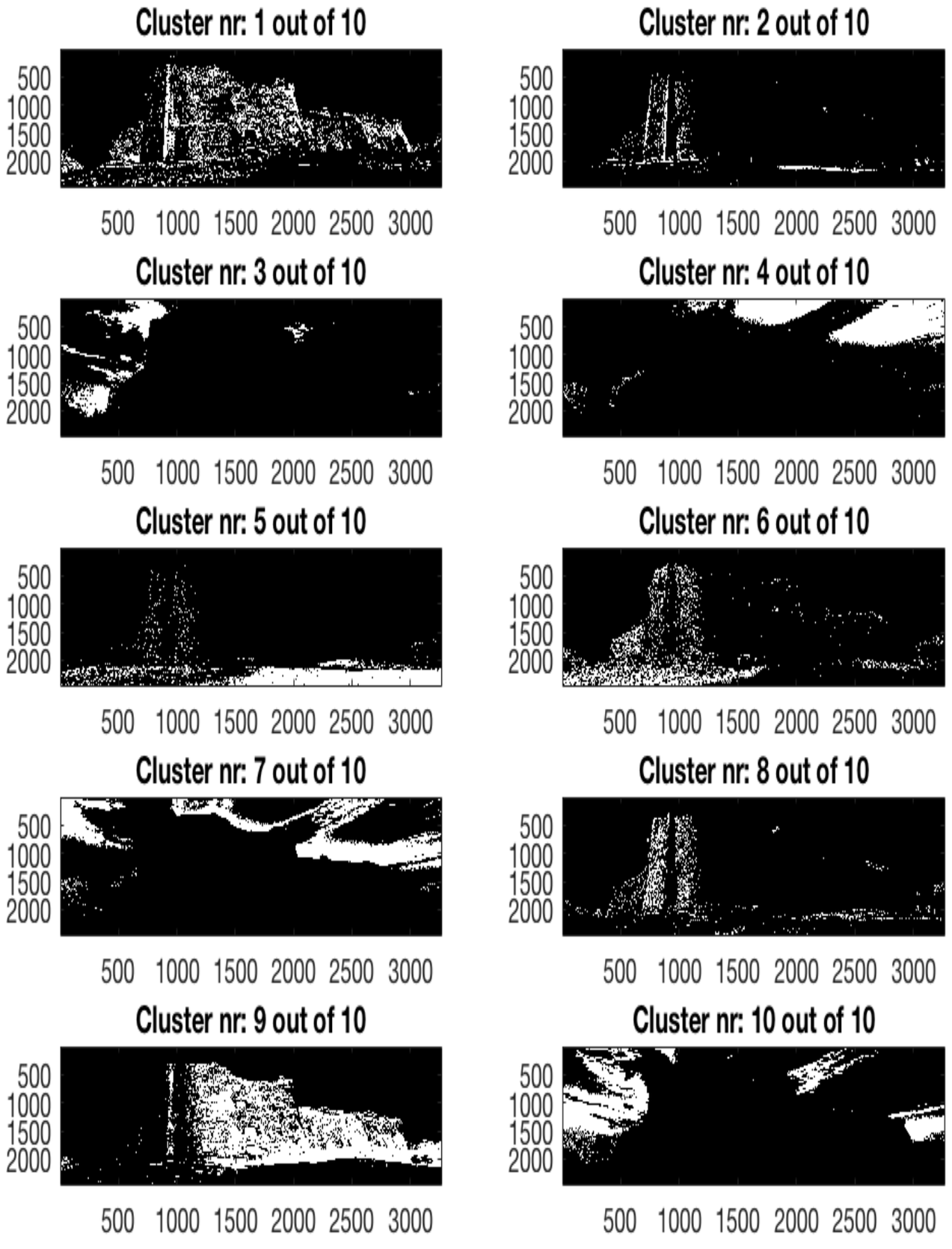
http://www.robots.ox.ac.uk/~vgg/publications/papers/sivic05b.pdf Sivic et al. ICCV 2005

Example:
Colour pixel classification

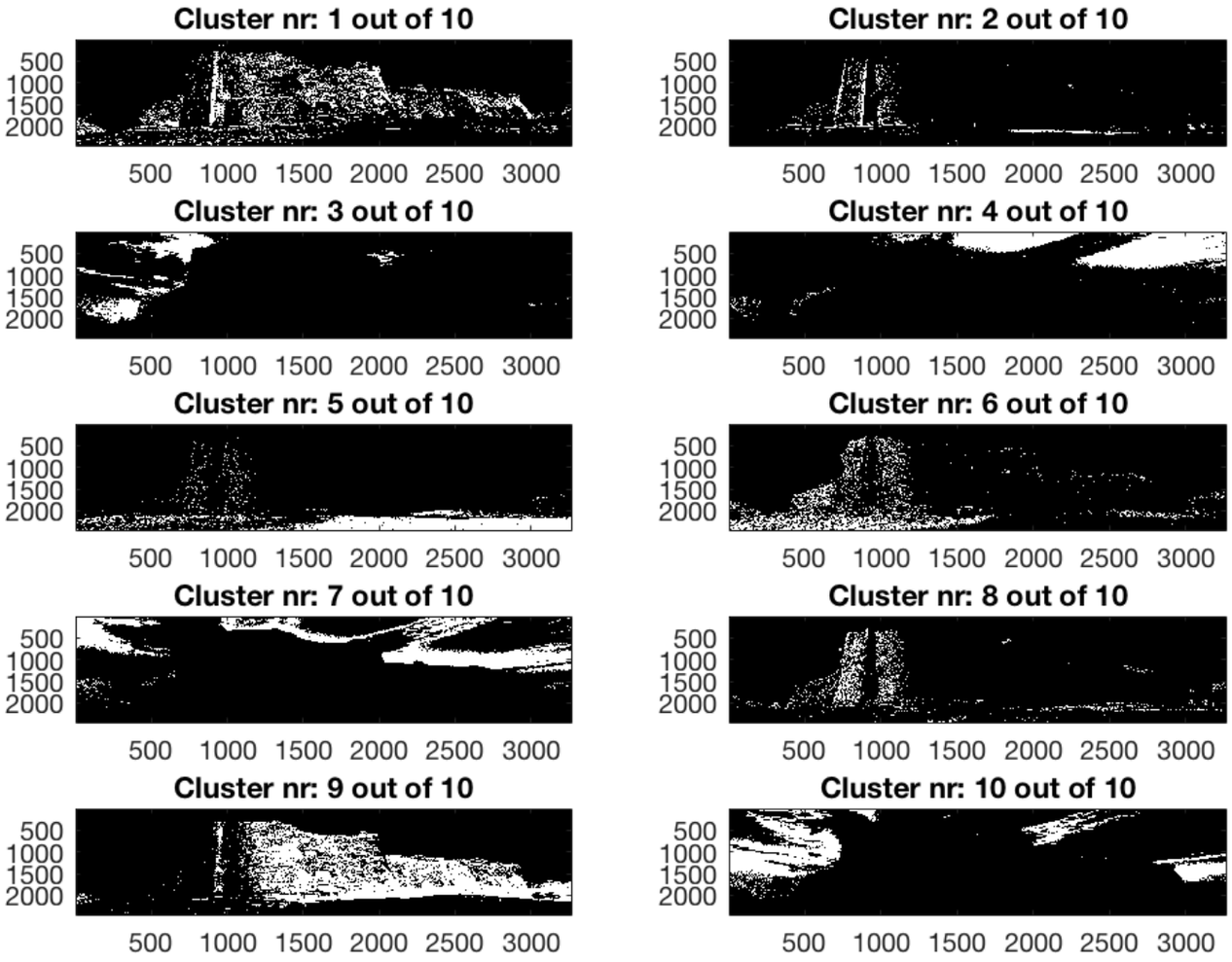
- Use clustering to assign codewords (bin nr) x 1-10 for each pixel
- Estimate measurement probabilites p(x|y) for each class y (1-grass, 2-castle, 3-sky) and each bin x.
- Use Bayes theorem to calculate p(y|x) for each pixel,
- Classify according to maximum a posteriori probability

LUND UNIVERSITY

Example: Colour pixel classification

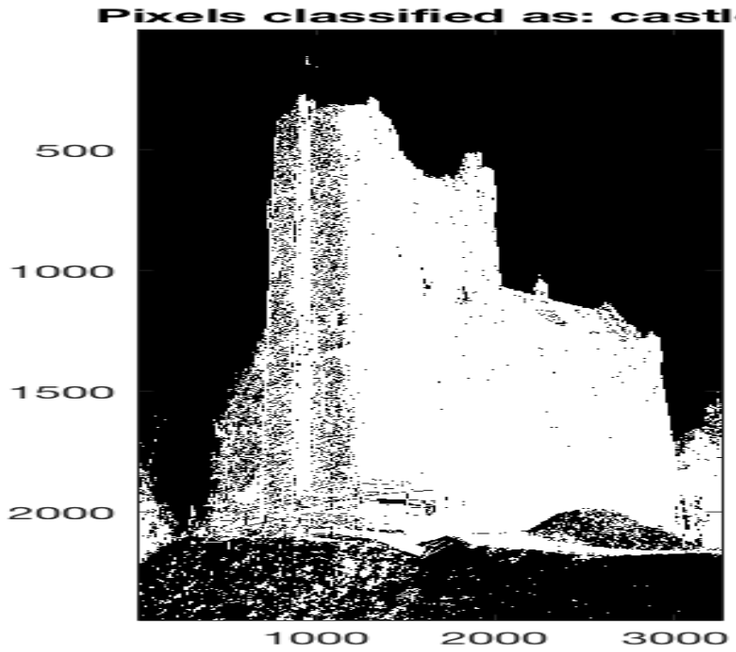


Example: Colour pixel classification

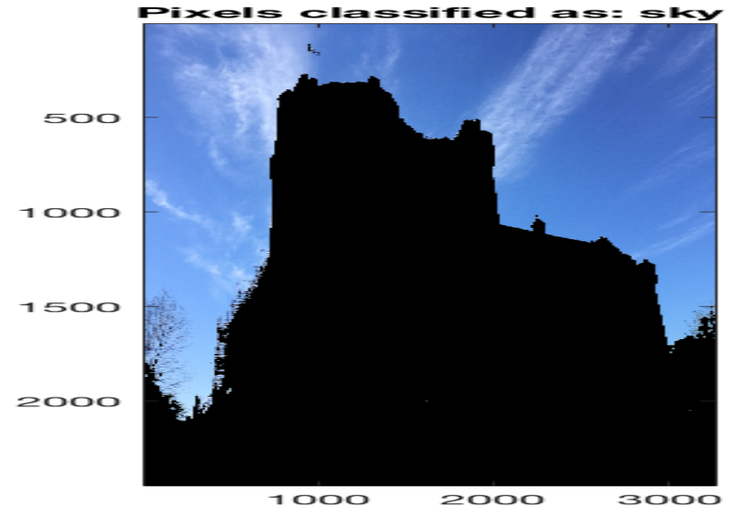


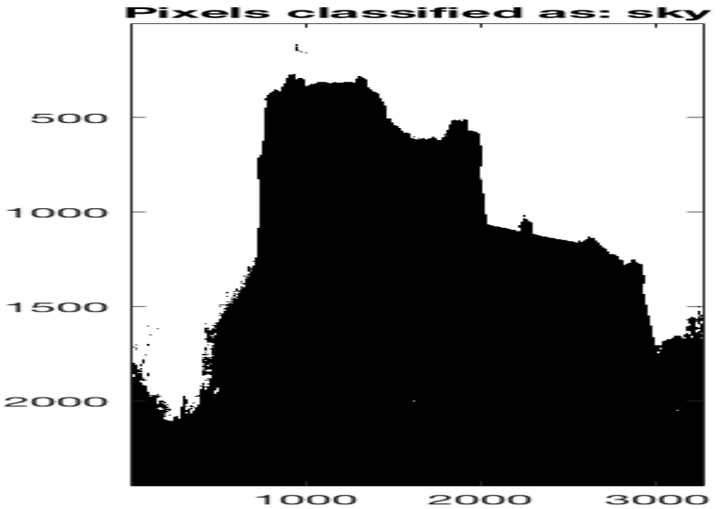

500
1000
1500
2000
1000
2000
3000


Pixels classified as: castle
500
1000
1500
2000
1000
2000
3000

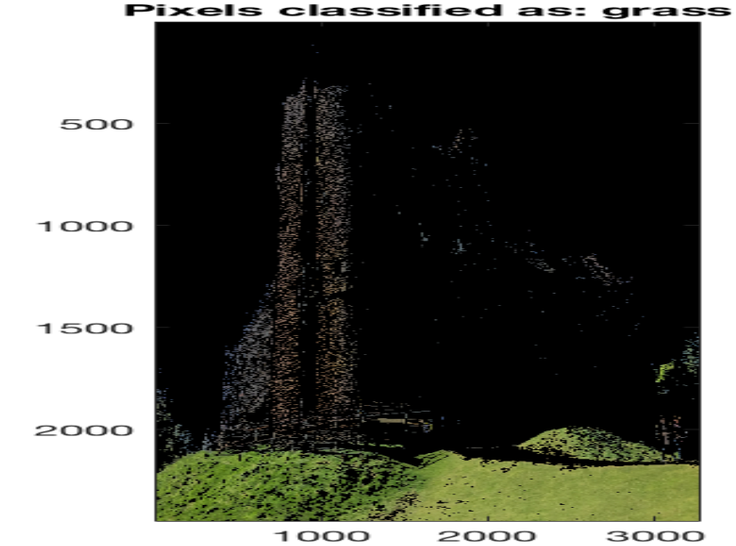

Pixels classified as: castle
500
1000
1500
2000
1000
2000
3000


500
1000
1500
2000
1000
2000
3000


Pixels classified as: sky
500
1000
1500
2000
1000
2000
3000


Pixels classified as: sky
500
1000
1500
2000
1000
2000
3000


500
1000
1500
2000
1000
2000
3000

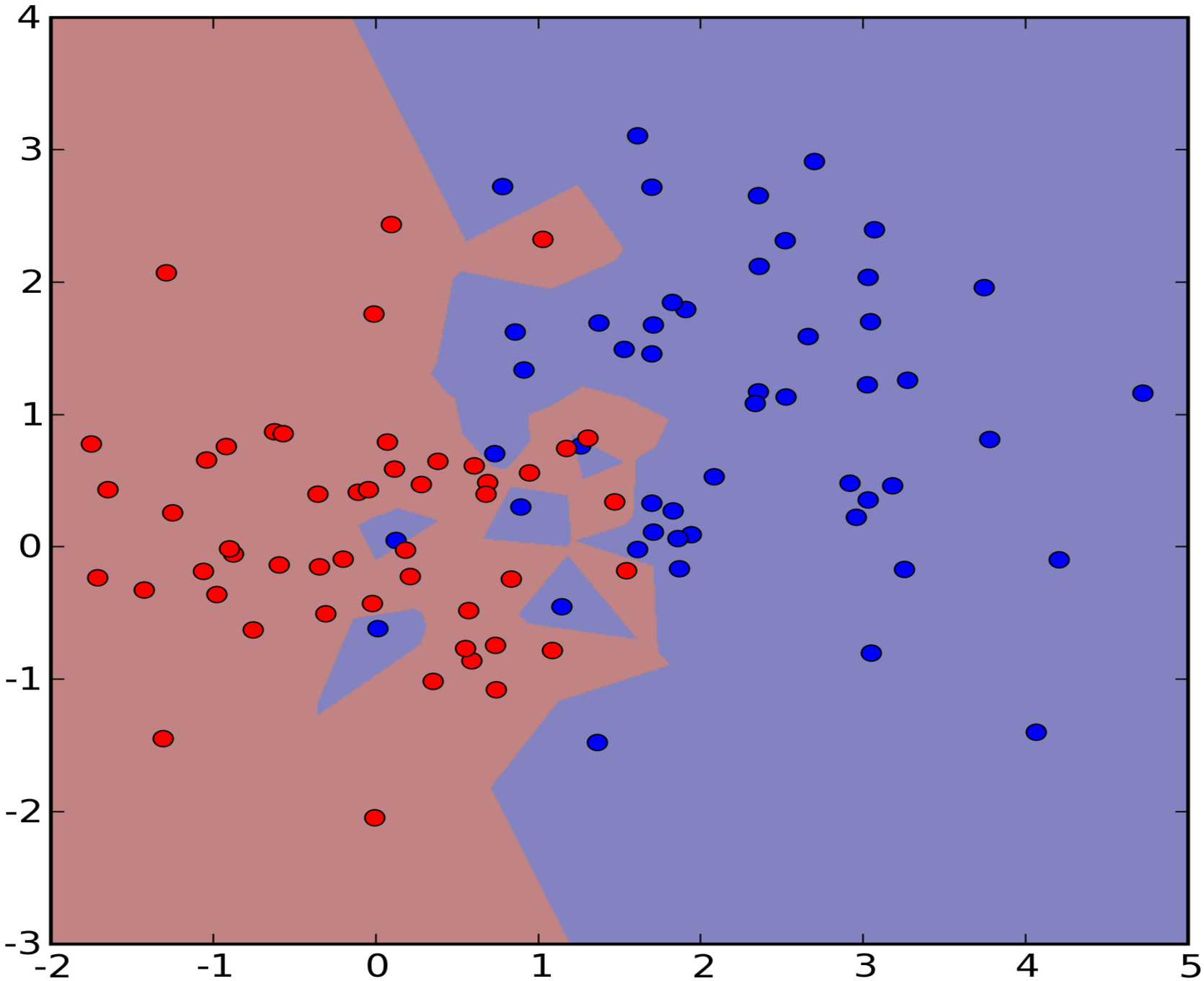

Pixels classified as: grass
500
1000
1500
2000
1000
2000
3000


Pixels classified as: grass
500
1000
1500
2000
1000
2000
3000

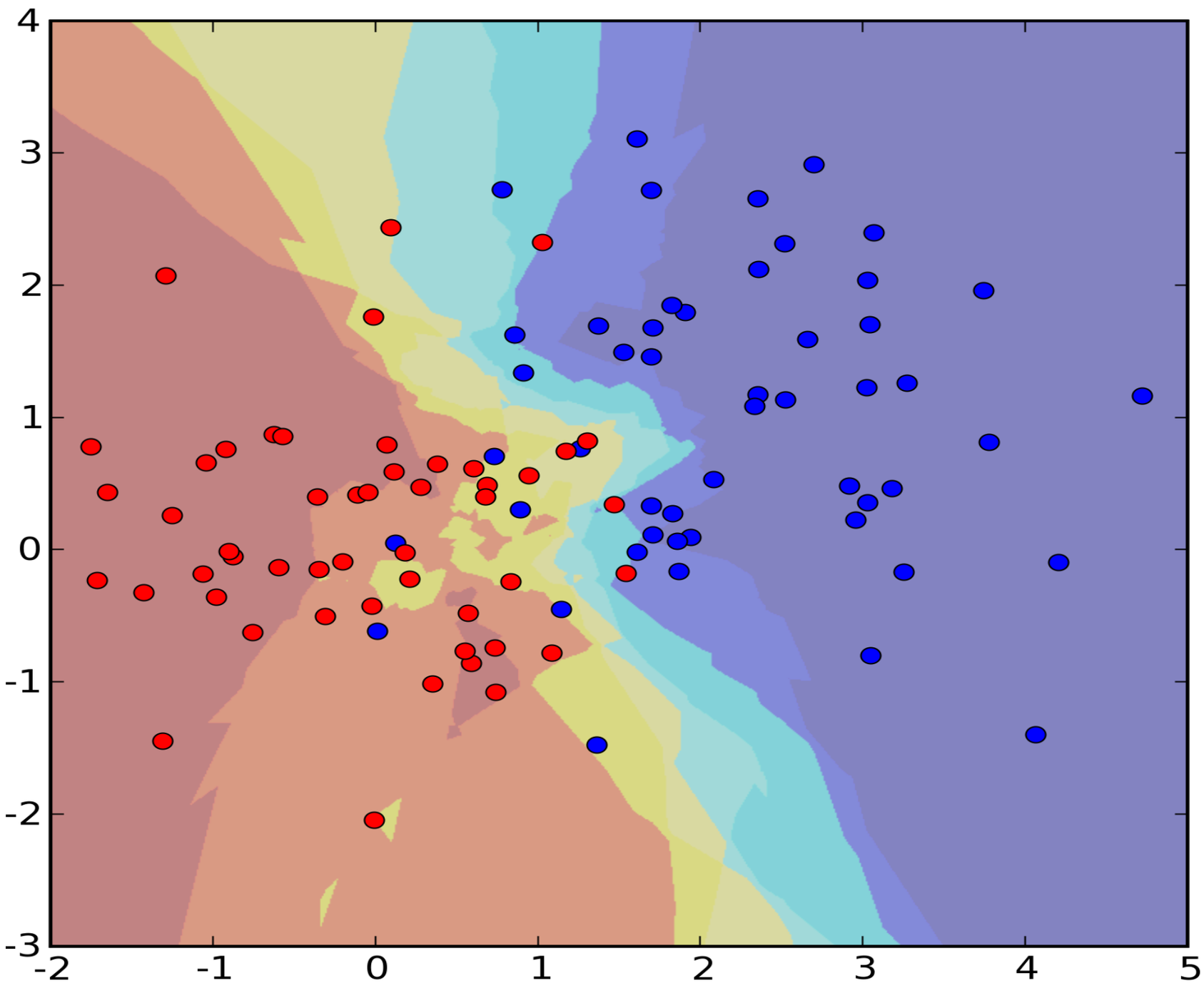
Nearest Neighbour Classification NN and K-NN

- Classify using training data (x_i,y_i)
- NN: Use the label of the nearest neighbour
- KNN: Use the label of the majority of the k nearest neigbhours
- Regression: Use the average of the value of the k nearest neighbours
- Easy to implement and understand
- Can use arbitrary distance functions between images
- Converges to the optimum
- Slow when using lots of data, need to store all training data, not smooth regression

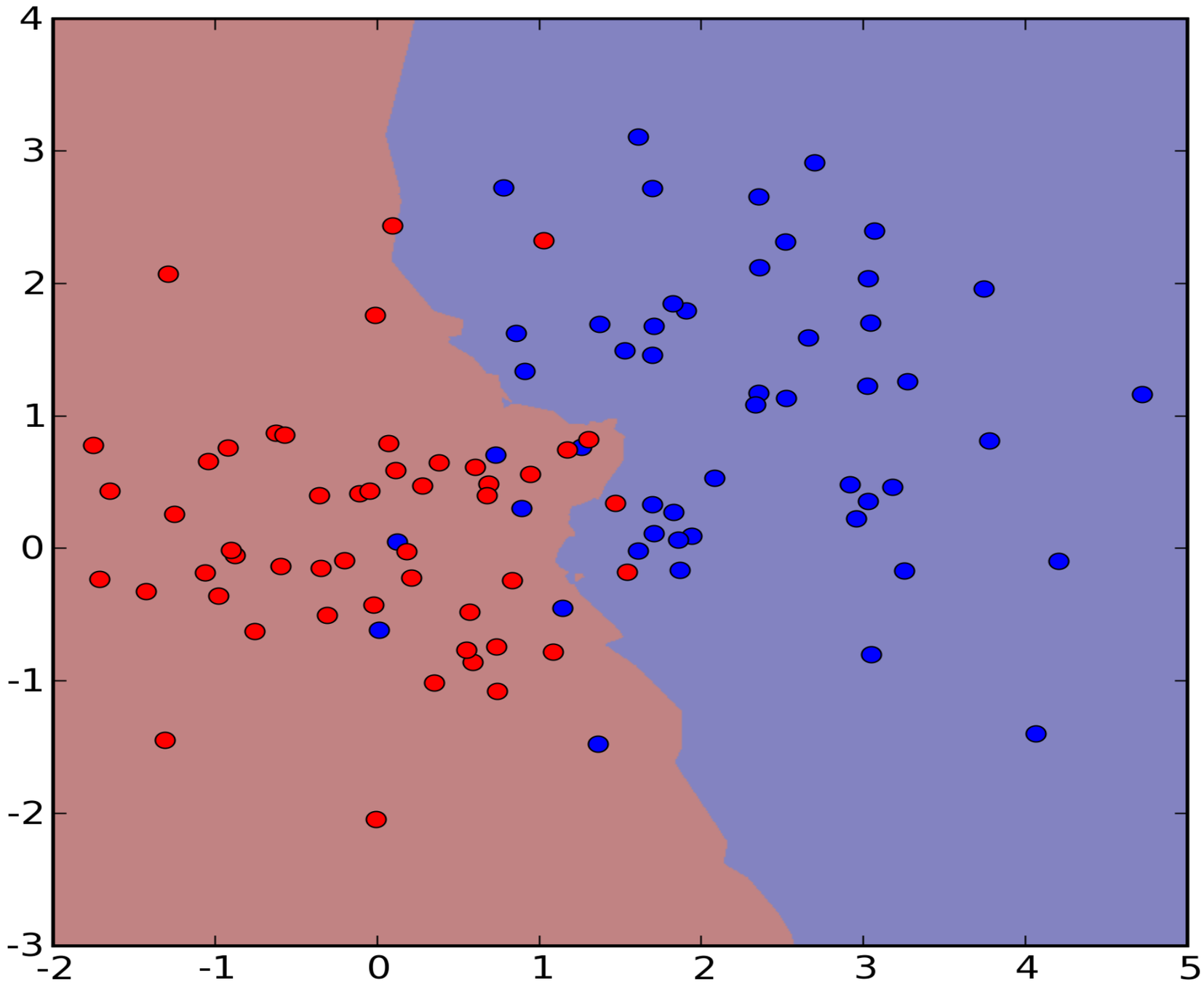
Nearest Neighbour Classification (discussion)



7 Nearest Neighbour Classification



7 Nearest Neighbour Classification



Nearest Neighbour Classification NN and K-NN

- Training is easy, just store the training data $T = \{(x_1,y_1), \ldots, (x_N,y_N)\}$
- Works in any dimension
- Works for regression also: Use the average of the value of the k nearest neighbours
- Easy to implement and understand
- Can use arbitrary distance functions between images
- Converges to the optimum

- Slow when using lots of data,
- Need to store all training data
- Not smooth regression