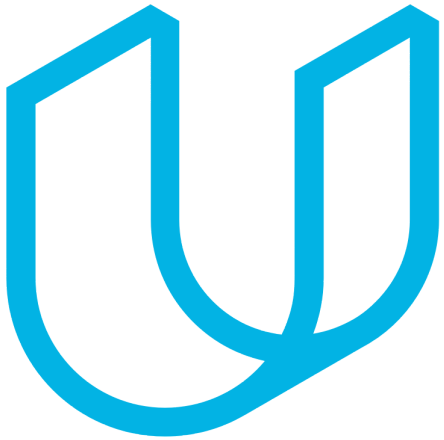
Computer Vision in Python: Object Detection Systems

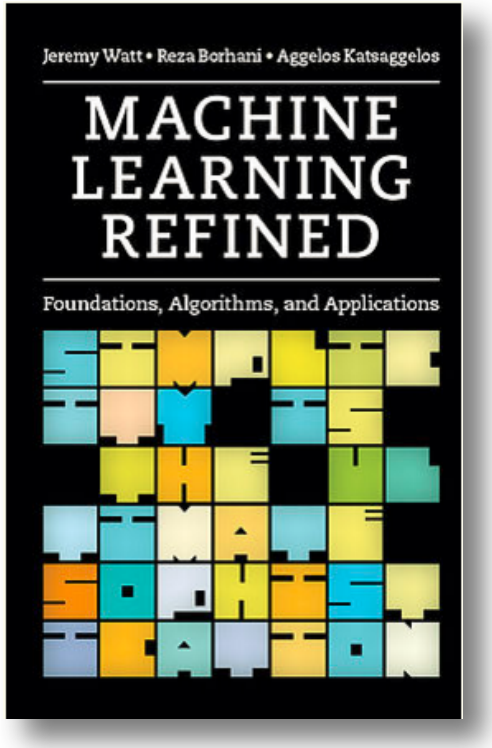
Jeremy Watt
jermwatt@gmail.com

Northwestern University



UDACITY

What is this talk based on?

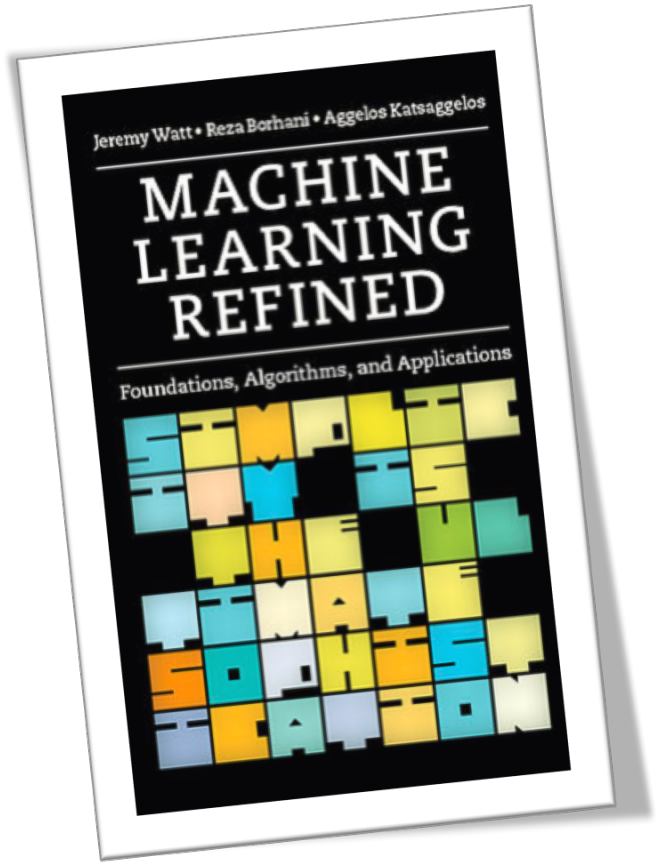


A new ML textbook!
Cambridge University Press
October 2016

Several large ML courses taught at
Northwestern University
Winter 2014, Winter 2015, Fall 2015

What makes our book unique?

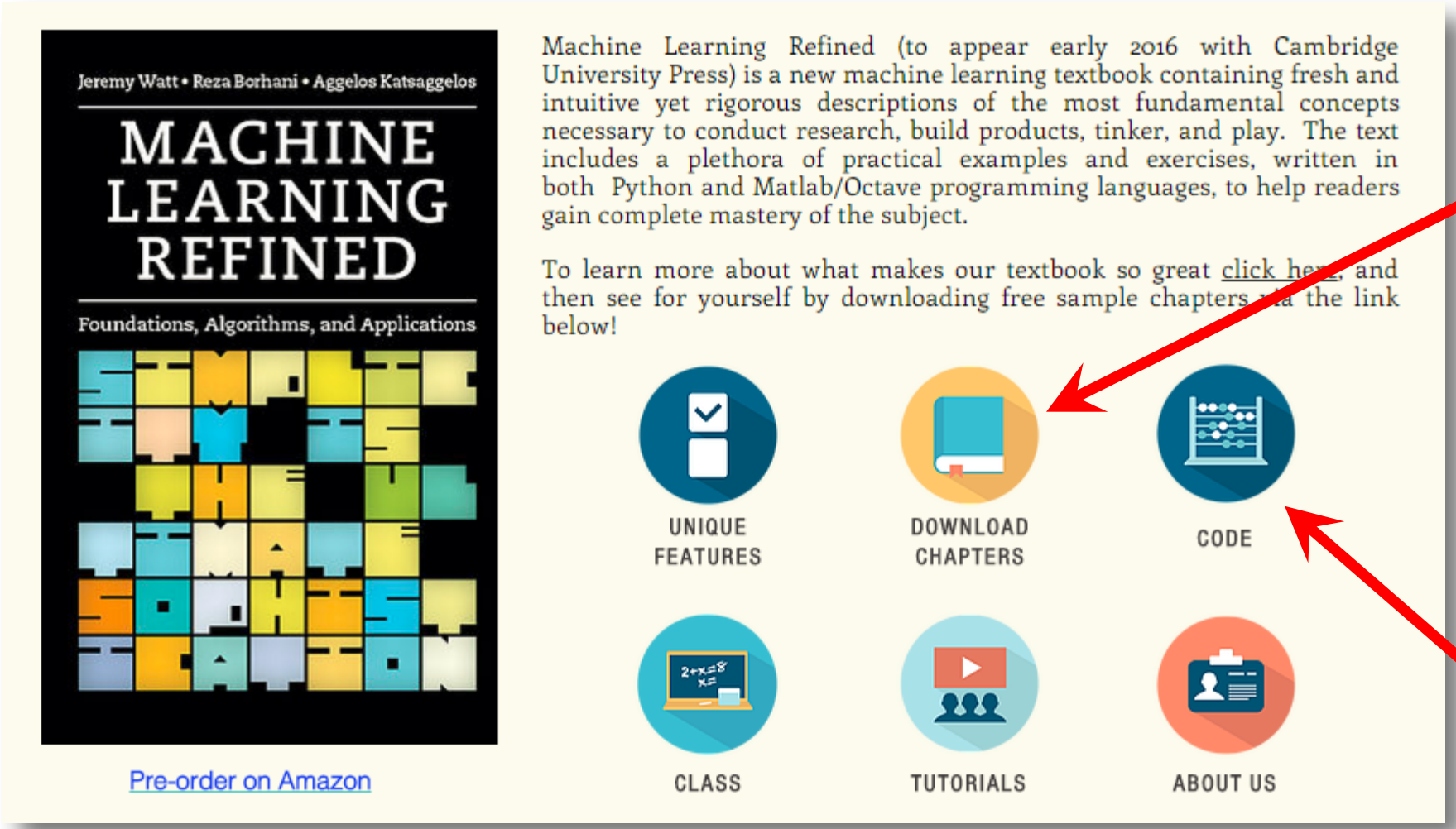
- A presentation built on lucid geometric intuition
  (w/ 200+ color illustrations)
- A learning experience where students learn by doing
  (w/ 50+ in depth coding exercises)
- Inclusion of real application to motivate concepts
  (computer vision, natural language processing, economics, neuroscience, recommender systems, physics, biology, etc.)



- A rigorous yet user friendly presentation of state-of-the-art numerical techniques
  (gradient descent, Newton's method, stochastic gradient descent, accelerated gradient descent, Lipschitz step-length rule, adaptive step-length selection, and more)

Where to download the slides?

www.MLrefined.com



Sample chapters of the book

Slides

What is this talk about?

- Goal: give you an overview of computer vision - object detection – (in Python)

- So, we'll talk about
  - What are the basic building blocks of an object detection system?
  - How do they (basically) work?
  - Where do you find the tools?

Classification tasks

Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

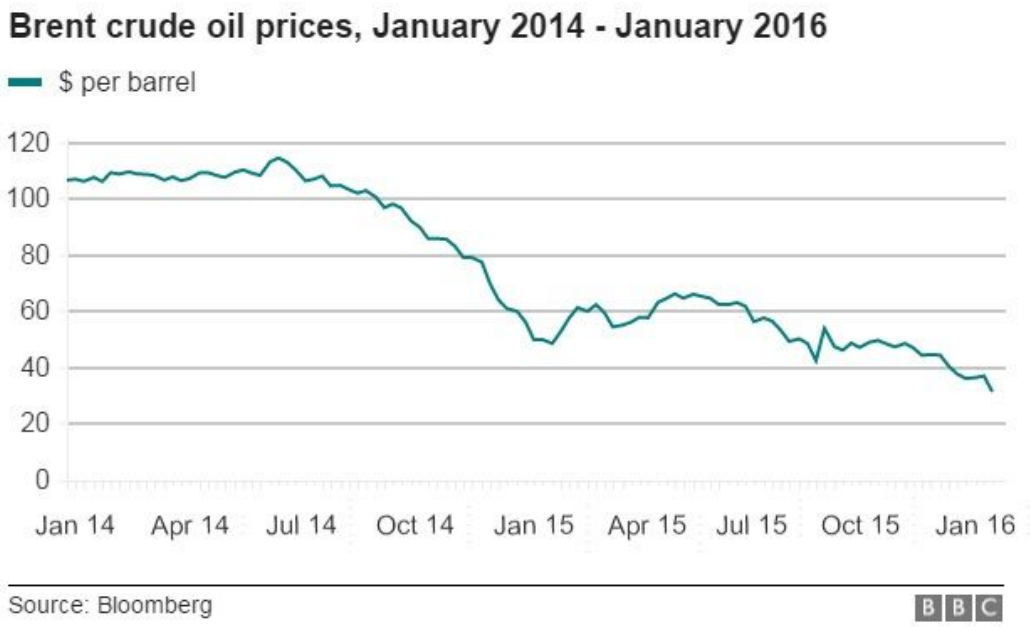
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Financial modeling



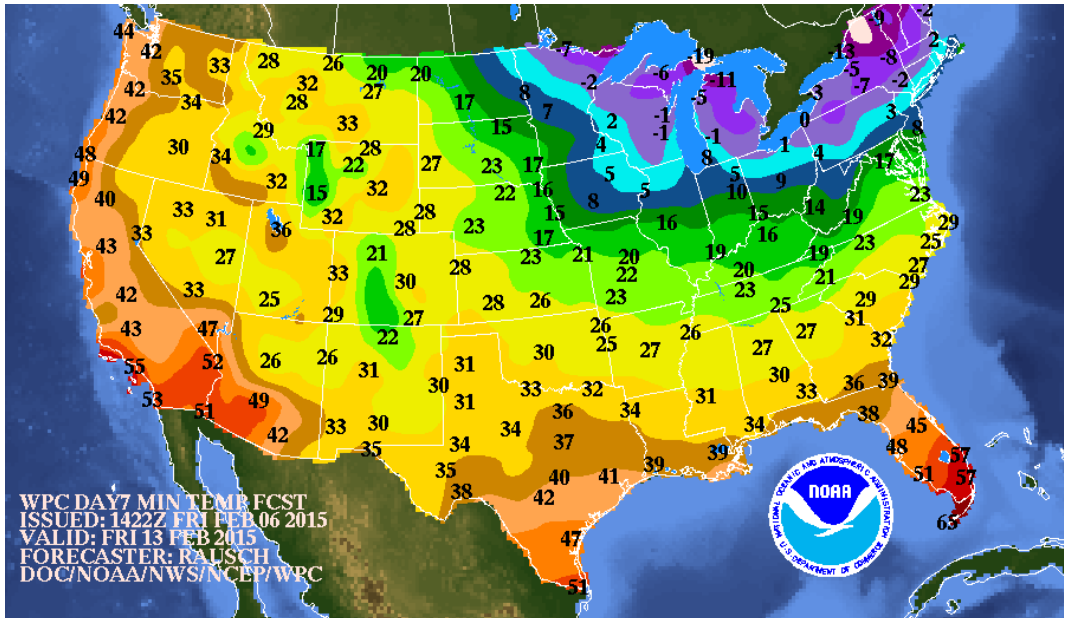
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

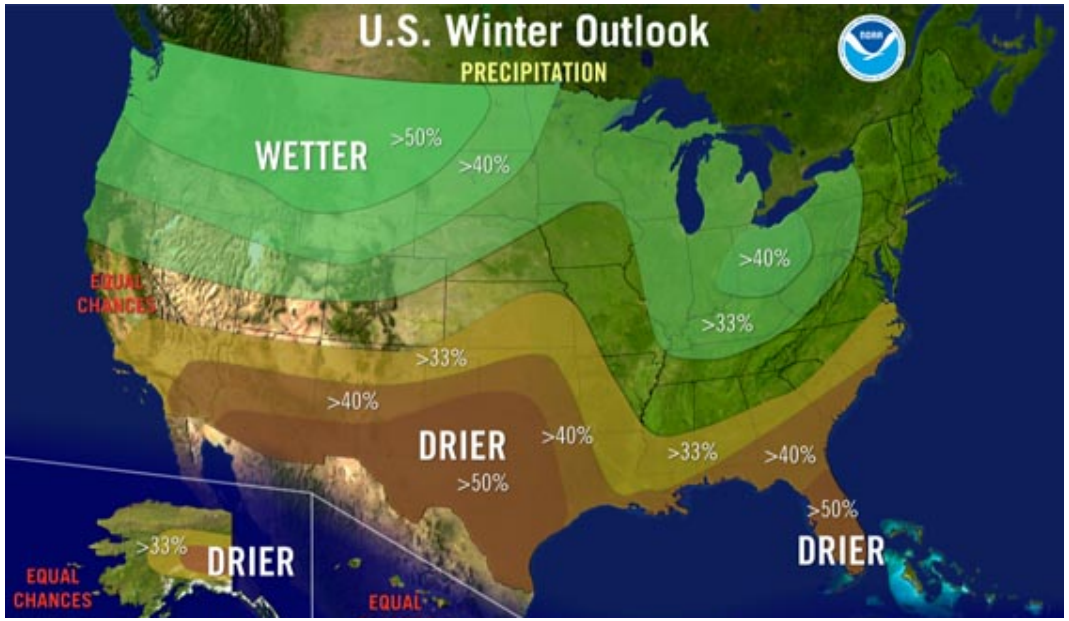
Ex. Weather forecasting



Regression

Predicting a *continuous-valued* variable

Ex. Weather forecasting



Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

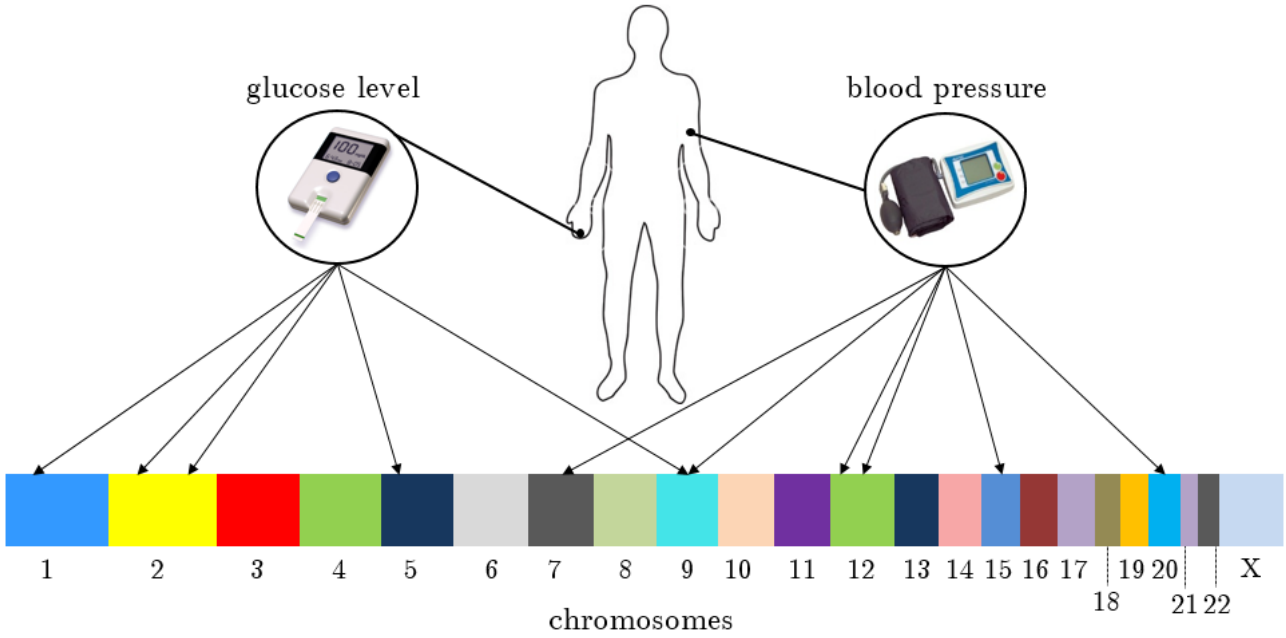
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Genetics



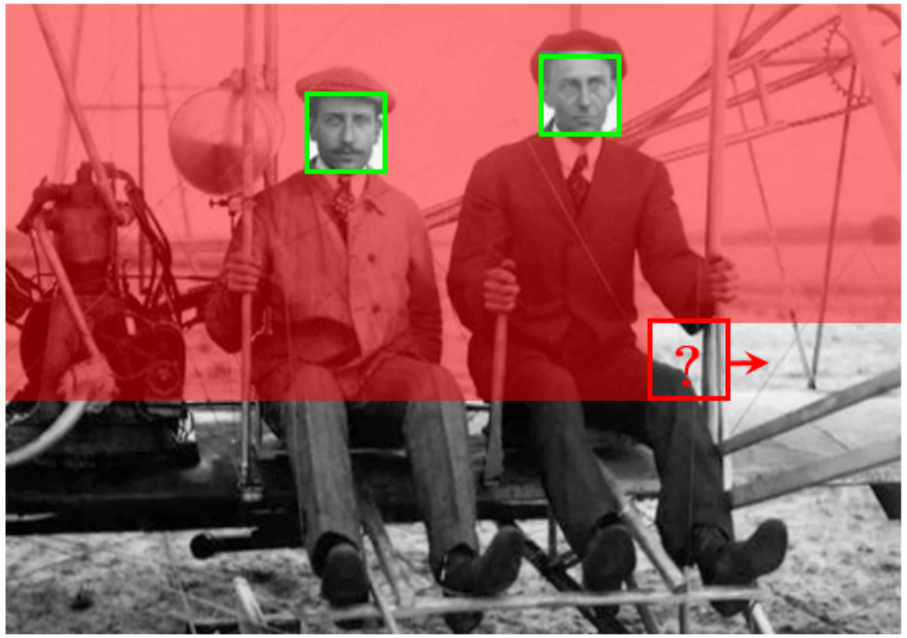
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Face detection



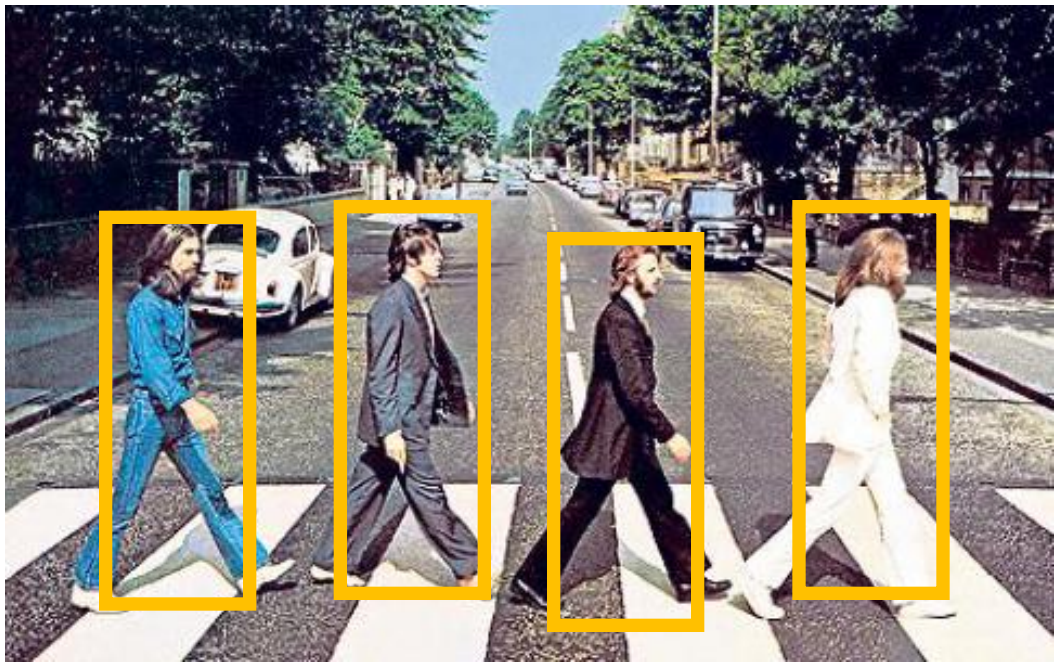
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Pedestrian detection



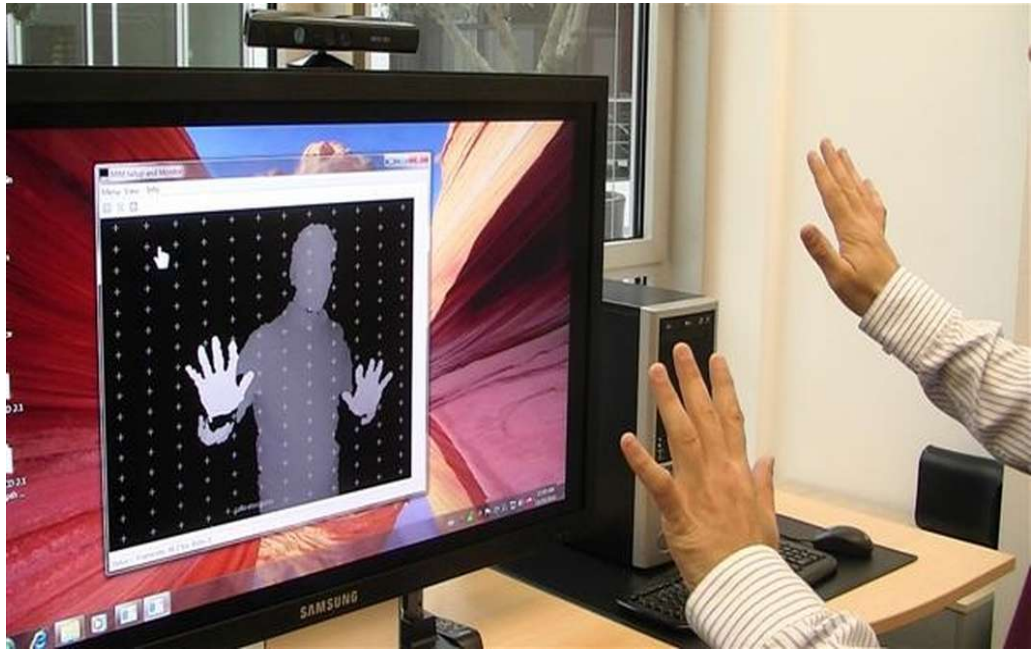
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Hand gesture recognition



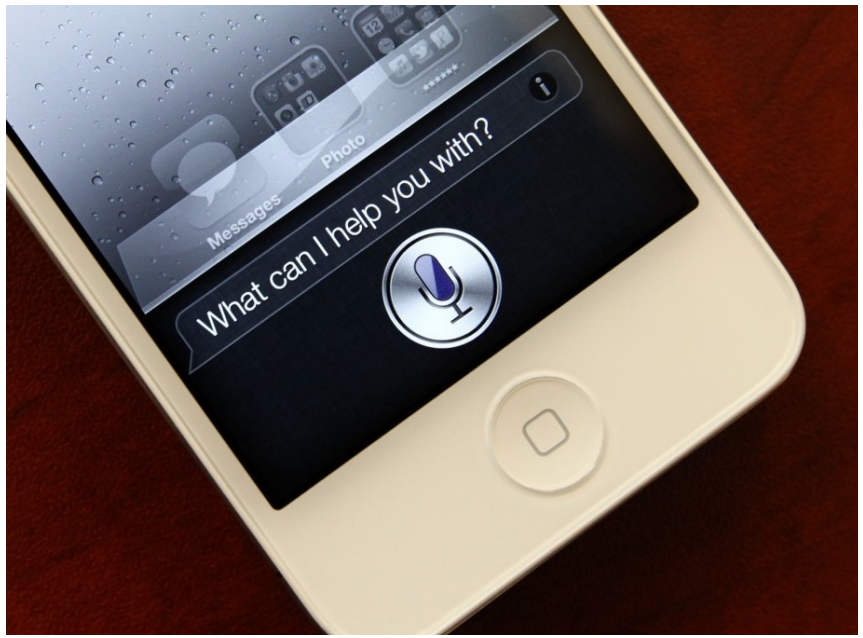
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Speech recognition



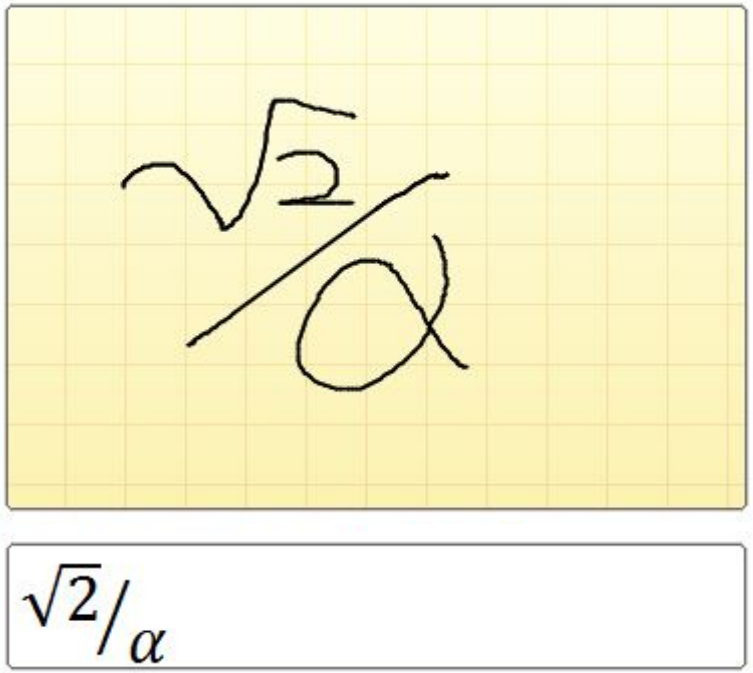
Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Optical character recognition



Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Sentiment analysis

Regression

Predicting a *continuous-valued* variable

Classification

Predicting a *discrete-valued* variable
i.e., distinguishing between different classes of data

Ex. Spam detection

Linear Classification

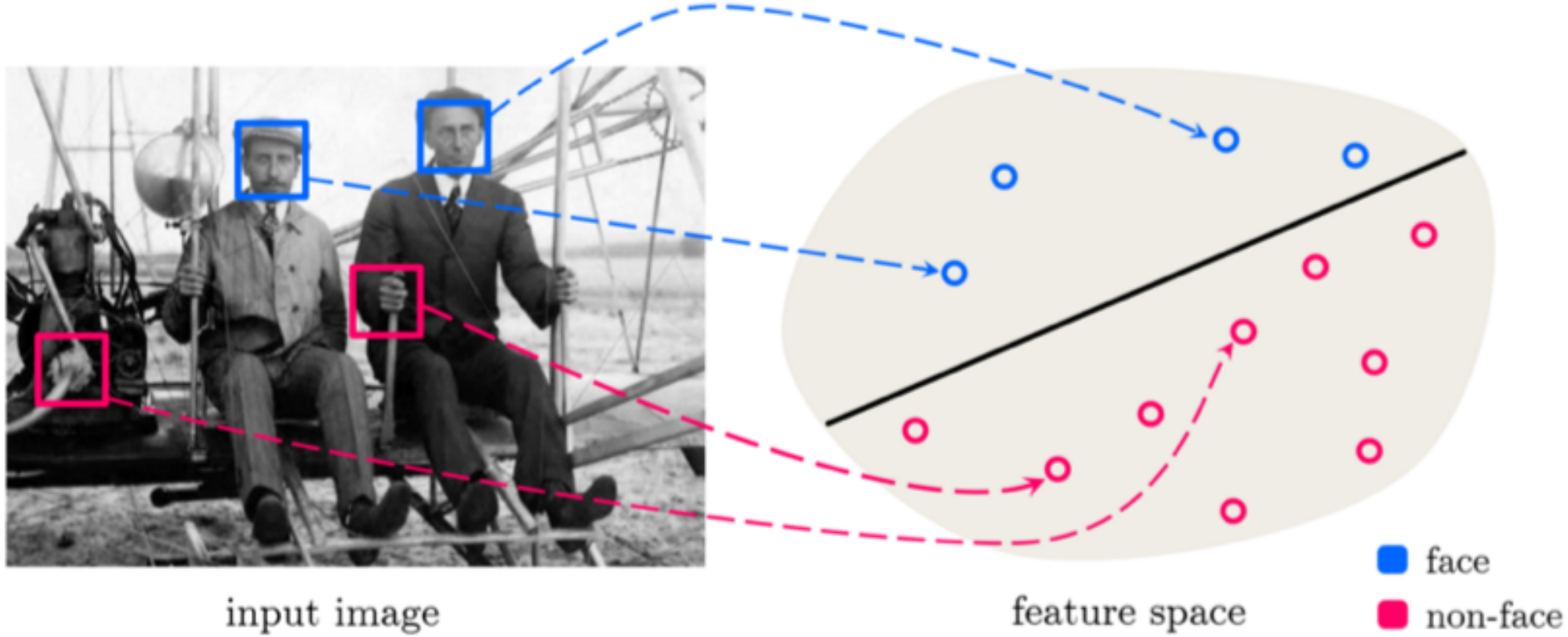


Image taken from [29]

Logistic regression

$$y(\mathbf{x}) = \text{sign}\left(b + \mathbf{x}^T\mathbf{w}\right)$$

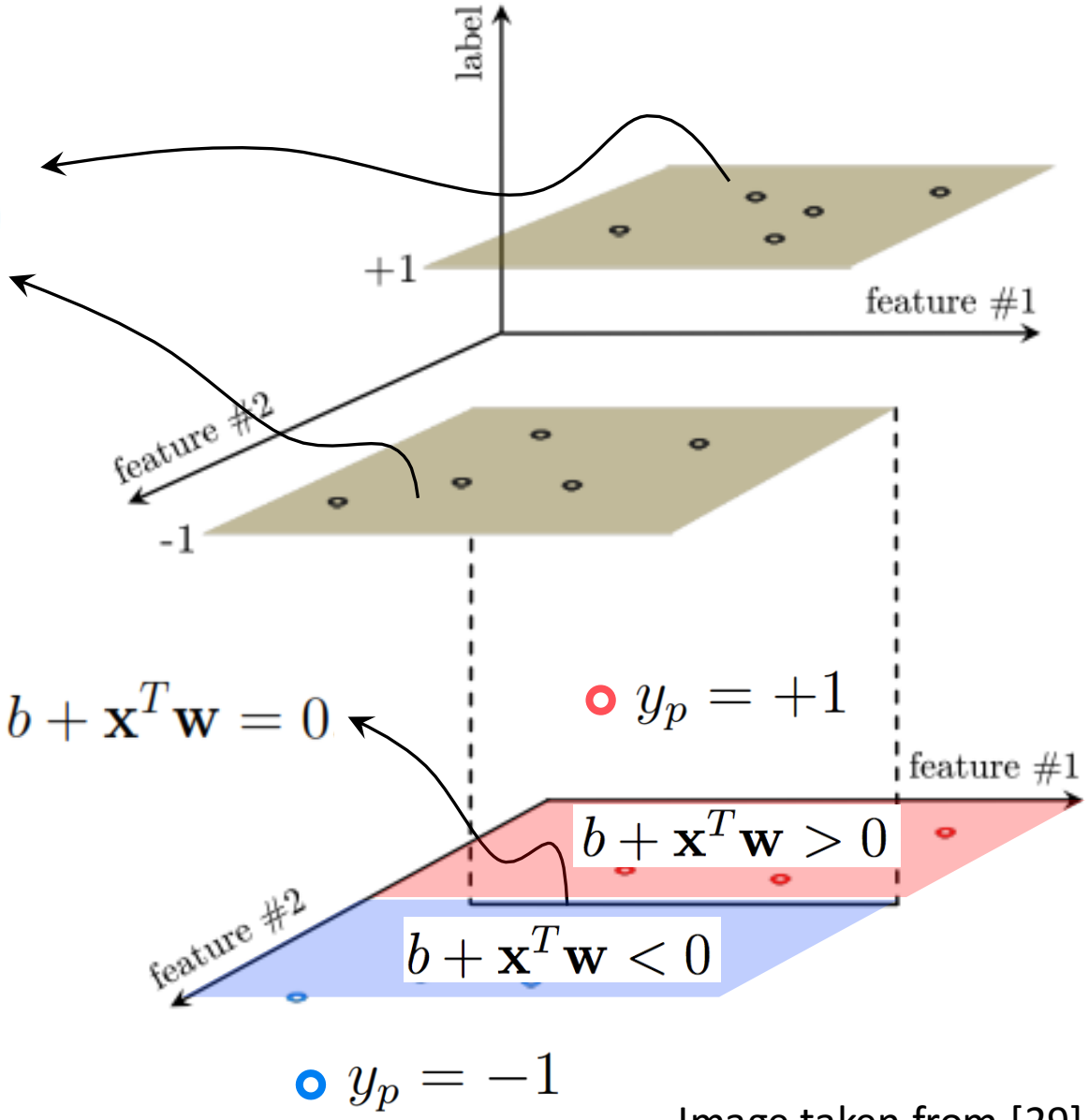


Image taken from [29]

Logistic regression

$$y(\mathbf{x}) = \tanh\left(b + \mathbf{x}^T\mathbf{w}\right)$$

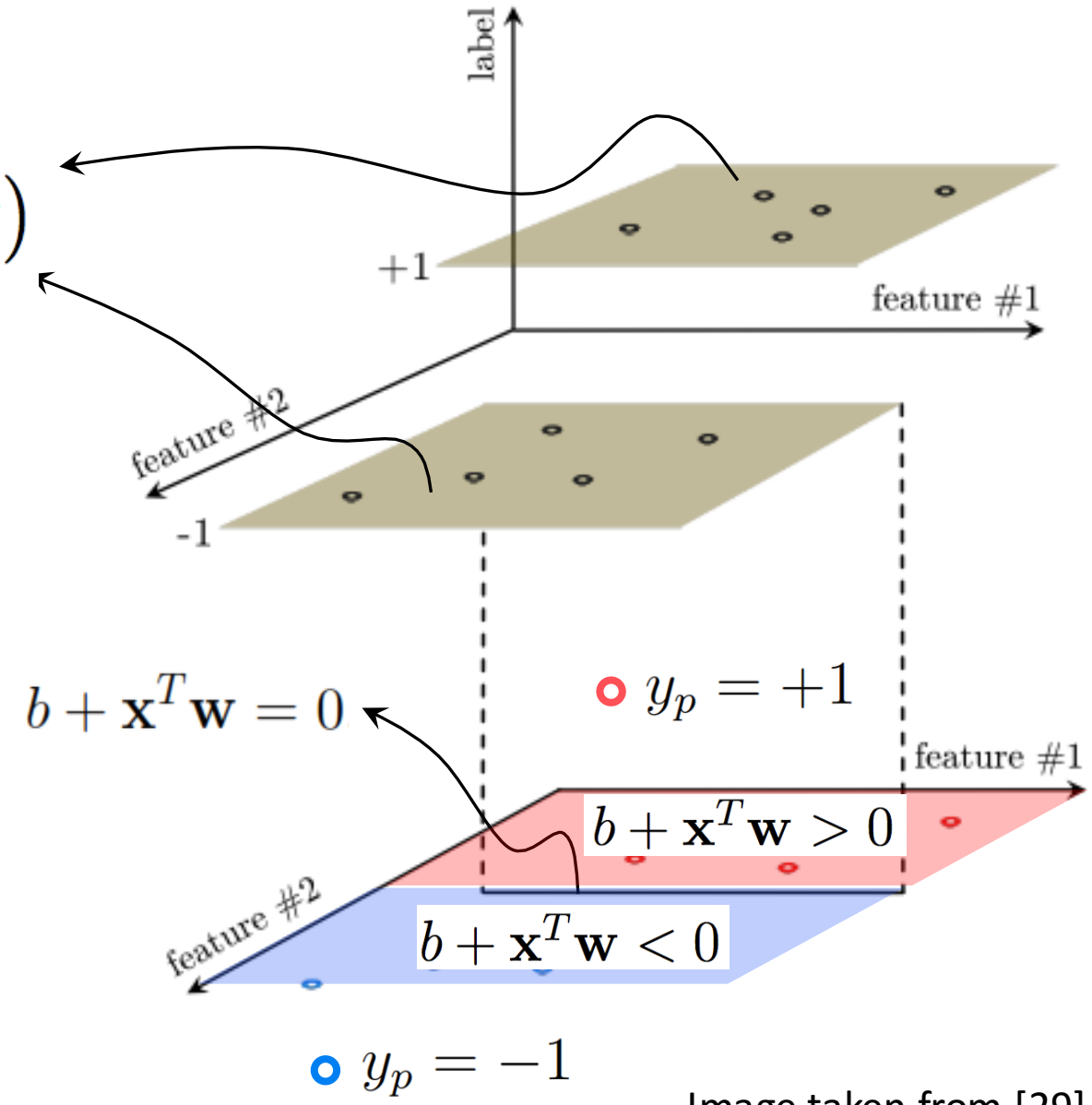


Image taken from [29]

of [29]) one can form a recovery problem whose solution gives parameters satisfying this desired relationship

$\Downarrow$

best parameters $(b^\star, \mathbf{w}^\star)$

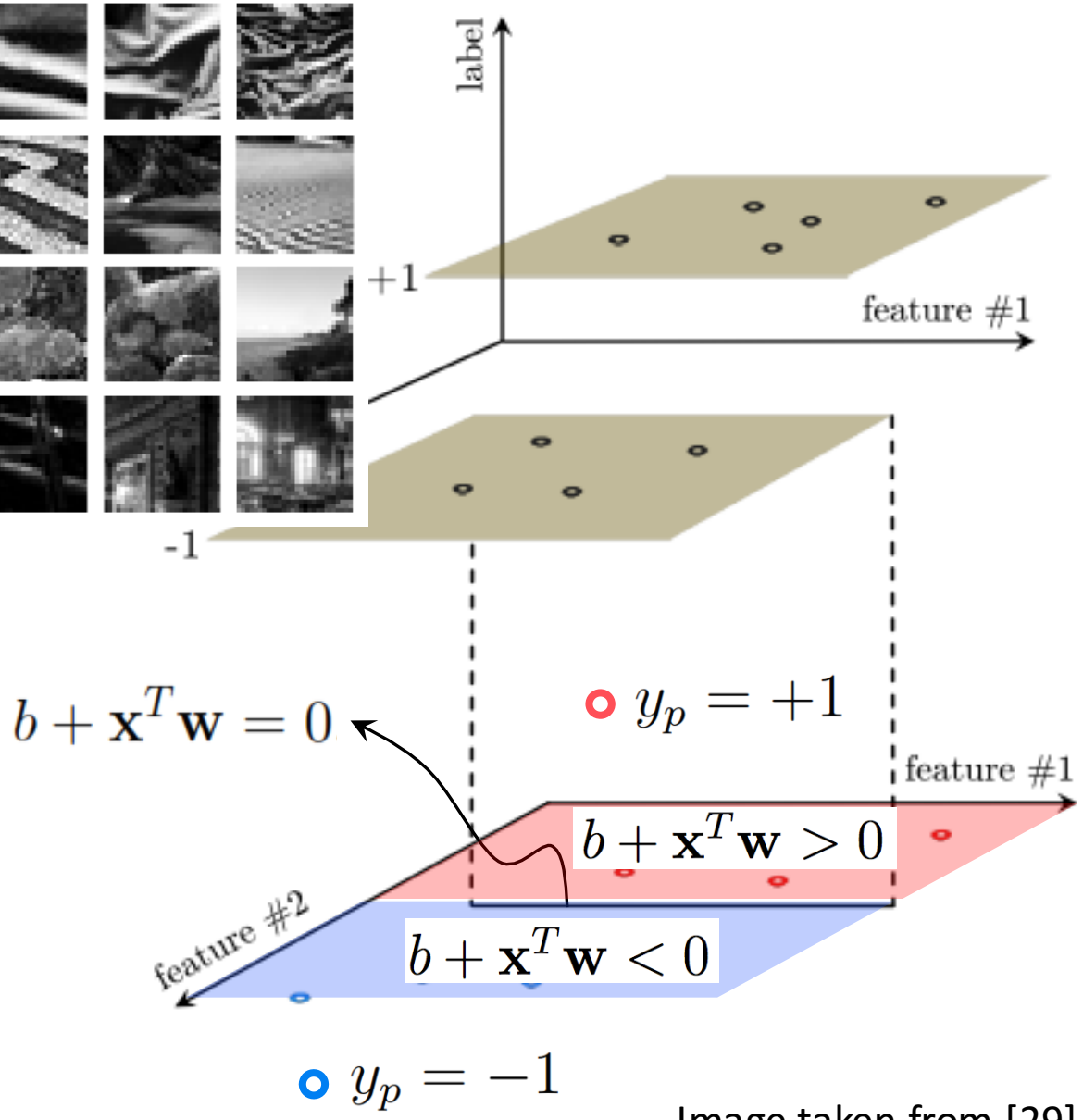


Image taken from [29]

What are “features” ?

Features

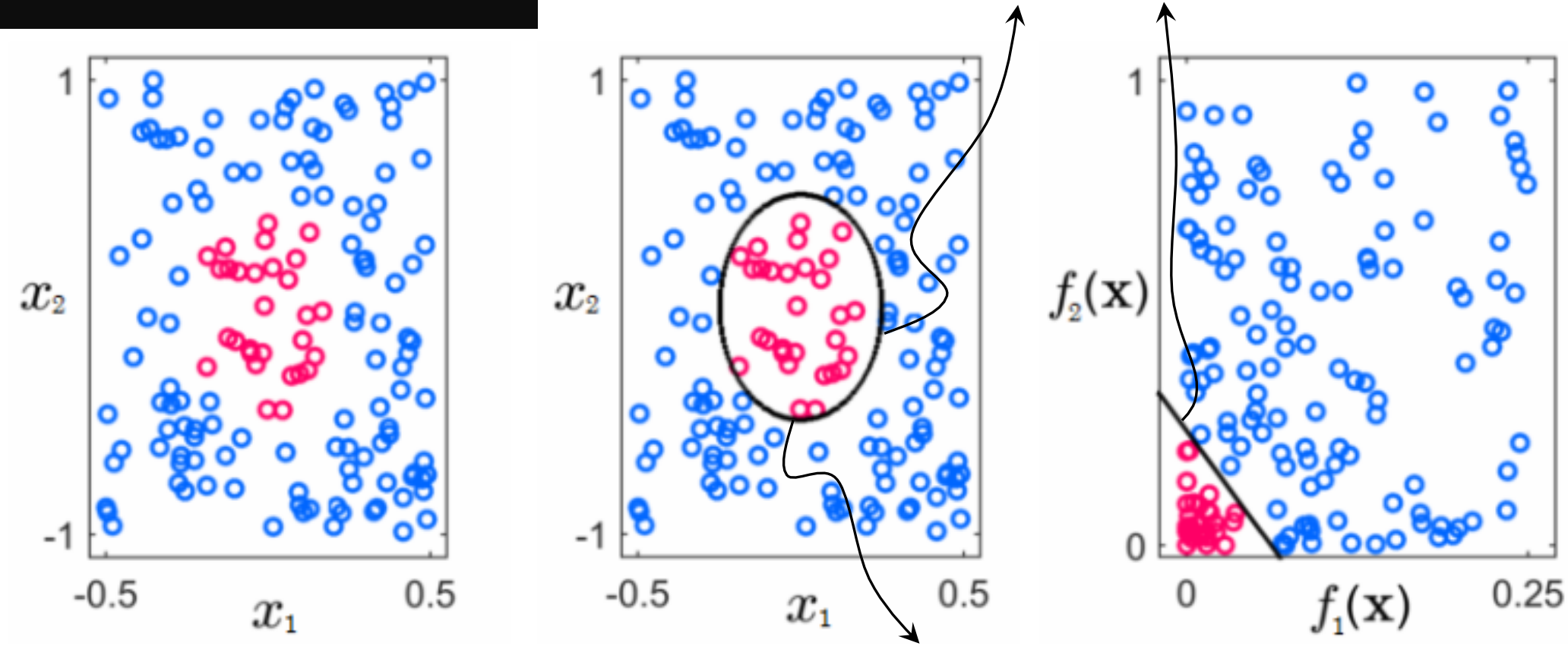
What are features?

- **Geometrically:** features are transformations of the input that provoke a good *nonlinear* fit/ separation

- **Philosophically:** features are those defining characteristics of the phenomenon underlying a dataset that allow for optimal learning

Feature design: classification

$$1 + f_1(\mathbf{x})\, w_1 + f_2(\mathbf{x})\, w_2 = 0$$

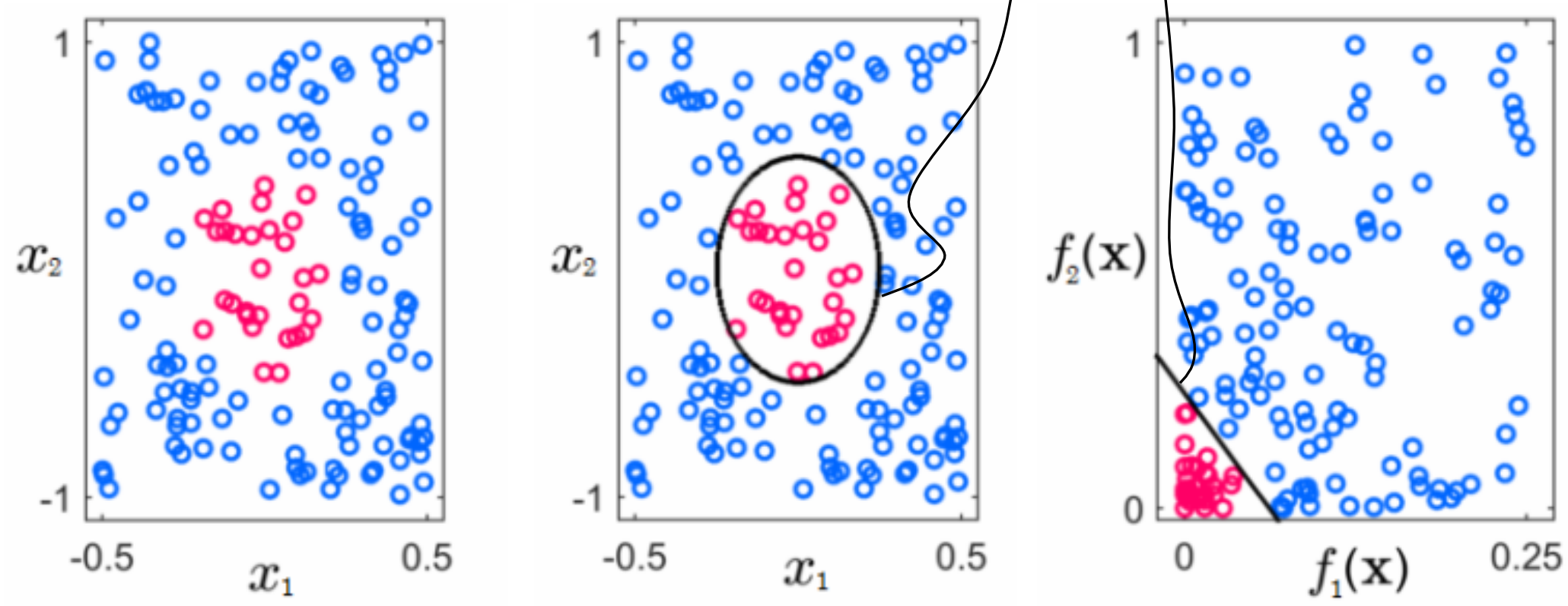


$$1 + x_1^2 w_1 + x_2^2 w_2 = 0$$

Note the feature and output are *linearly* related in w_1 and w_2

Feature design: classification

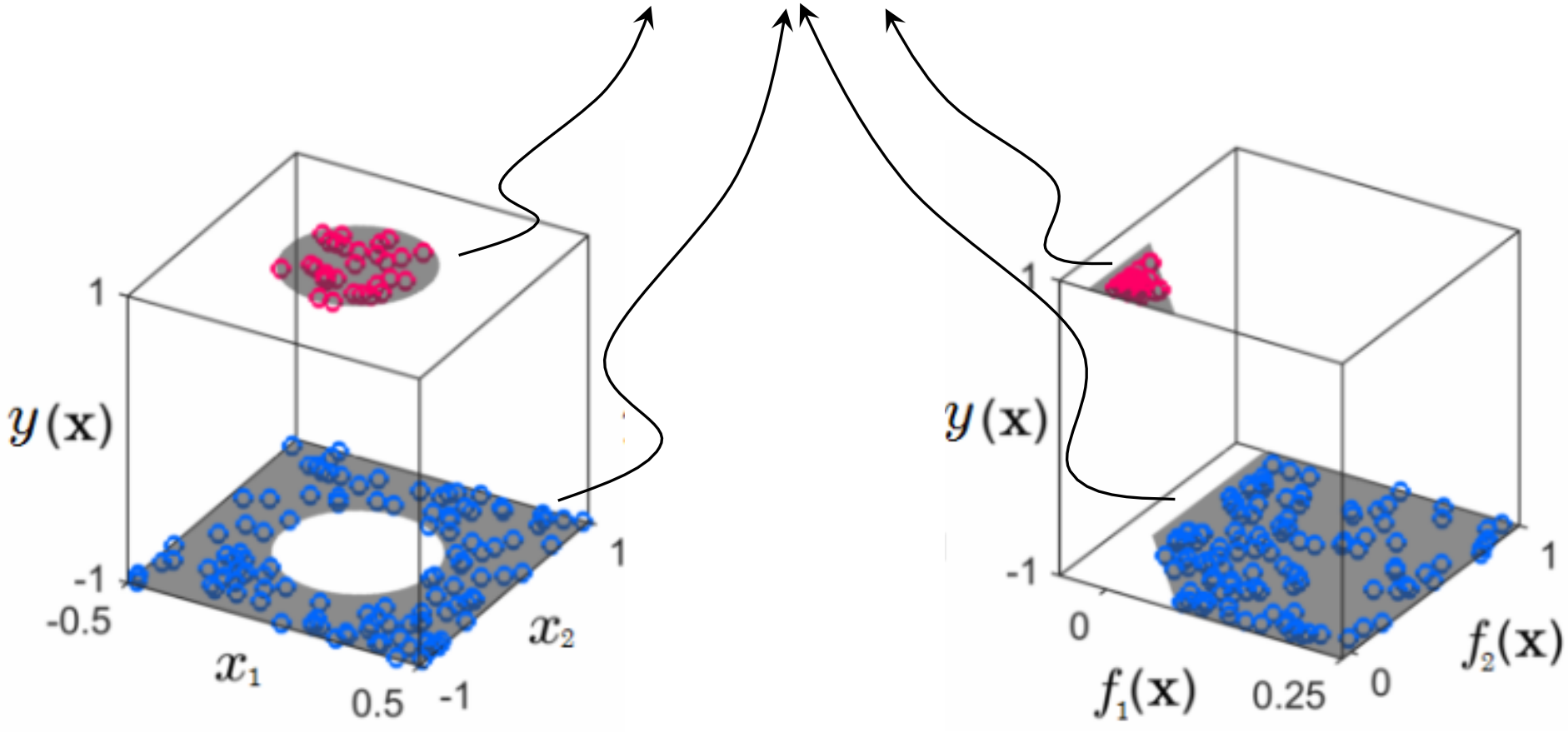
$$1 + f_1(\mathbf{x})\, w_1 + f_2(\mathbf{x})\, w_2 = 0$$



Properly designed features for linear classification provide good *nonlinear* separation in the original feature space and, simultaneously, good *linear* separation in the transformed feature space.

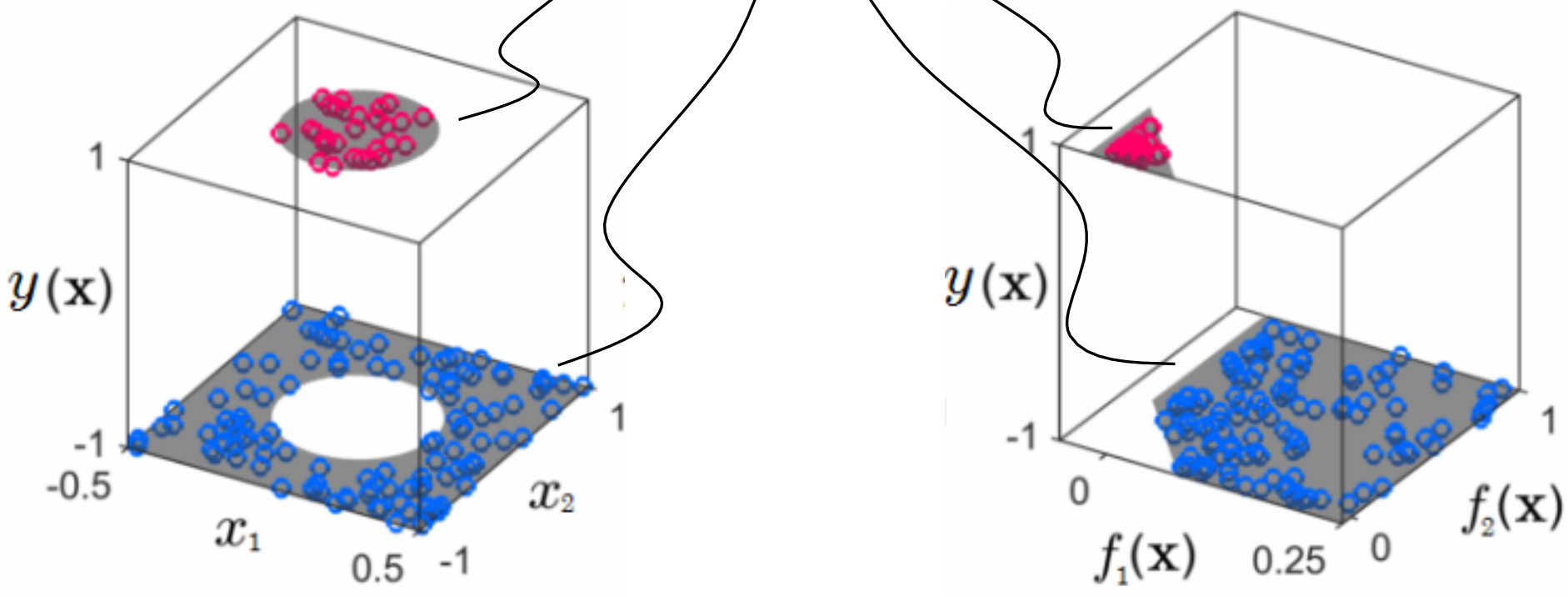
Full view of data

$$y(\mathbf{x}) = \text{sign}\left(1 + f_1(\mathbf{x})\, w_1 + f_2(\mathbf{x})\, w_2\right)$$



Full view of data

$$y(\mathbf{x}) = \text{sign}\left(b + \sum_{m=1}^{M} f_m(\mathbf{x})\, w_m\right)$$



Features: summary (re-visited)

What are features?

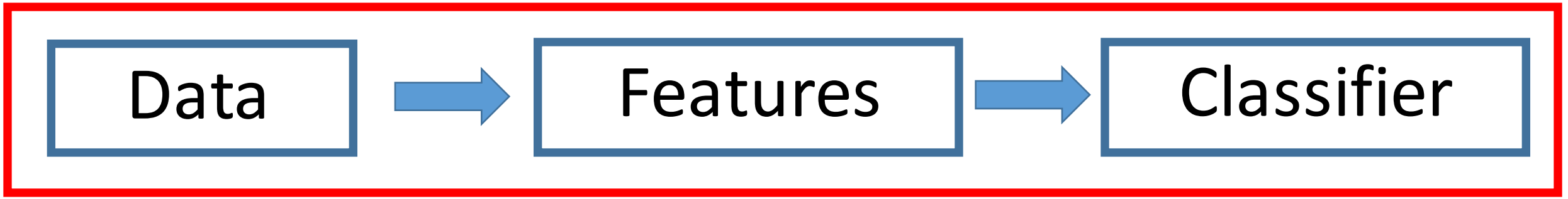
- **Geometrically:** features are transformations of the input that provoke a good *nonlinear* fit/separation

- **Philosophically:** features are those defining characteristics of the phenomenon underlying a dataset that allow for optimal learning

How do we design good features?

- **By hand (engineered):** translate understanding of a phenomenon into a set of geometric transformations of the input – often requires expertise
  - Requires strong knowledge to produce reasonable results
- **Learn from data (automatic):** use bases (e.g., neural networks) to determine directly from the data
  - Requires large datasets to produce reasonable results (more on this in part II of talk!)

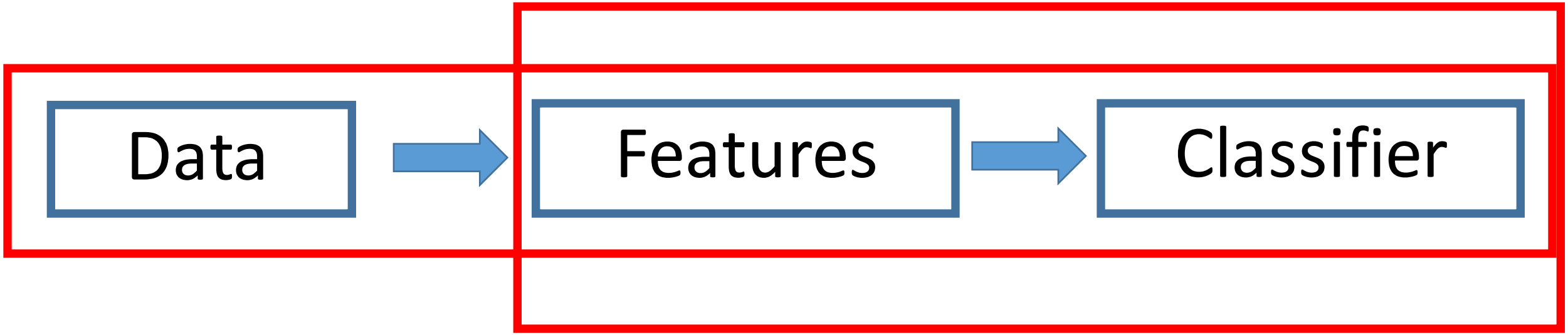
Detection Pipeline



- UCI ML repo
- KD Nuggets
- Kaggle
- ....

- scikit-learn
- scikit-image
- opencv
- tensorflow / theano / caffe

Detection Pipeline



These two can be performed simultaneously

- e.g., Viola-Jones face detector, deep neural nets, kernels,...

Classic (hand) engineered features

Commonly-used engineered features

TEXT

- Sentiment analysis

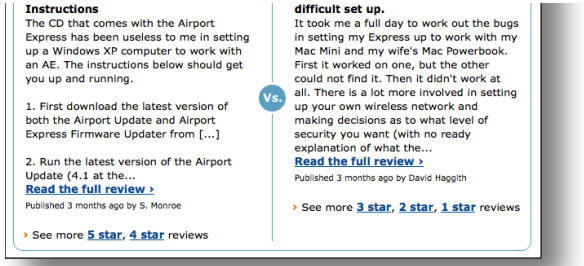
IMAGE

- Pedestrian detection

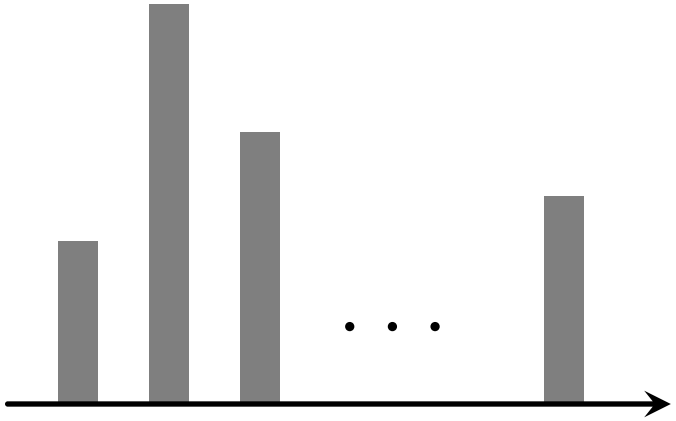
AUDIO

- Speech recognition

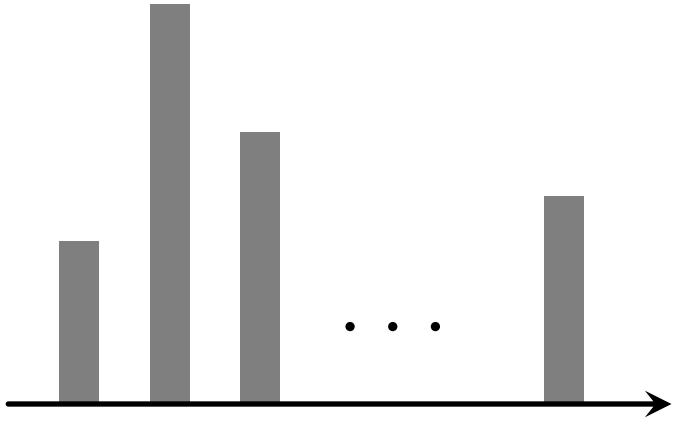
Feature transformations for real data often consist of discrete processing steps which aim at ensuring that instances within a single class are "similar" while those from different classes are "dissimilar". These processing steps are still feature transformations $f_1(\mathbf{x}), ..., f_M(\mathbf{x})$ of the input data, but they are not so easily expressed algebraically.



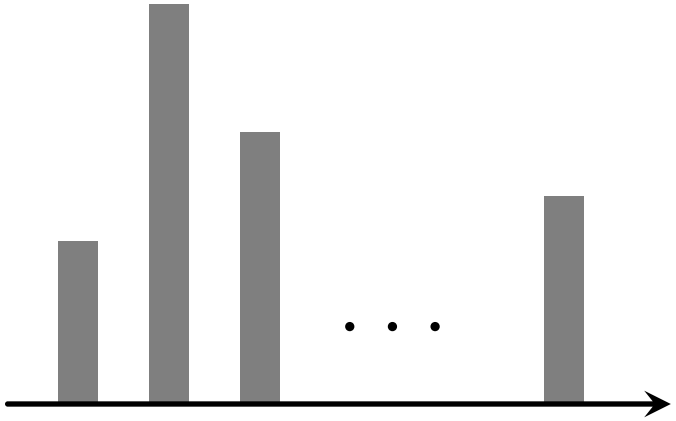
Histogram features



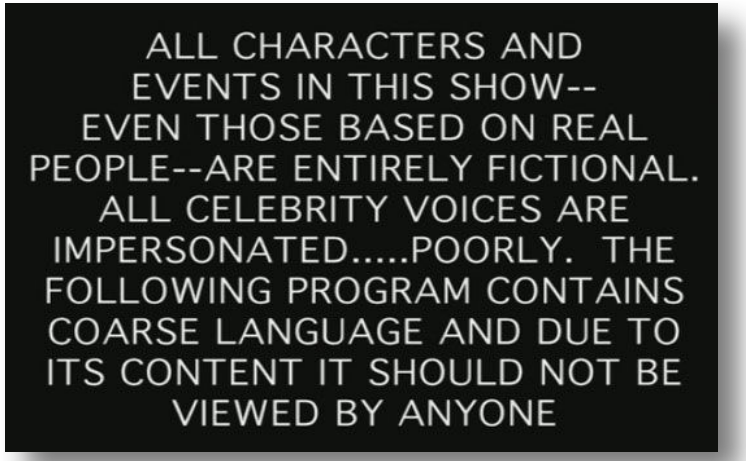
words

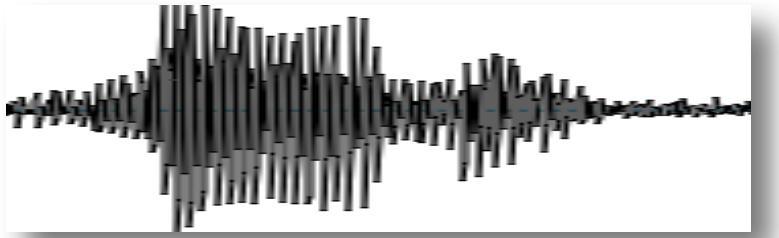


edge content



frequency





Bag of Words (BoW) histogram

1) dogs are the best
2) cats are the worst

- ❑ Parsing
- ❑ Stop word removal
- ❑ Stemming
- ❑ Merging
- ❑ Normalized vector representation

Parsing	dogs / are / the / best	cats / are / the / worst
Stop word removal	dogs / ~~are~~ / ~~the~~ / best	cats / ~~are~~ / ~~the~~ / worst
Stemming	dog~~s~~ best	cat~~s~~ worst
Merging	best cat dog worst	

$$\mathbf{x}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{pmatrix} \text{best} \\ \text{cat} \\ \text{dog} \\ \text{worst} \end{pmatrix} \qquad \mathbf{x}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \begin{pmatrix} \text{best} \\ \text{cat} \\ \text{dog} \\ \text{worst} \end{pmatrix}$$

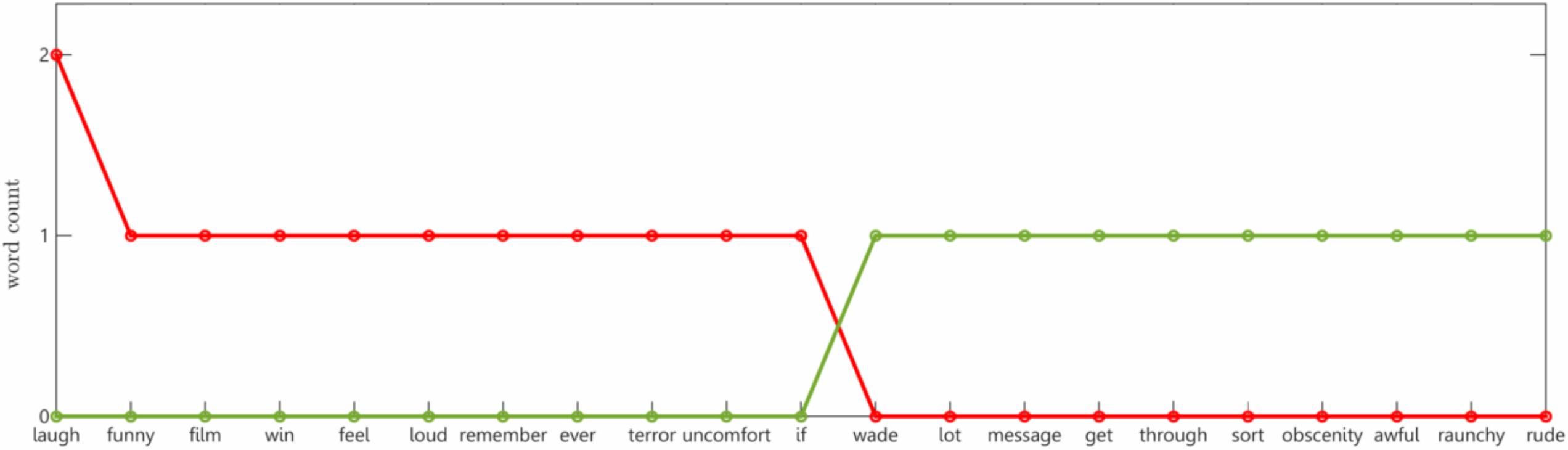
Sentiment analysis with BoWs

One of the funniest films you will ever feel this uncomfortable about laughing out loud at. Remember if you laugh, the terrorists win!

Kevin Ranson
MovieCrypt.com

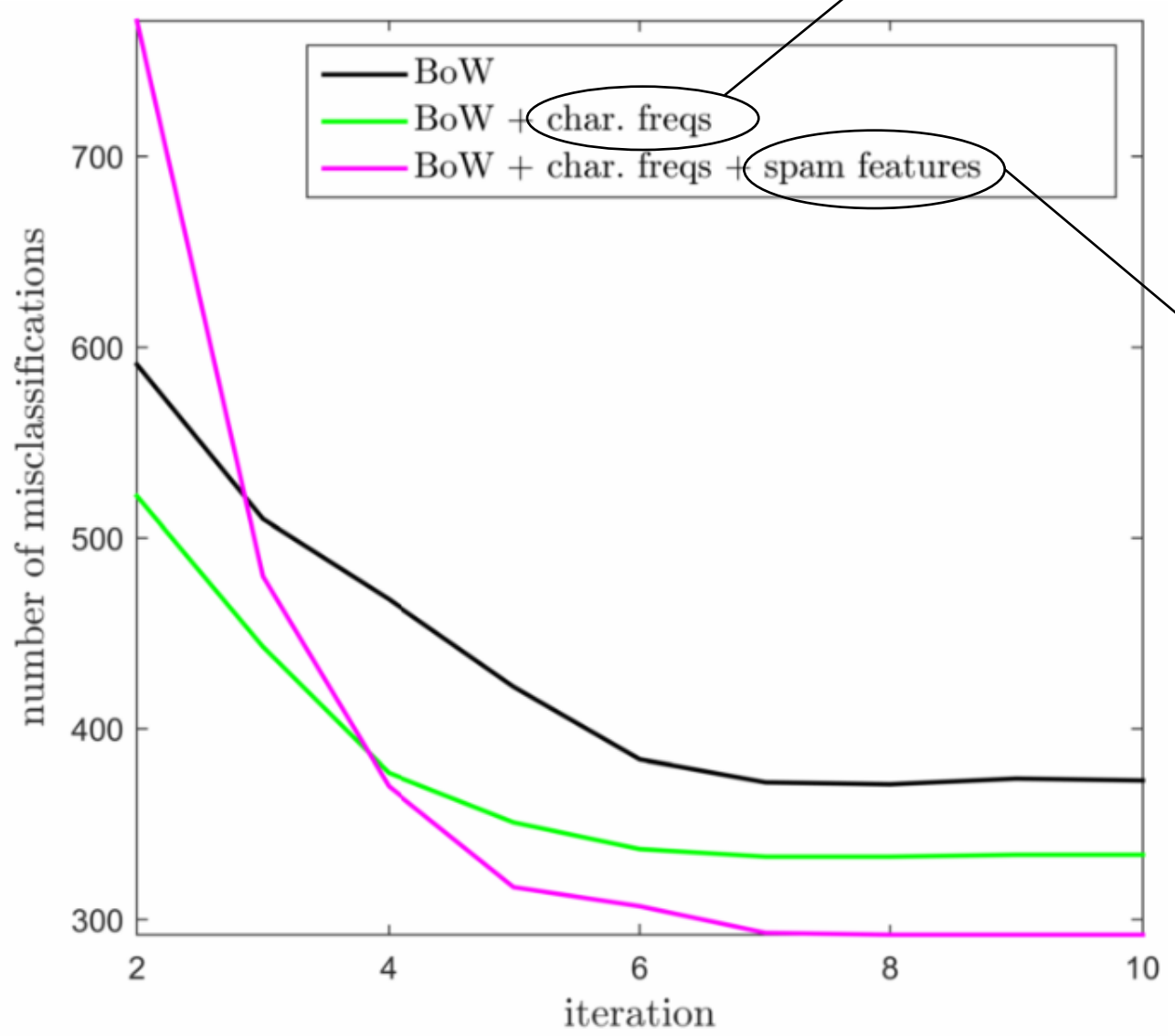
It is rude and raunchy, and it has a message, sort of. You have to wade through an awful lot of obscenity to get to it though.

Roger Moore
Orlando Sentinel



Spam detection

Frequency of certain characters like ‘ ! ’ and ‘ * ’

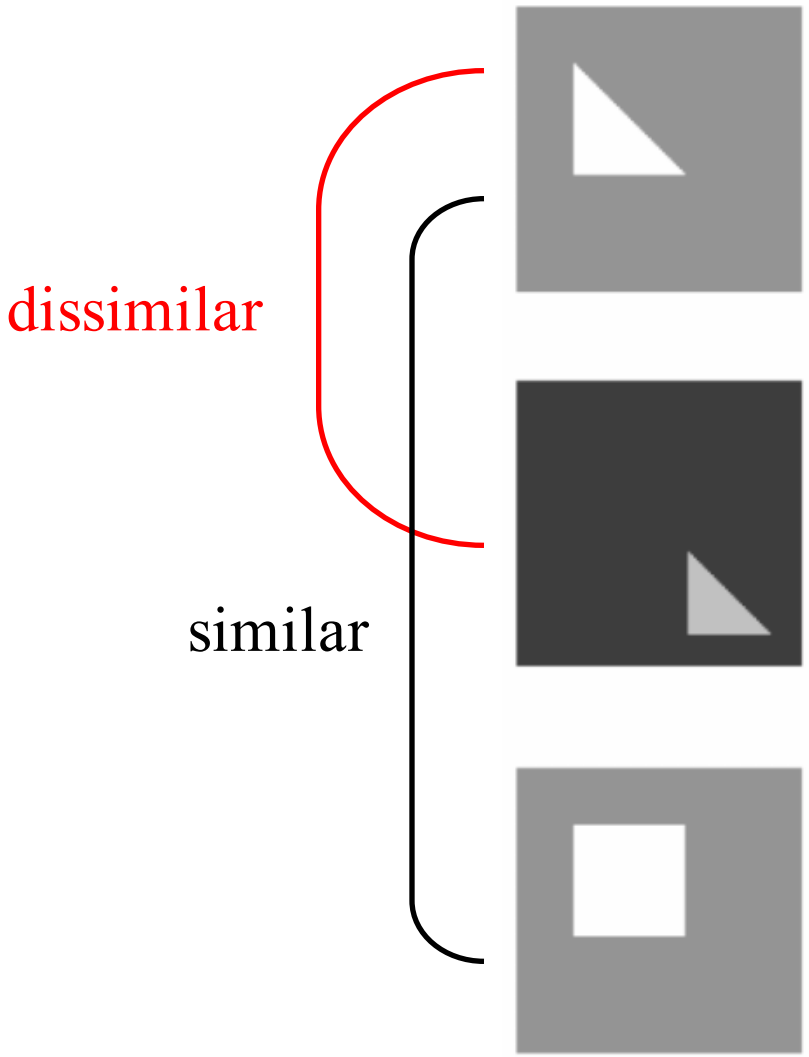


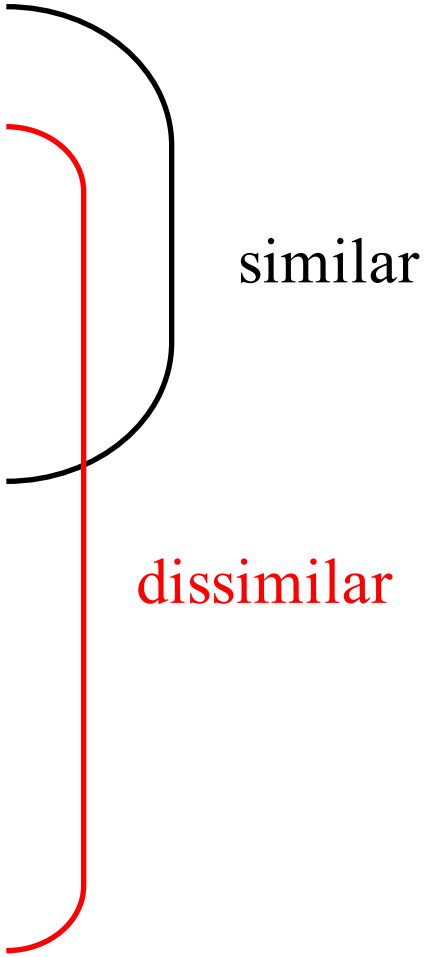
- Percentage of capital letters
- Length of longest uninterrupted sequence of capital letters
- …

Images and edges

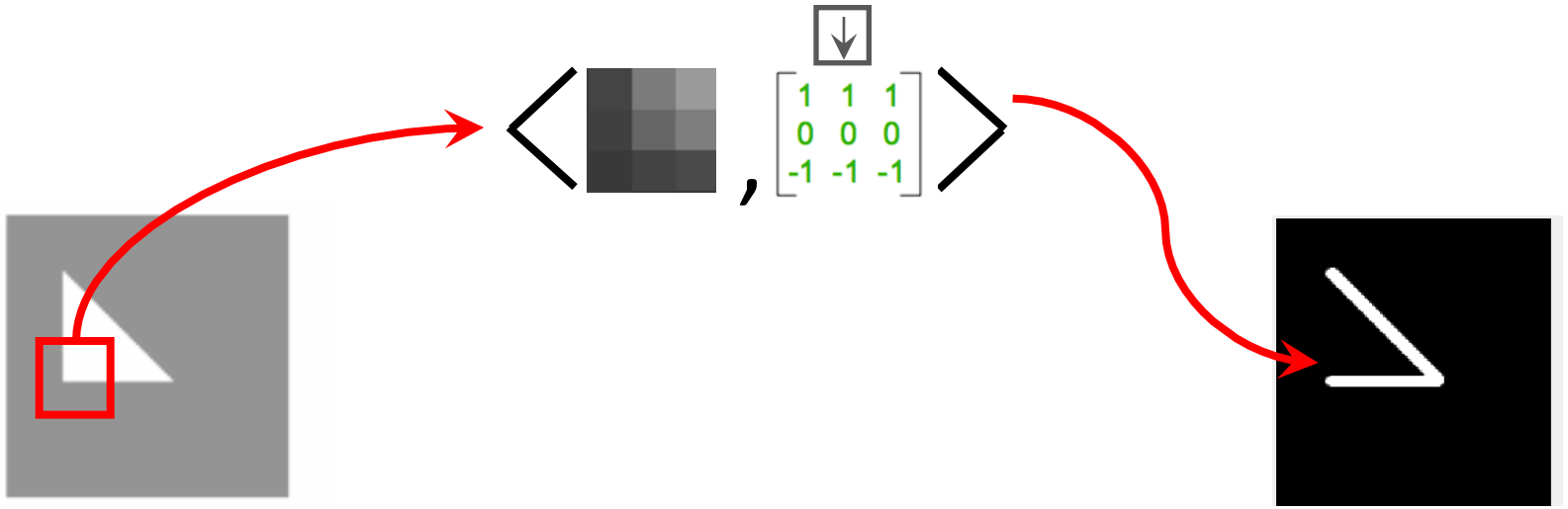
Image taken from [29]

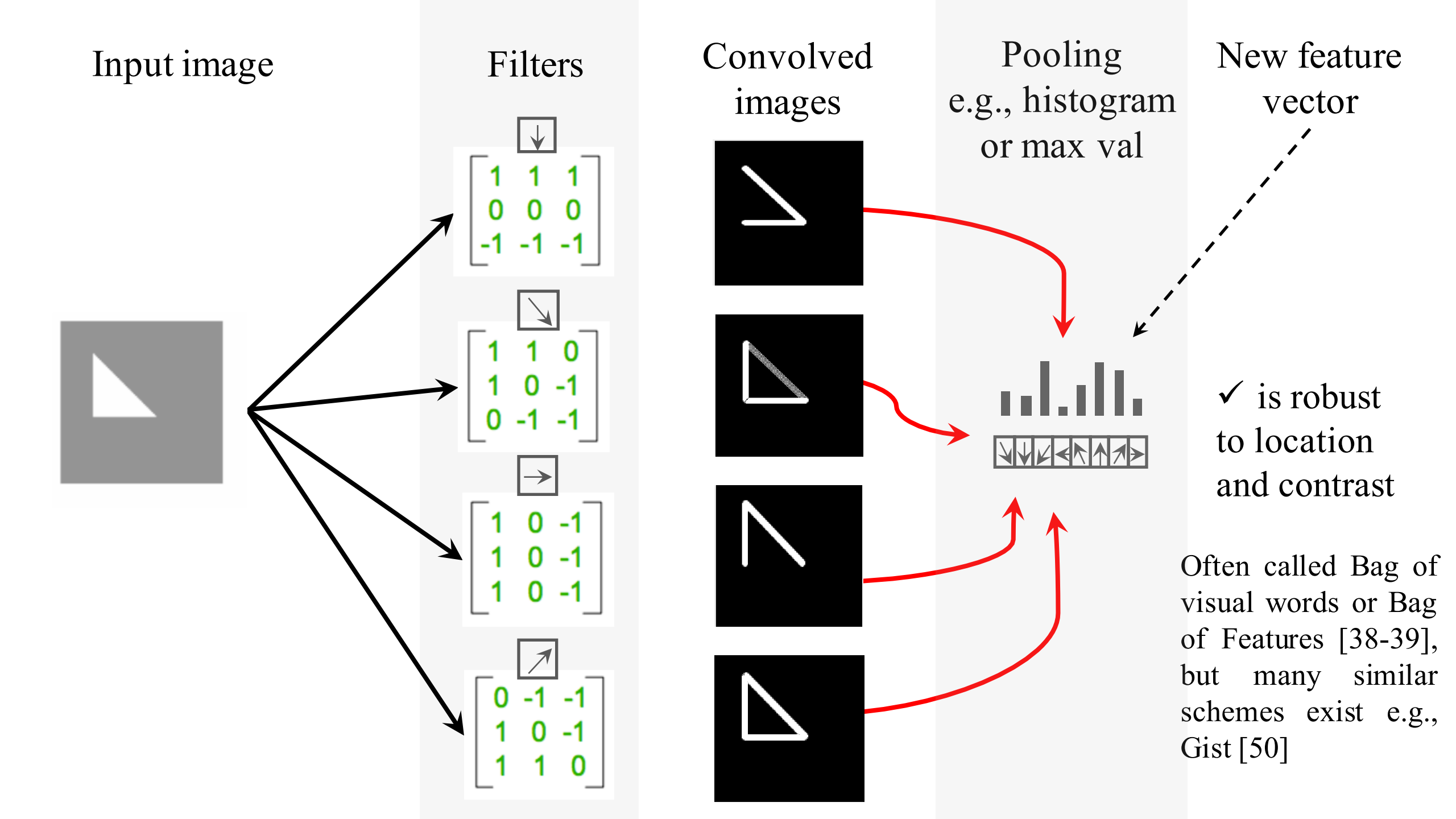
Image histogram features: pixel values or edge orientations?



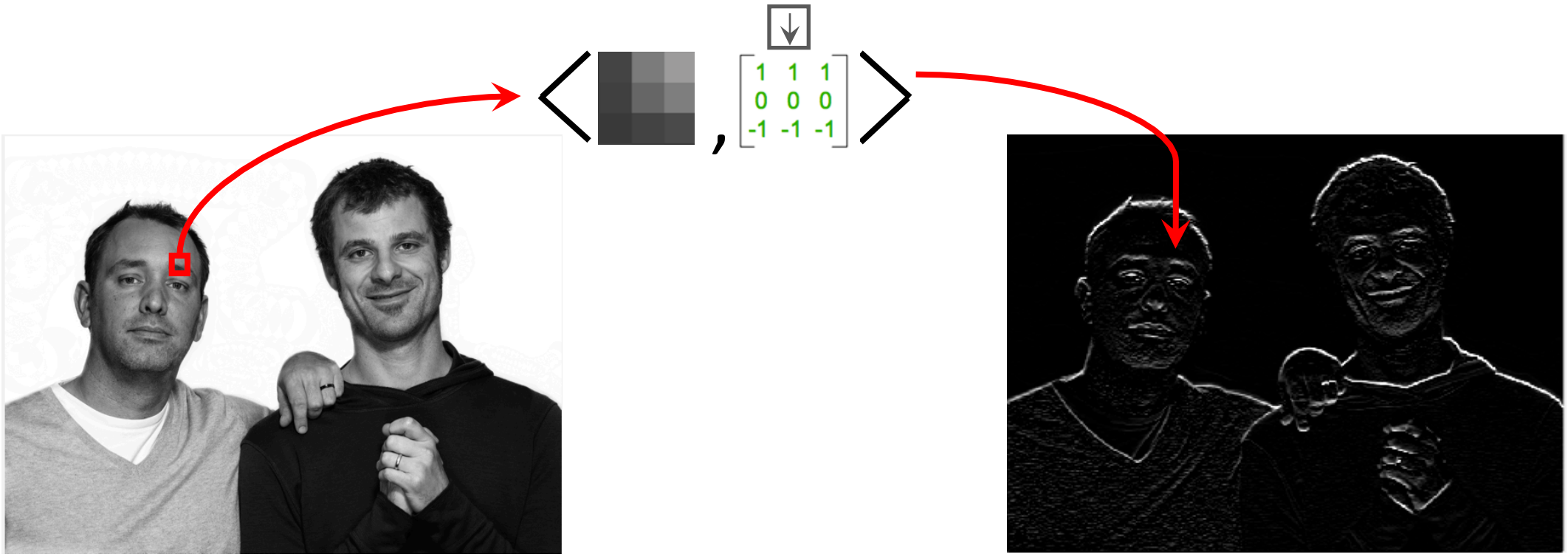


Discrete convolution



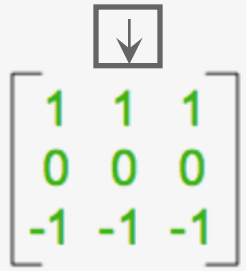

Input image
Filters
Convolved images
Pooling
e.g., histogram or max val
New feature vector
1 1 1
0 0 0
-1 -1 -1
1 1 0
1 0 -1
0 -1 -1
1 0 -1
1 0 -1
1 0 -1
0 -1 -1
1 0 -1
1 1 0
✓ is robust to location and contrast
Often called Bag of visual words or Bag of Features [38-39], but many similar schemes exist e.g., Gist [50]

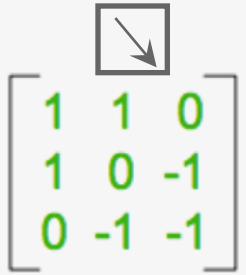
Discrete convoultion

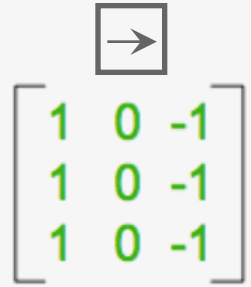


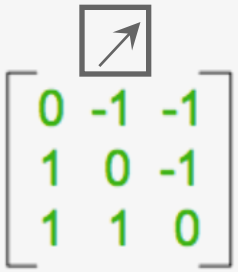
Input image

Filters

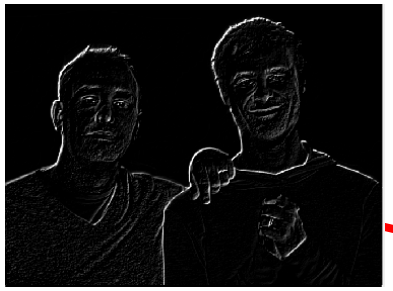








Convolved images



Pooling
e.g., histogram or max val

New feature vector

✓ is robust to contrast but not localization

Often called Bag of visual words or Bag of Features [38-39], but many similar schemes exist e.g., Gist [50]

Input image

Filters

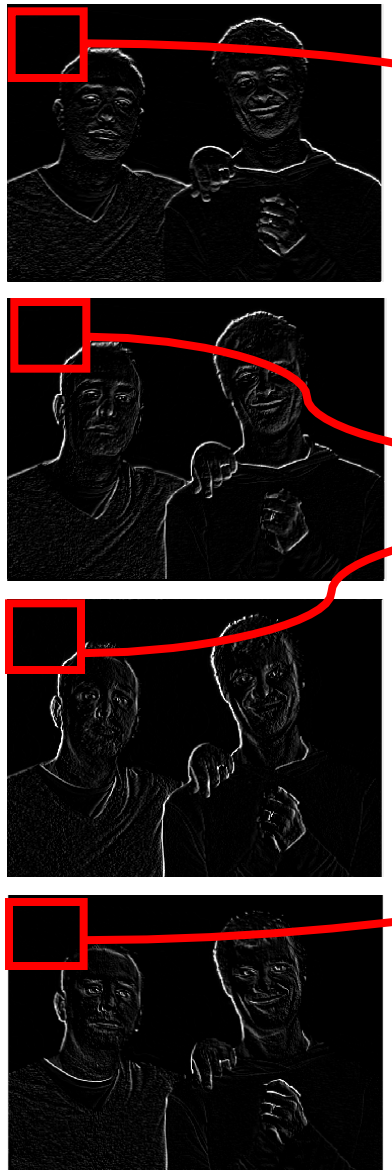
$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

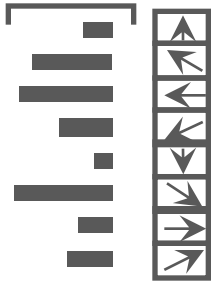
$$\begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$$

Convolved images



Pooling
e.g., histogram or max val

New feature vector



Input image

Filters

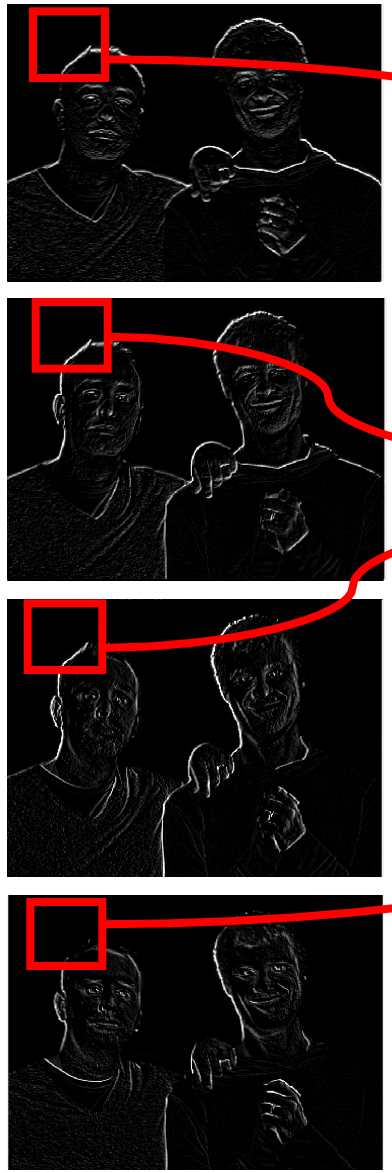
$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

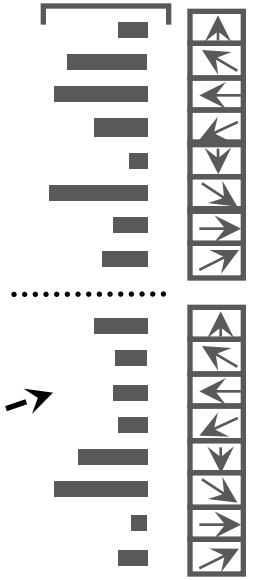
$$\begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$$

Convolved images



Pooling
e.g., histogram or max val

New feature vector



Input image

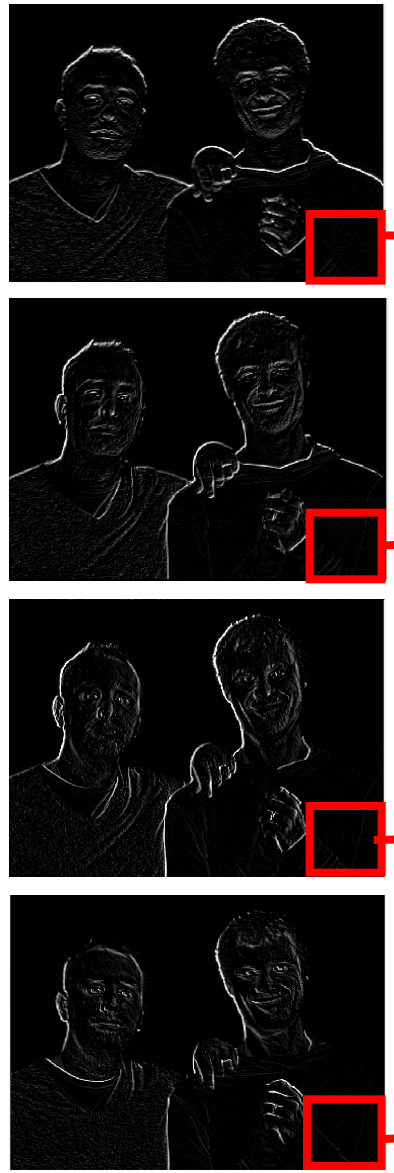
Filters

Convolved images

Pooling e.g., histogram or max val

New feature vector

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$$



Often called Histogram of Oriented Gradients [22], many similar schemes exist (e.g., SIFT [43]) as well as extensions e.g., spatial pyramid matching [40-41]

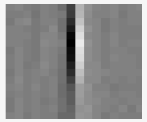
Input image

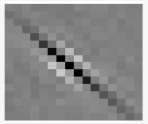
Filters

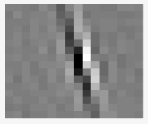
Convolved images

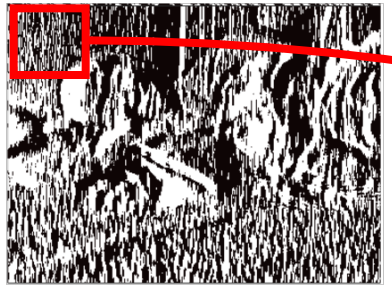
Pooling e.g., histogram or max val

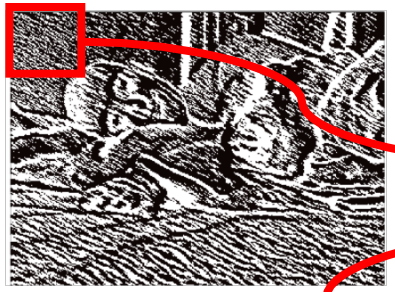
New feature vector

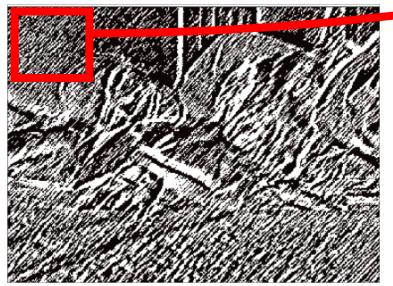


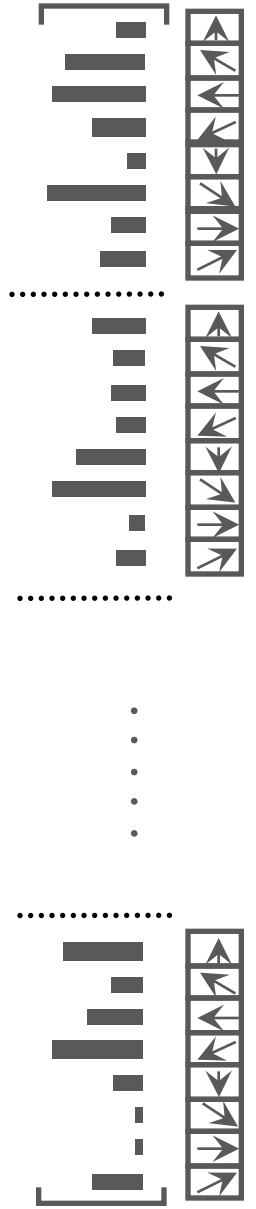












Often called Histogram of Oriented Gradients [22], many similar schemes exist (e.g., SIFT [43]) as well as extensions e.g., spatial pyramid matching [40-41]

Biological inspiration

- Visual cortex consists primarily of simple oriented edge detectors that act on small overlapping patches of visual field

- These detections are then combined (or “pooled”), see e.g., [34 – 36]

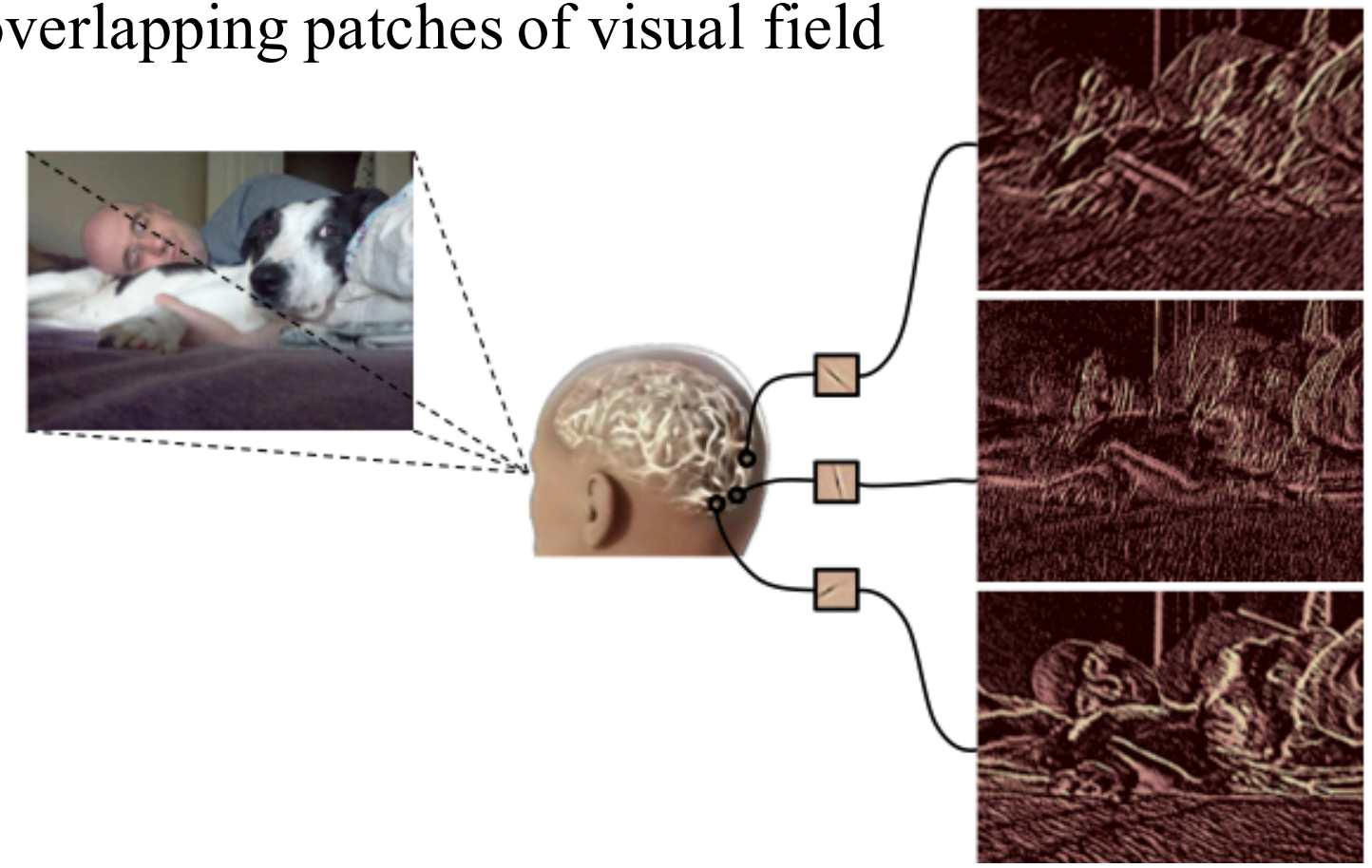
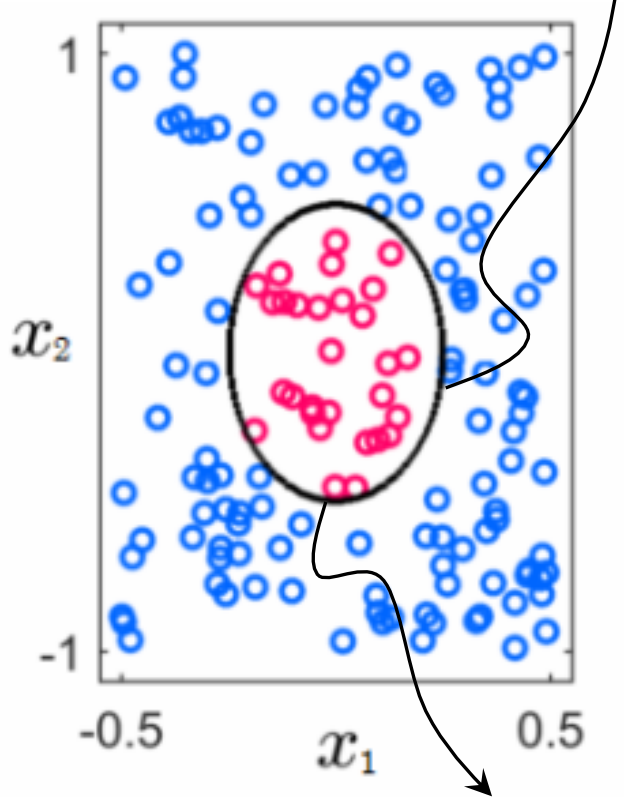


Image taken from [29]

Commonly-used engineered featu

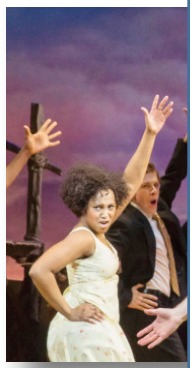
Feature transformations for real dat
aim at ensuring that instances witl
different classes are "dissimilar". T
mations $f_1(\mathbf{x}), ..., f_M(\mathbf{x})$ of the in
algebraically.

$$1 + f_1(\mathbf{x})\, w_1 + f_2(\mathbf{x})\, w_2 = 0$$



$$1 + x_1^2 w_1 + x_2^2 w_2 = 0$$

EVEN THOSE BASED ON REAL PEOPLE--ARE ENTIRELY FICTIONAL. ALL CELEBRITY VOICES ARE IMPERSONATED.....POORLY. THE FOLLOWING PROGRAM CONTAINS COARSE LANGUAGE AND DUE TO ITS CONTENT IT SHOULD NOT BE VIEWED BY ANYONE



Features: summary (re-visited)

What are features?

- **Geometrically:** features are transformations of the input that provoke a good *nonlinear* fit/separation

- **Philosophically:** features are those defining characteristics of the phenomenon underlying a dataset that allow for optimal learning

How do we design good features?

- **By hand (engineered):** translate understanding of a phenomenon into a set of geometric transformations of the input – often requires expertise
  - Requires strong knowledge to produce reasonable results
- **Learn from data (automatic):** use bases (e.g., neural networks) to determine directly from the data
  - Requires large datasets to produce reasonable results (more on this in part II of talk!)