

Lecture Notes for **Machine Learning in Python**



Professor Eric Larson
Dimensionality Reduction and Images

Class Logistics and Agenda

- **Logistics:**
 - Lab grading...
 - **Next Time:** Flipped Module
 - Turn in one per team (HTML), please include team member names from canvas
- **Agenda**
 - Common Feature Extraction Methods for Images
 - Begin Town Hall, if time

Class Overview, by topic

Table Data
Visualization

Numpy, Pandas, Seaborn
Overviews with some in-depth discussion

Dimension
Reduction and
Image Processing

Scikit-learn, Scikit Image,
Intuition only, Some mathematics

Linear and
Logistic
Regression

Numpy, Recreate API for Scikit-learn
Detailed mathematics for simple optimization
intuition for advanced optimization

Neural Networks
and Back Prop.

Numpy
Detailed mathematics for NN operations

Wide and Deep
Networks

Convolutional
Networks

Recurrent
Networks

Keras, Tensorflow
Intuition, Detailed implement.

Ethics in
Language Models

ConceptNet
Case studies

Last time...

$E1$	$E2$
0.85	0.85
0.52	-0.52

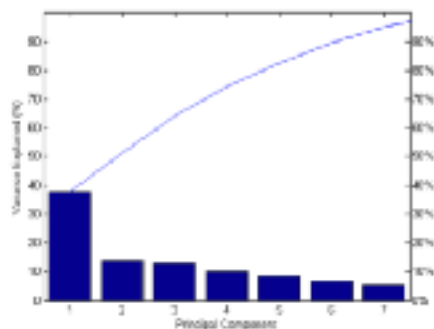
37.1	-6.7
-6.7	43.9

	$A1$	$A2$
1	66	33.6
2	54	26.6
3	69	23.3
4	73	28.1
5	61	43.1
6	62	25.6

	$A1$	$A2$
1	1.83	3.55
2	-10.1	-3.45
3	4.83	-6.75
4	8.83	-1.95
5	-3.17	13.05
6	-2.17	-4.45

zero mean

$$r_q = \frac{\sum_{j=1}^q \lambda_j}{\sum_{j=1}^p \lambda_j}$$

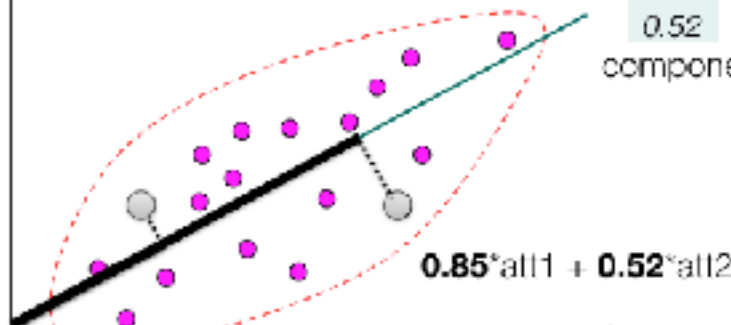


attribute₂

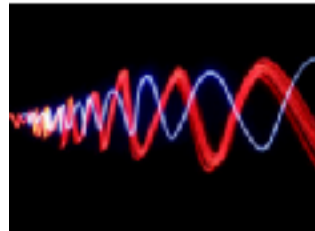
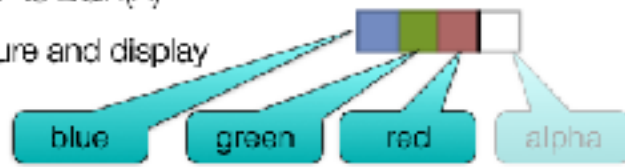
now I can represent data as if it only had one attribute!!
vector weights are also meaningful...

$E1$
0.85
0.52

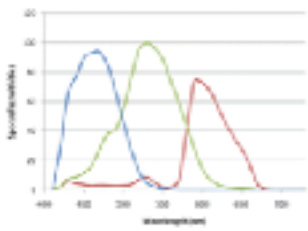
component



- an image can be represented in many ways
- most common format is a matrix of pixels
 - each "pixel" is BGR(A)
- used for capture and display



- sensor
- sensor
- sensor



Review: Image Representation, Features

Problem: need to represent image as table data

- need a compact representation

1	4	2	5	6	9
1	4	2	5	5	9
1	4	2	8	8	7
3	4	3	9	9	8
1	0	2	7	7	9
1	4	3	9	8	6
2	4	2	8	7	9

Review: Image Representation, Features

Problem: need to represent image as table data

- need a compact representation

Solution: row concatenation (also, vectorizing)

Row 1	1	4	2	5	6	9	1	4	2	5	5	9	1	4	2	8	8	7	3
Row 2	1	4	2	8	8	7	3	4	3	9	9	8	1	4	2	5	5	9	1
...																			
Row N	9	4	6	8	8	7	4	1	3	9	2	1	1	5	2	1	5	9	1

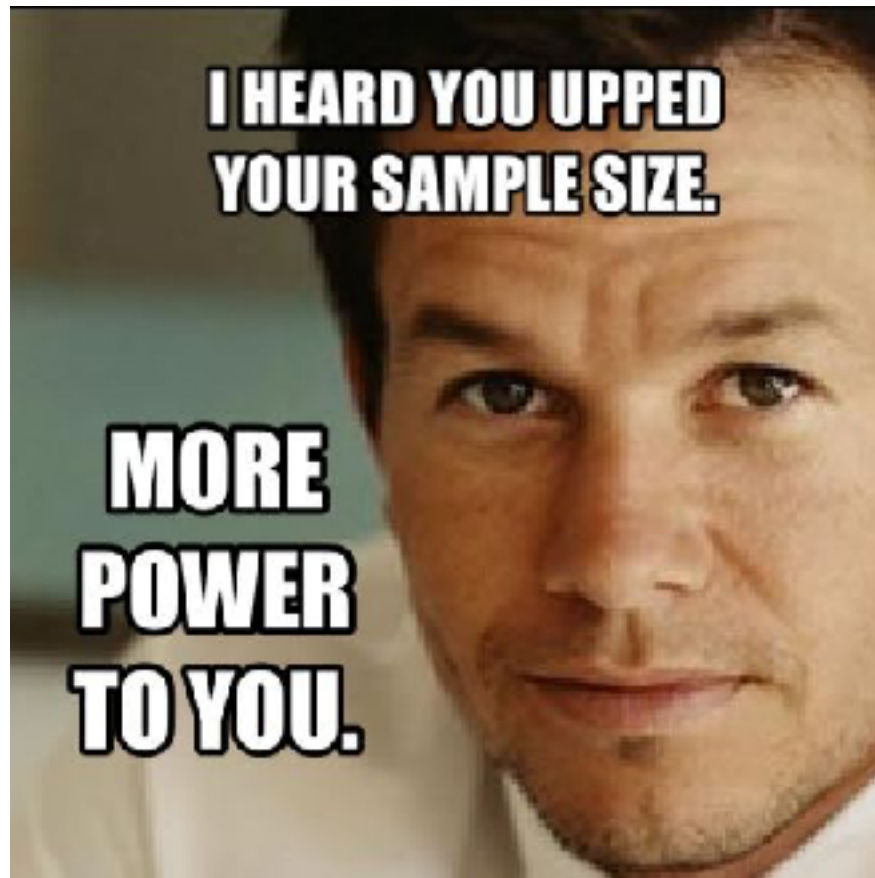
“if needed” Demo

Images Representation
in PCA and
Randomized PCA



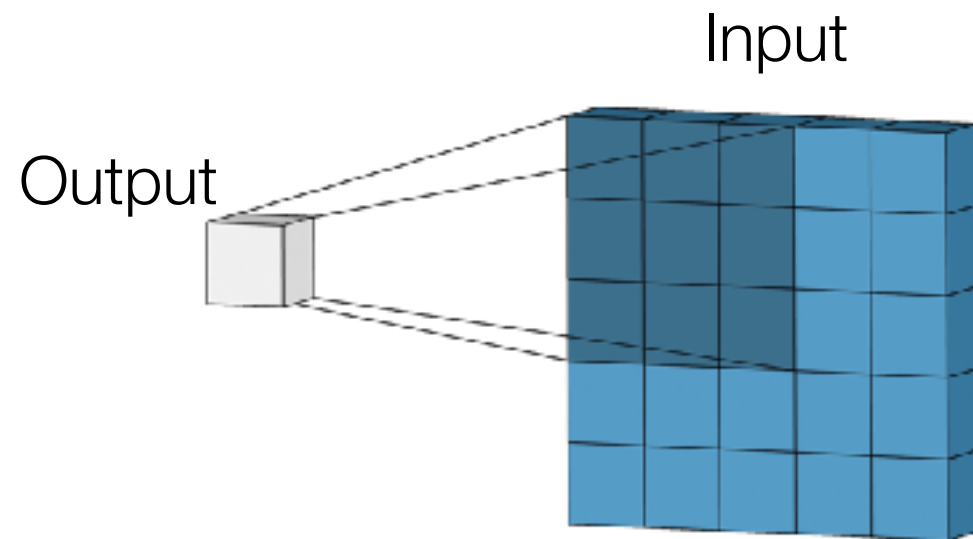
04.Dimension Reduction and Images.ipynb

Features of Images



Extracting Features: Convolution

- For images:
 - kernel (matrix of values)
 - slide kernel across image, pixel by pixel
 - multiply and accumulate



This Example:

3x3 Kernel (dark)

Ignoring edges of input

Input Image is 5x5

Output is then 3x3

Convolution

$$\sum \left(\mathbf{I} \left[i \pm \frac{r}{2}, j \pm \frac{c}{2} \right] \odot \mathbf{k} \right) = \mathbf{O}[i, j] \quad \text{output image at pixel } i, j$$

input image slice centered in i, j
with range $r \times c$

kernel of size, $r \times c$
usually $r=c$

0	0	0	0	0	0	0	0	0
0	1	2	3	4	12	9	8	0
0	5	2	3	4	12	9	8	0
0	5	2	1	4	10	9	8	0
0	7	2	1	4	12	7	8	0
0	7	2	1	4	14	9	8	0
0	5	2	3	4	12	7	8	0
0	5	2	1	4	12	9	8	0
0	0	0	0	0	0	0	0	0

input image, \mathbf{I}

0	0	0
2	3	4
2	3	4

1	2	1
2	4	2
1	2	1

kernel
filter, \mathbf{k}
3x3

20	21	36
...
...
...
...
...
...
...

output image, \mathbf{O}

Convolution Examples



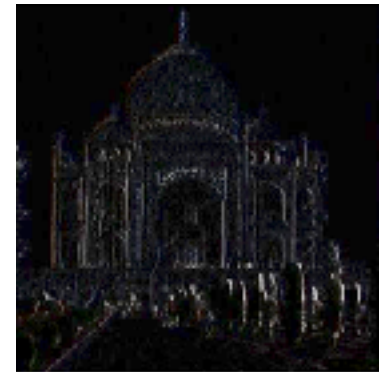
Blur

1	1	1
1	1	1
1	1	1



Vertical Edges

-1	0	1
-1	0	1
-1	0	1



Sharpen

0	-1	0
-1	5	-1
0	-1	0



Self test:

What does this do?

0	0	0
1	0	0
0	0	0

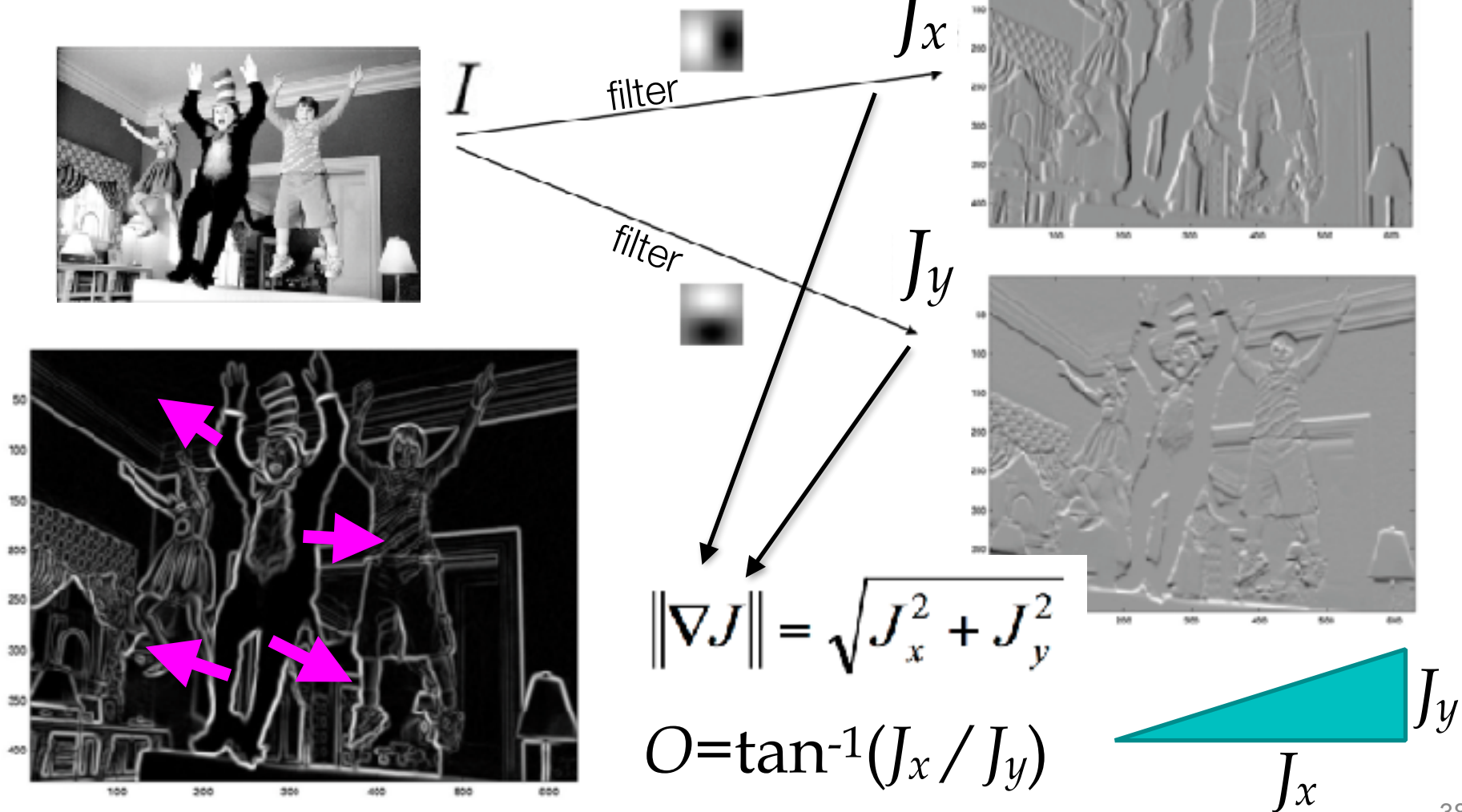
A. move left one pixel

B. move right one pixel

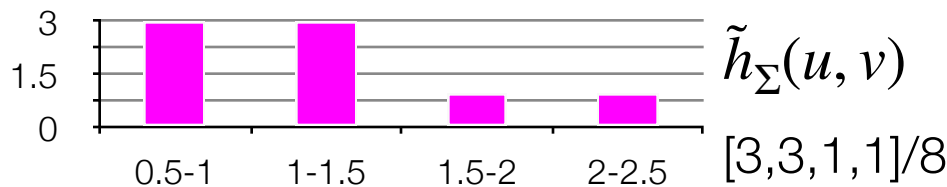
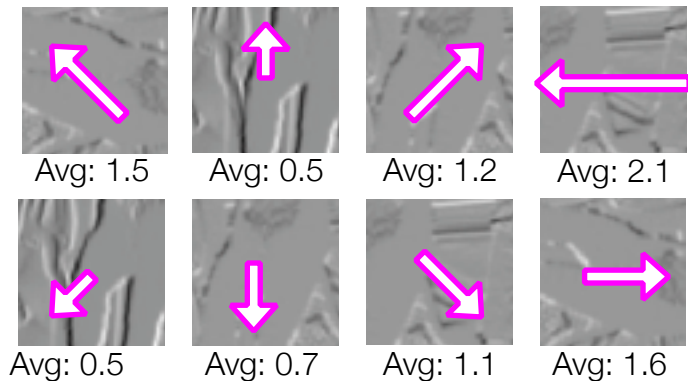
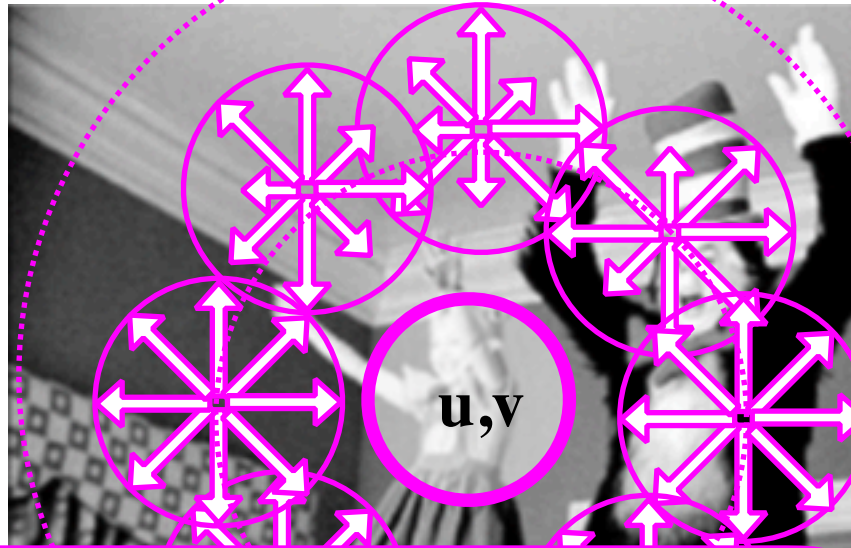
C. blur

Common operations

- the gradient (2D derivative)

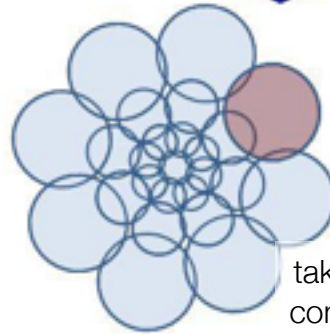
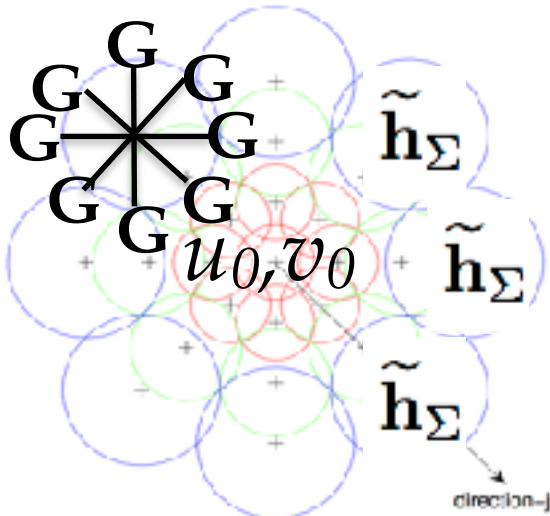


DAISY: same features, regardless of orientation



1. Select u, v pixel location in image and radius
2. Take histogram of average gradient magnitudes in circle for each orientation $\tilde{h}_{\Sigma}(u, v)$
3. Select circles in a ring, R
4. For each circle on the ring, take another histogram $\tilde{h}_{\Sigma}(\mathbf{I}_O(u, v, R_1))$
5. Repeat for more rings
6. Save all histograms as “descriptors”
 $[\tilde{h}_{\Sigma}(\cdot), \tilde{h}_{\Sigma}(\mathbf{I}_1(\cdot, R_1)), \tilde{h}_{\Sigma}(\mathbf{I}_2(\cdot, R_1)) \dots]$
7. Can concatenate descriptors as “feature” vector at that pixel location

Summary DAISY



take histogram of
convolved images
at points u, v

gradient
orientations
(separate
convs)

convolve with
Gaussian kernel

Concatenate Histograms

$$\mathcal{D}(u_0, v_0) =$$

$$\begin{bmatrix} \tilde{h}_{\Sigma_1}^\top(u_0, v_0), \\ \tilde{h}_{\Sigma_1}^\top(\mathbf{l}_1(u_0, v_0, R_1)), \dots, \tilde{h}_{\Sigma_1}^\top(\mathbf{l}_T(u_0, v_0, R_1)), \\ \tilde{h}_{\Sigma_2}^\top(\mathbf{l}_1(u_0, v_0, R_2)), \dots, \tilde{h}_{\Sigma_2}^\top(\mathbf{l}_T(u_0, v_0, R_2)), \end{bmatrix}$$

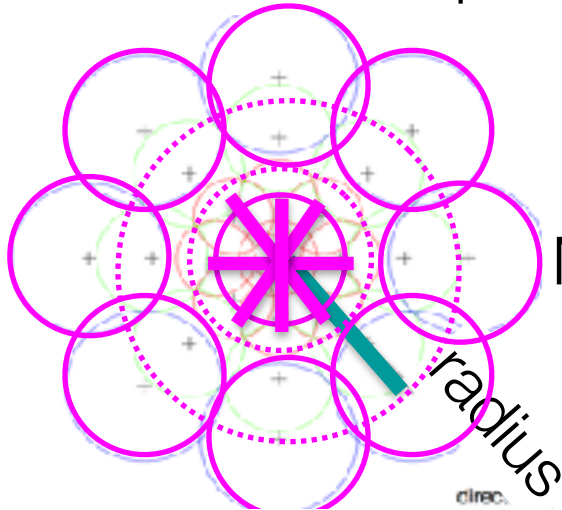
take **normalized** histogram of magnitudes

$$\tilde{h}_\Sigma(u, v) = \left\| [\mathbf{G}_1^\Sigma(u, v), \dots, \mathbf{G}_H^\Sigma(u, v)]^\top \right\|$$

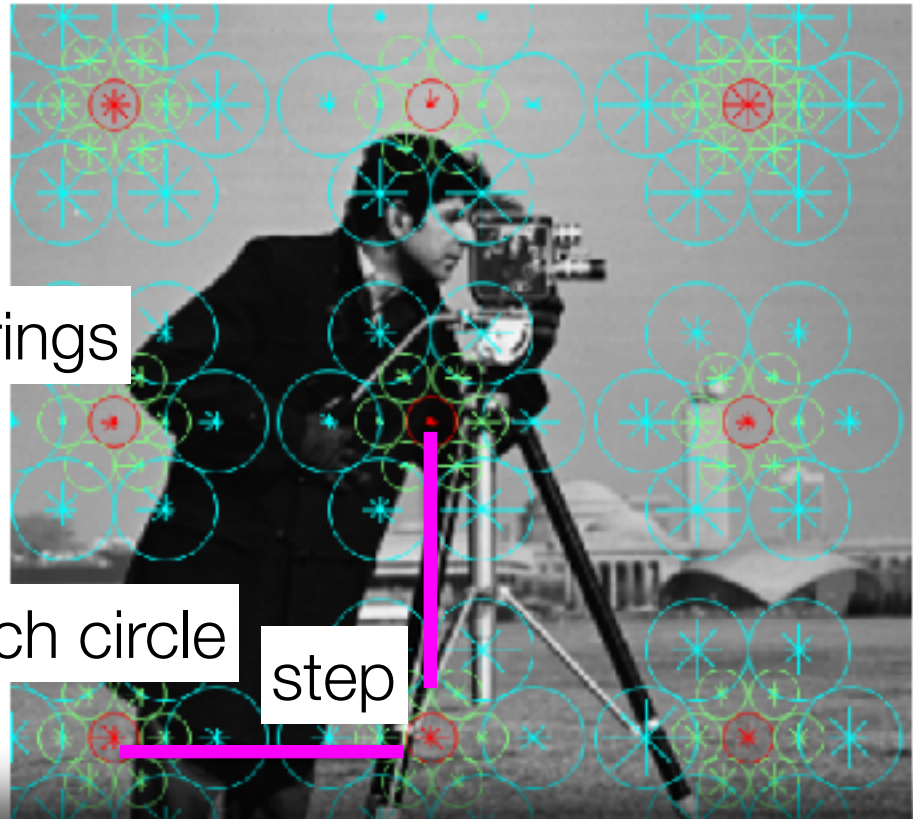
Tola et al. "Daisy: An efficient dense descriptor applied to wide-baseline stereo." Pattern Analysis and Machine Intelligence, IEEE

Free Parameters in DAISY

Num circles/per ring



Num rings



Num orientations within each circle

step

Num of bins in histogram

```
daisy(img, step=180, radius=58, rings=2, histograms=6,  
      orientations=8, visualize=True)
```

Params

step, radius, num rings, num histograms per ring,
orientations, *bins per histogram*

Gradients

DAISY

(if time) Gabor Filter Banks

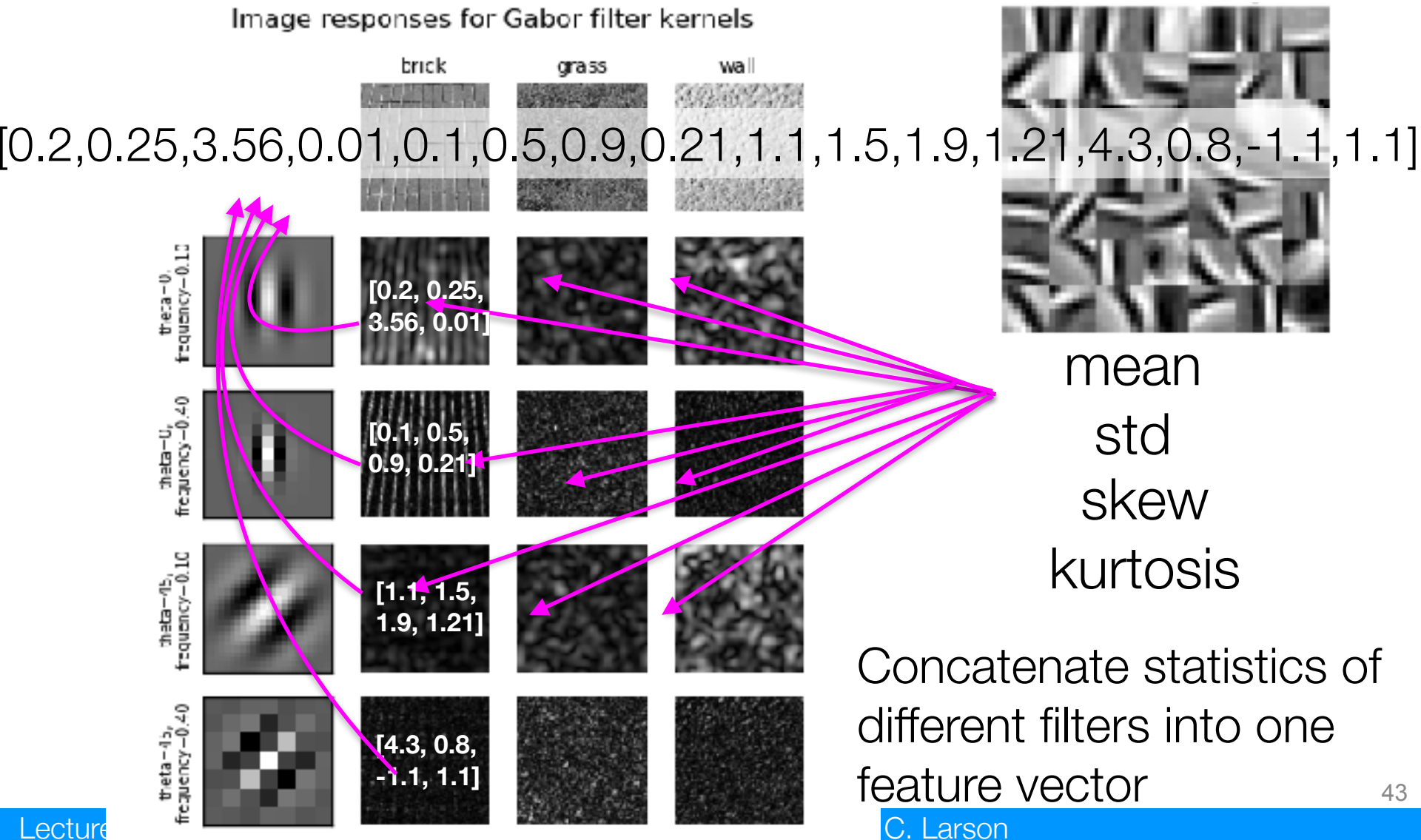


Other Tutorials:

http://scikit-image.org/docs/dev/auto_examples/

Common operations: Gabor filter Banks (if time)

- common used for texture classification



Matching versus Bag of Features

- Not a difference of vectors, but a percentage of matching points



- SURF, ORB, SIFT, DAISY

Feature Matching

Matching test image to source dataset

1. Choose src image from dataset
2. Take keypoints of src image
3. Take keypoints of test image
4. For each kp in src:
 1. Match with closest kp in test
 2. *How to define match?*
5. Count number of matches between images
6. Determine if src and test are similar based on number of matches
7. Repeat for new src image in dataset
8. Once all images measured, choose best match as the target for the test image



Scikit-image Implementation

match_descriptors

```
skimage.feature.match_descriptors(descriptors1, descriptors2, metric=None, p=2,  
max_distance=inf, cross_check=True, max_ratio=1.0)
```

[\[source\]](#)

Brute-force matching of descriptors.

For each descriptor in the first set this matcher finds the closest descriptor in the second set (and vice-versa in the case of enabled cross-checking).

Town Hall for Lab 2, Images

- **Quiz is live:** Image Processing!
- **Next Time:** Logistic Regression



Supplemental Slides

Peruse these at your own leisure!

These slides might assist you as additional visual aides

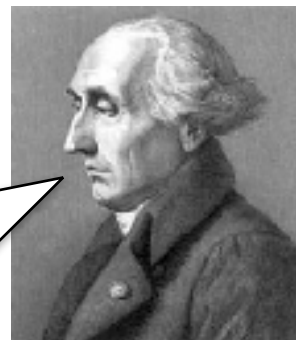
Slides courtesy of Tan, Steinbach, Kumar
Introduction to Data Mining

Dimensionality Reduction: LDA

- PCA tell us variance explained by the data in different directions, but it ignores class labels
- Is there a way to find “components” that will help with **discriminate** between the classes?

$$\arg \max_{comp.} \frac{\sum \text{differences between classes}}{\sum \text{variance within classes}}$$

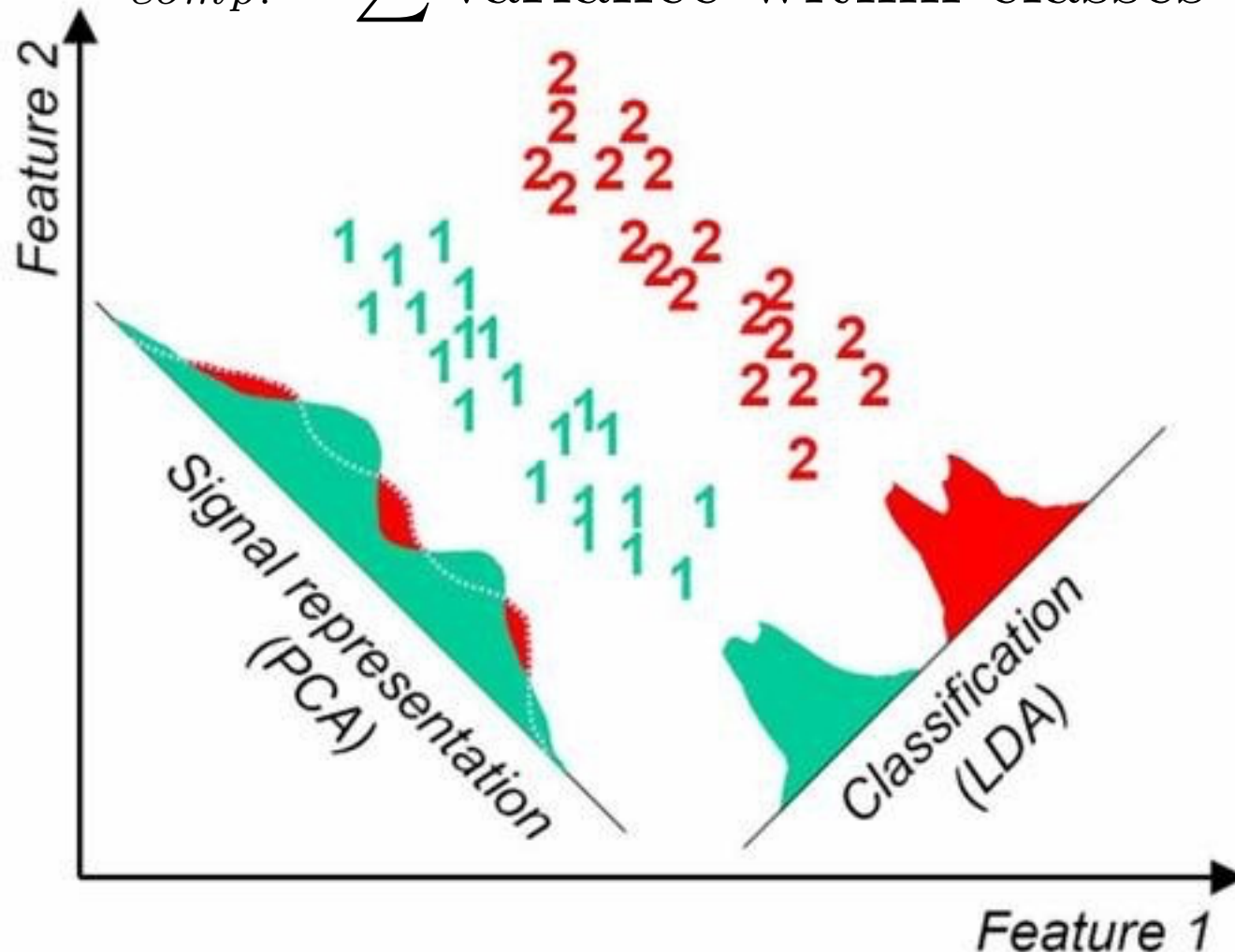
- ❑ called Fisher’s discriminant
- ❑ ...but we need to solve this using using *Lagrange multipliers* and gradient-based optimization
- ❑ which we haven’t covered yet



I invented Lagrange multipliers... and ...*nothing* impresses me...

Dimensionality Reduction: LDA versus QDA

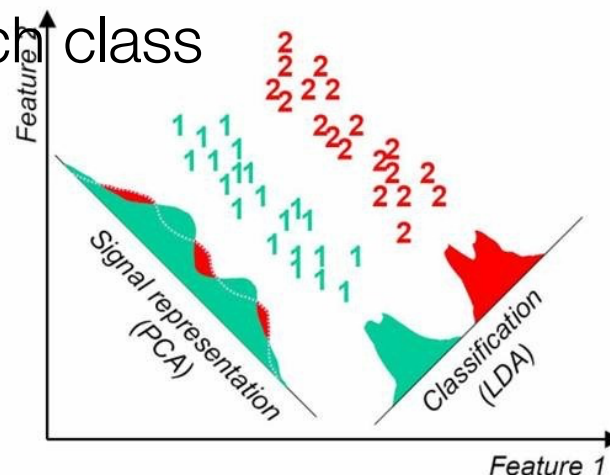
$$\arg \max_{comp.} \frac{\sum \text{differences between classes}}{\sum \text{variance within classes}}$$



Dimensionality Reduction: LDA versus QDA

$$\arg \max_{comp.} \frac{\sum \text{differences between classes}}{\sum \text{variance within classes}}$$

- “*differences between classes*” is calculated by trying to separate the **mean value** of each **feature** in each **class**
- Linear discriminant analysis:
 - assume the covariance in each class is the same
- Quadrature discriminant analysis:
 - estimate the covariance for each class



Self Test ML2b.2

LDA only allows as many components as there are unique classes in a dataset.

- A. True
- B. False

❑ Need more help with the PCA algorithm (and python)?

❑ check out Sebastian Raschka's notebooks:

http://nbviewer.ipython.org/github/rasbt/pattern_classification/blob/master/dimensionality_reduction/projection/principal_component_analysis.ipynb