Today: Outline

- **Unsupervised Learning: Mixtures of Gaussians**
- **Unsupervised Learning: Anomaly Detection**
- **Semi-supervised Learning**

- Announcements:
  - Pre-lecture Material 4, *due: Jun 10*
  - Exam Jun 22 in class

    (and ~12 hrs before for remote only students)

https://distill.pub/

Distill ABOUT PRIZE SUBMIT

March 4, 2021

PEER-REVIEWED

Multimodal Neurons in Artificial Neural Networks

Gabriel Goh, Nick Cammarata, Chelsea Voss, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah

We report the existence of multimodal neurons in artificial neural networks, similar to those found in the human brain.

Nov. 17, 2020

PEER-REVIEWED

Understanding RL Vision

Jacob Hilton, Nick Cammarata, Shan Carter, Gabriel Goh, and Chris Olah

With diverse environments, we can analyze, diagnose and edit deep reinforcement learning models using attribution.

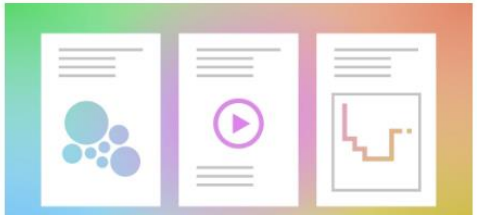
Sept. 11, 2020

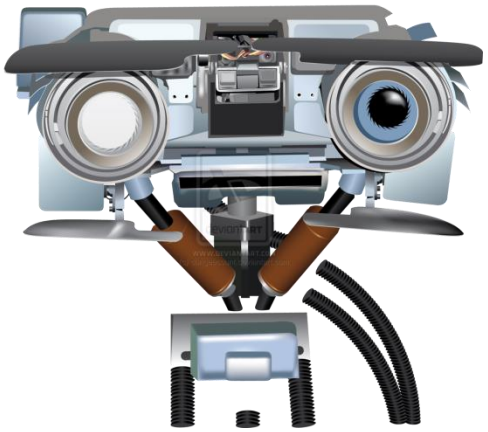
COMMENTARY

Communicating with Interactive Articles

Fred Hohman, Matthew Conlen, Jeffrey Heer, and Duen Horng (Polo) Chau

Examining the design of interactive articles by



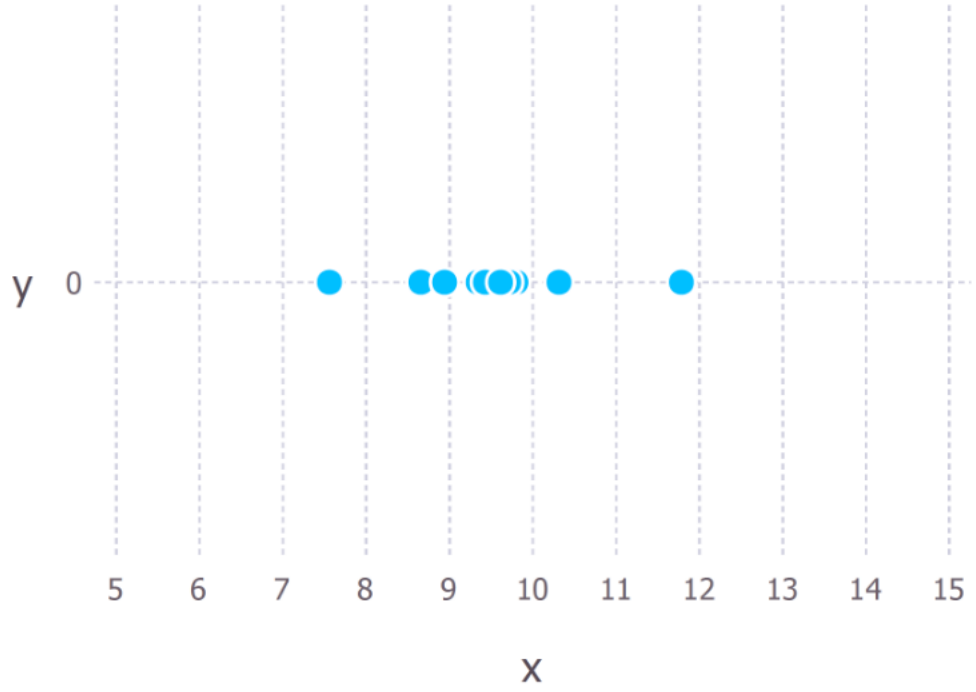


Unsupervised Learning

Mixtures of Gaussians

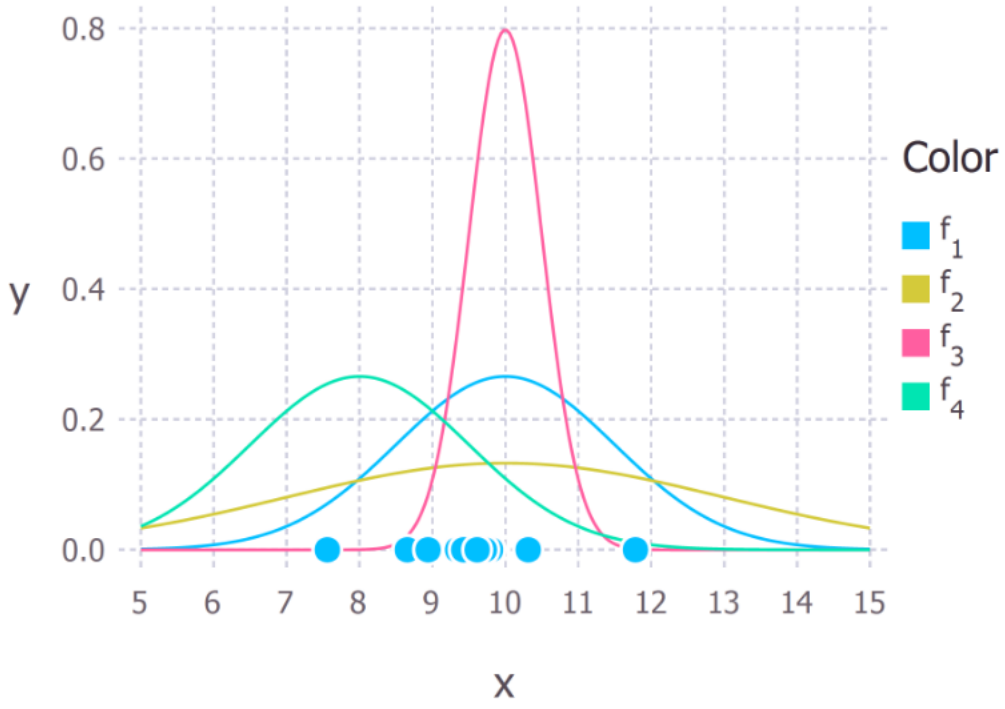
Observed Data from a Single Gaussian

- Ten observed data points from some process



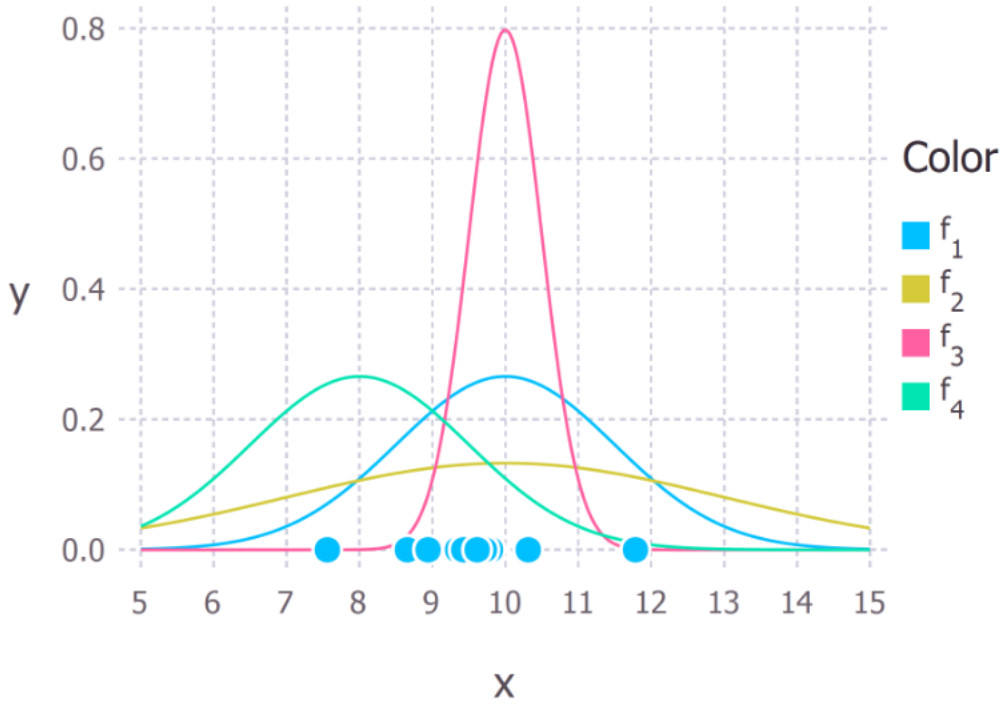
Learning the Model

- We want to know *which curve was most likely responsible for creating the data points that we observed?*



Maximum Likelihood

- Maximum likelihood estimation is a method that will find the values of μ and σ that result in the curve that best fits the data.



Calculating Maximum Likelihood Estimates

- What we want to calculate is the total probability of observing all of the data, *i.e.* the joint probability distribution of all observed data points.

- To do this we would need to calculate some conditional probabilities, which can get very difficult.

- So it is here that we'll make our first assumption. *The assumption is that each data point is generated independently of the others*.

Calculating Maximum Likelihood Estimates

The probability density of observing a single data point x, that is generated from a Gaussian distribution is given by:

$$P(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

In our example the total (joint) probability density of observing the three data points is given by:

$$P(9, 9.5, 11; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(9-\mu)^2}{2\sigma^2}\right) \times \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(9.5-\mu)^2}{2\sigma^2}\right) \times \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(11-\mu)^2}{2\sigma^2}\right)$$

Calculating Maximum Likelihood Estimates

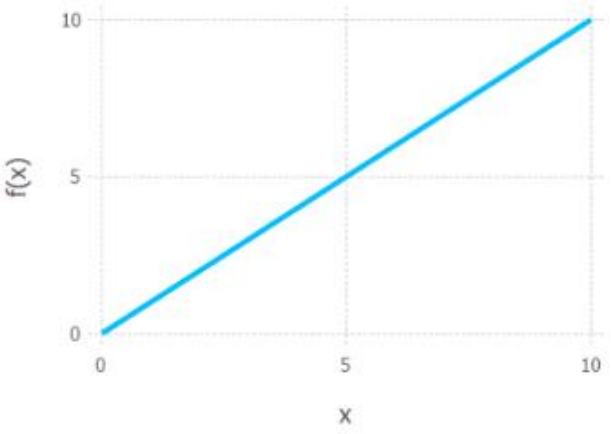
- We need to find the values of μ and σ that results in giving the maximum value of the above expression.

- The above expression for the total probability is difficult to differentiate.

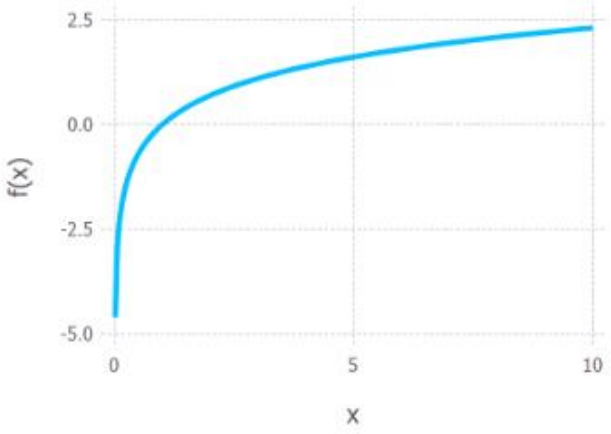
- It is almost always simplified by taking the natural logarithm of the expression.

Log Likelihood

- This is absolutely fine because the natural logarithm is a monotonically increasing function.



(a) $f(x) = x$



(b) $f(x) = \ln(x)$

Log Likelihood

Taking logs of the original expression gives us:

$$\ln(P(x;\mu,\sigma)) = \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{(9-\mu)^2}{2\sigma^2} + \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{(9.5-\mu)^2}{2\sigma^2} + \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{(11-\mu)^2}{2\sigma^2}$$

This expression can be simplified again using the laws of logarithms to obtain:

$$\ln(P(x;\mu,\sigma)) = -3\ln(\sigma) - \frac{3}{2}\ln(2\pi) - \frac{1}{2\sigma^2}\left[(9-\mu)^2 + (9.5-\mu)^2 + (11-\mu)^2\right]$$

Computing μ_{ML}

This expression can be differentiated to find the maximum. In this example we'll find the MLE of the mean, μ. To do this we take the partial derivative of the function with respect to μ, giving

$$\frac{\partial \ln(P(x;\mu,\sigma))}{\partial \mu} = \frac{1}{\sigma^2}\left[9 + 9.5 + 11 - 3\mu\right].$$

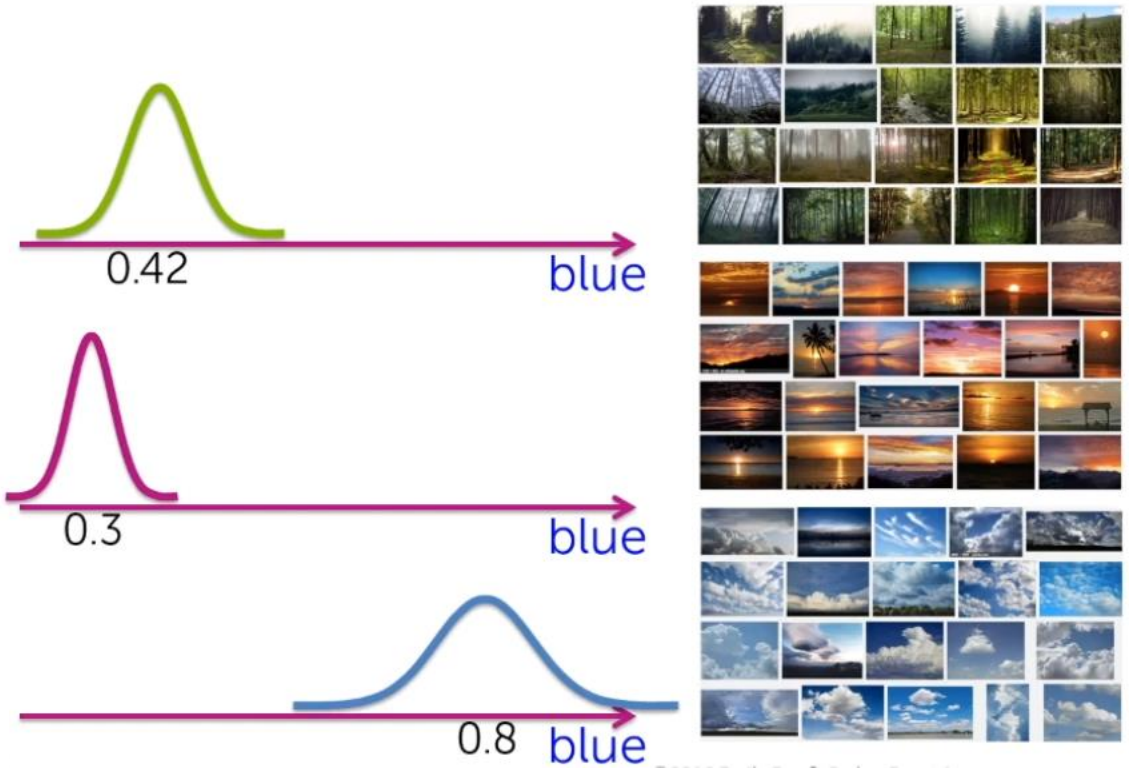
Finally, setting the left hand side of the equation to zero and then rearranging for μ gives:

$$\mu = \frac{9 + 9.5 + 11}{3} = 9.833$$

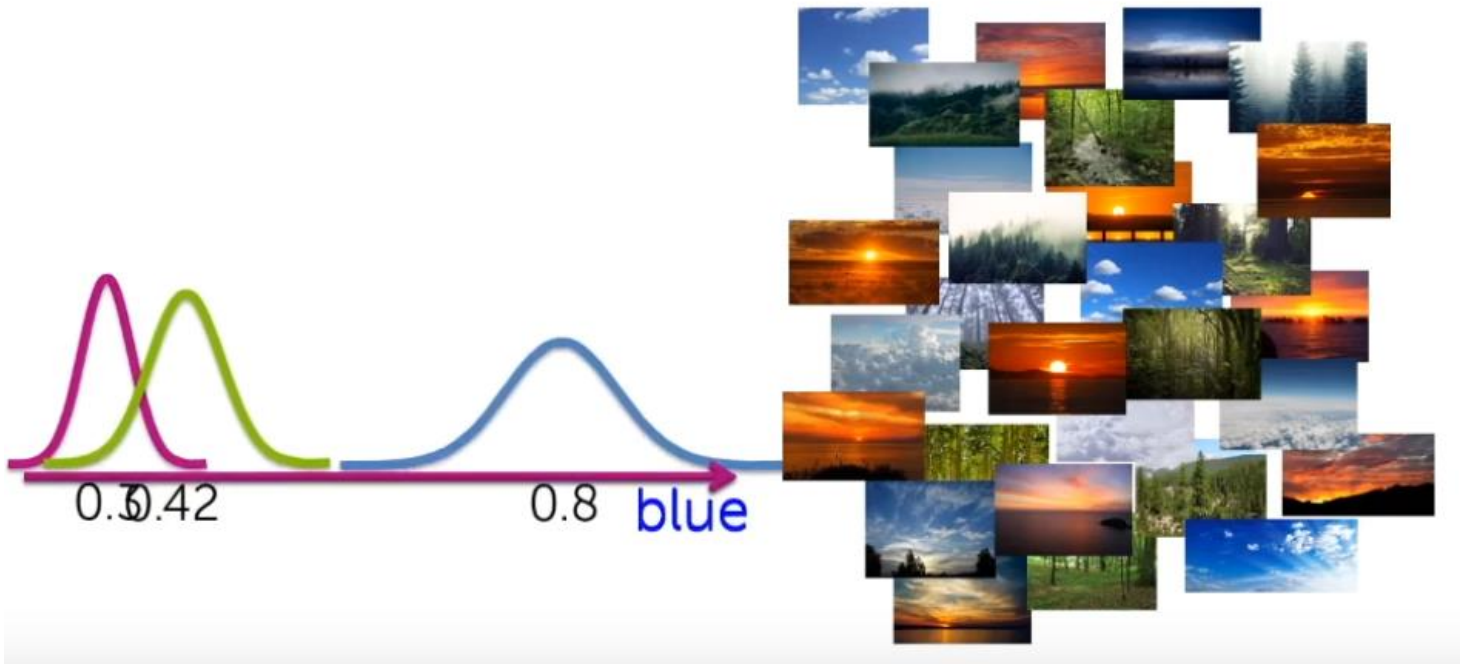
μ_{ML}

Do the same for σ

Mixtures of Gaussians

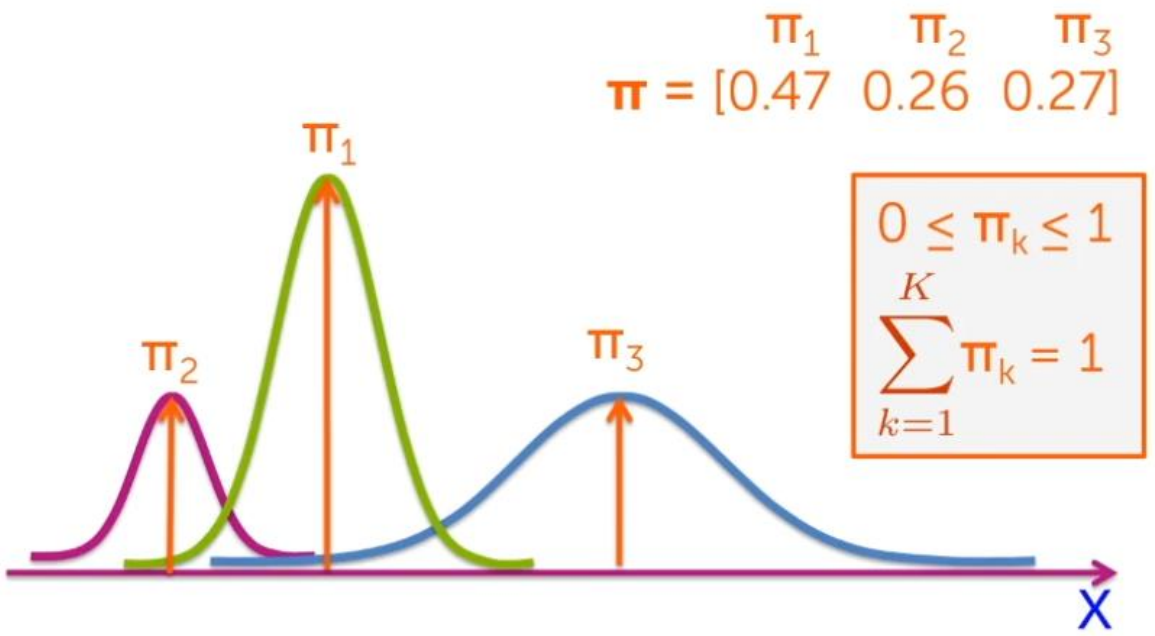


Mixtures of Gaussians



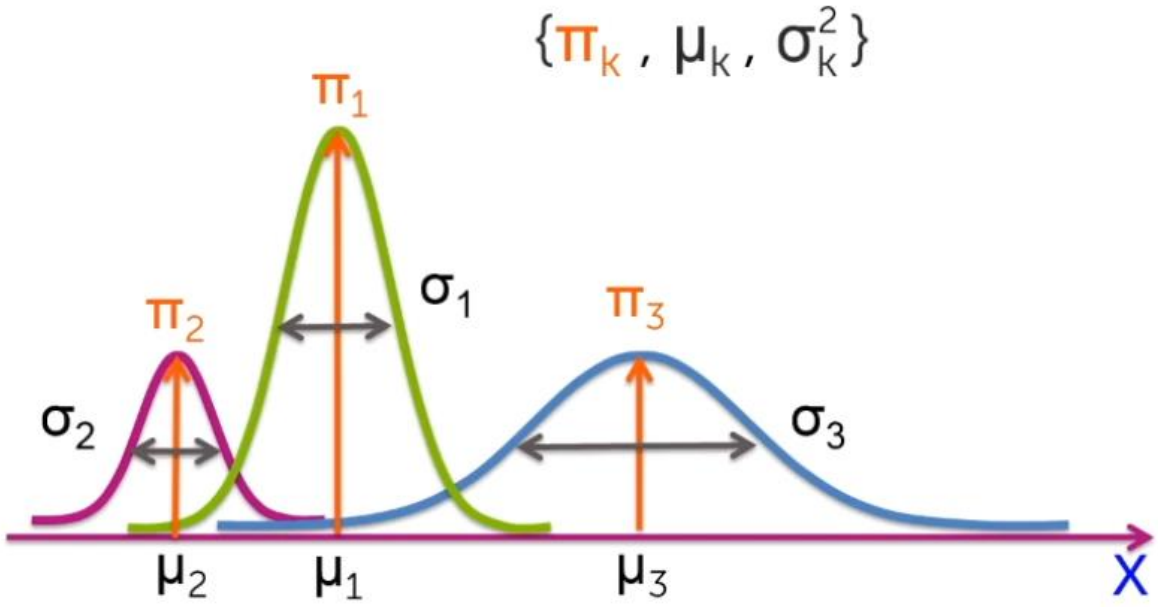
Mixtures of Gaussians

- Associate a weight π_k with each Gaussian Component: "The mixing coefficients"

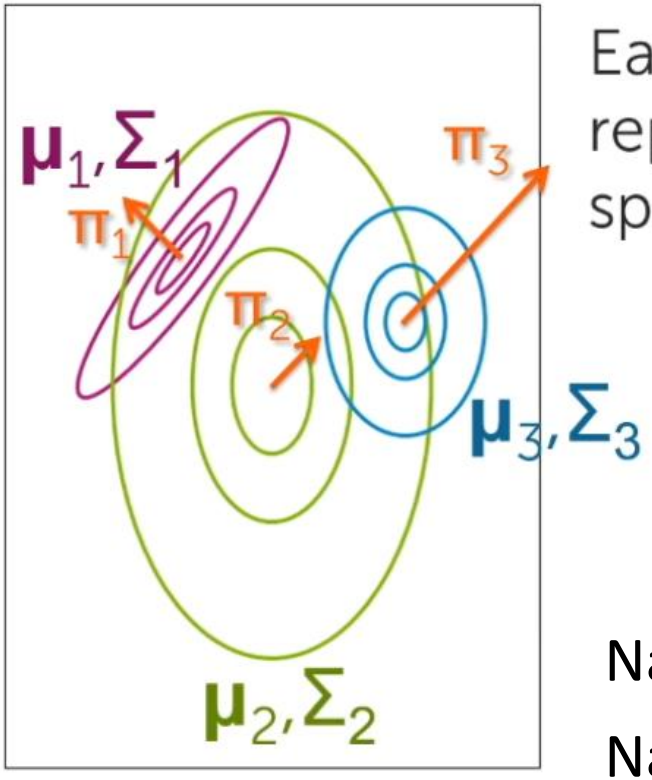


Mixtures of Gaussians

- Location and spread for the distributions comprising the Gaussians



Higher Dimensions



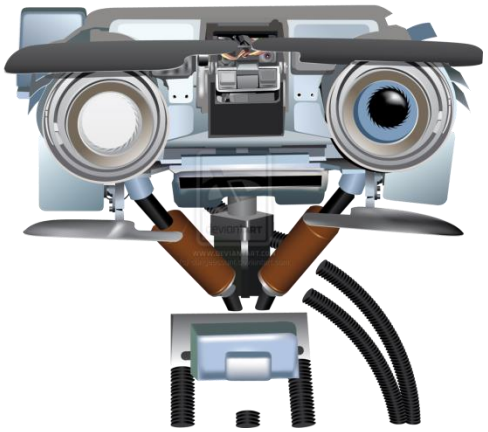
Each mixture component represents a unique cluster specified by:

$$\{\pi_k, \mu_k, \Sigma_k\}$$

Naturally generated clusters!

Naturally a generative model!

vs. discriminative models



Unsupervised Learning

Anomaly Detection

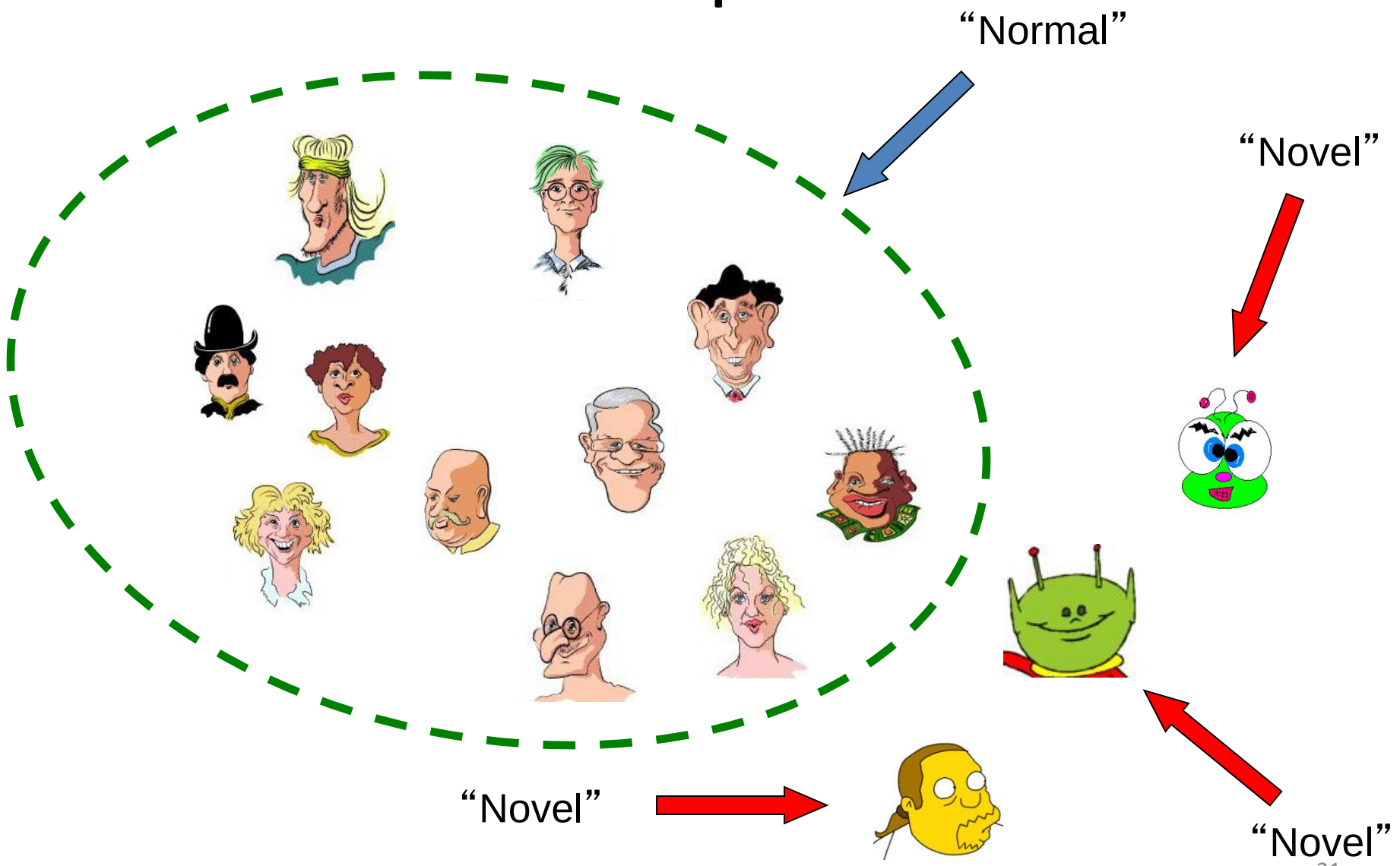
What is an anomaly?

Anomaly Detection is

- An unsupervised learning problem (data unlabeled)

- About the identification of new or unknown data or signal that a learning system is not aware of during training

Example 1



So what seems to be the problem?

It's a 2-Class problem.
"Normal" vs. "Novel"

So what seems to be the problem?

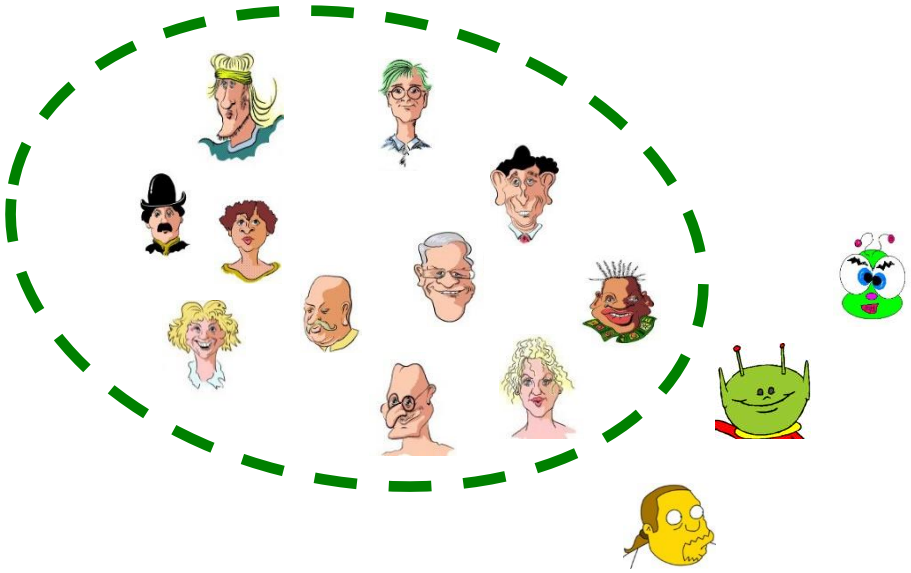
It's a C-Class problem.

"Normal" system level

Wrong!

The Problem is

That “All positive examples are alike but each negative example is negative in its own way”.



One-class recognition

- Suppose we want to build a classifier that recognizes anomalous activities in an airport

- How can we collect a training data?
  - We easily assemble videos of normal airport activities like walking, checking in, etc., as positive examples.

- What about negative examples ?
  - The negative examples are… all other activities!!

- So the negative examples come from an unknown # of negative classes.

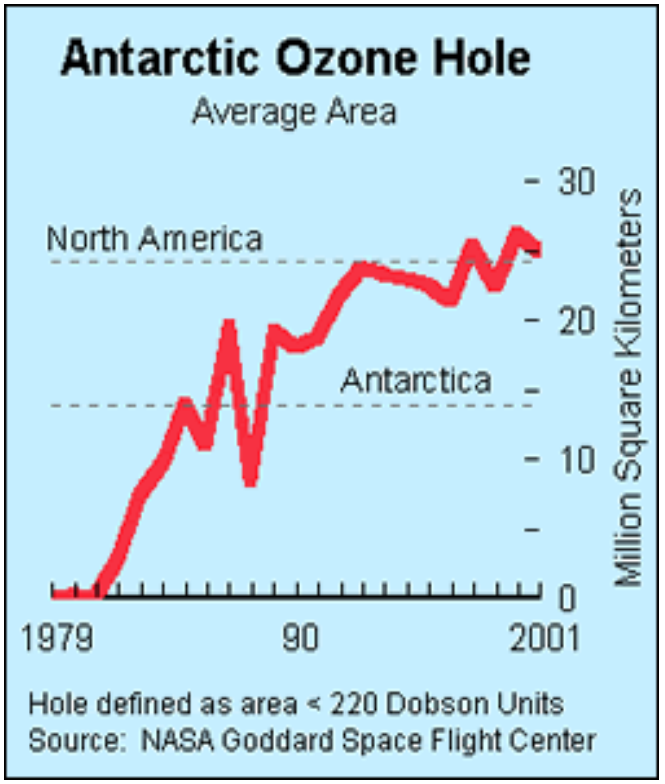
Importance of Anomaly Detection

Ozone Depletion History

- In 1985 three researchers (Farman, Gardinar and Shanklin) were puzzled by data gathered by the British Antarctic Survey showing that ozone levels for Antarctica had dropped 10% below normal levels

- Why did the Nimbus 7 satellite, which had instruments aboard for recording ozone levels, not record similarly low ozone concentrations?

- The ozone concentrations recorded by the satellite were so low they were being treated as outliers by a computer program and discarded!



Sources:

http://exploringdata.cqu.edu.au/ozone.html

http://www.epa.gov/ozone/science/hole/size.html

Real World Anomalies

- Credit Card Fraud
  - An abnormally high purchase made on a credit card

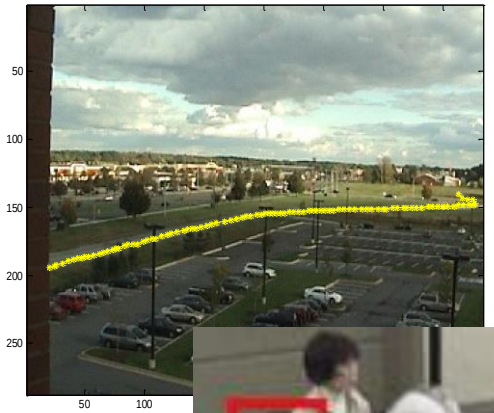
- Cyber Intrusions
  - A web server involved in *ftp* traffic

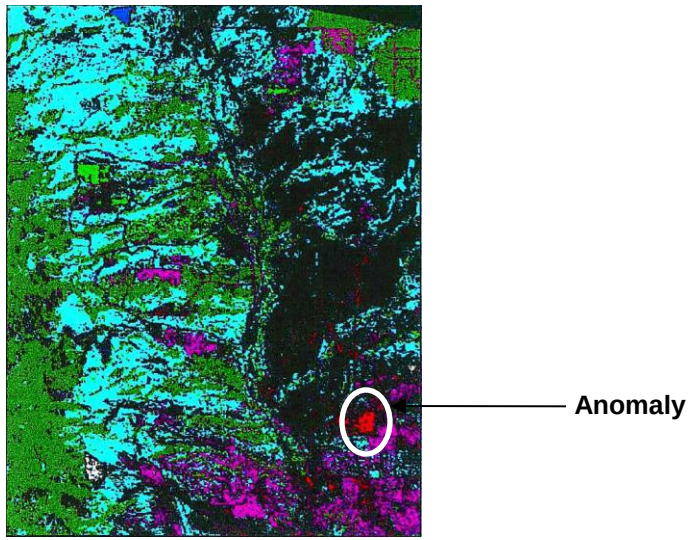
- Healthcare Informatics
  - Indicate disease outbreaks, instrumentation errors, etc.

- Industrial Damage Detection
  - Example: Aircraft Safety
    - Anomalies in engine combustion data

Image Processing

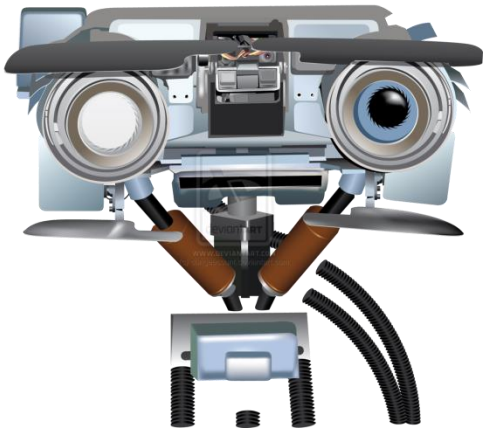
- Detecting outliers in an image monitored over time
- Detecting anomalous regions within an image
- Used in
  - mammography image analysis
  - video surveillance
  - satellite image analysis
- Key Challenges
  - Detecting collective anomalies
  - Data sets are very large





Video Surveillance

https://arxiv.org/pdf/1606.08455.pdf



Density Estimation Method

Anomaly Detection

Anomaly Detection Example

Aircraft engine features:

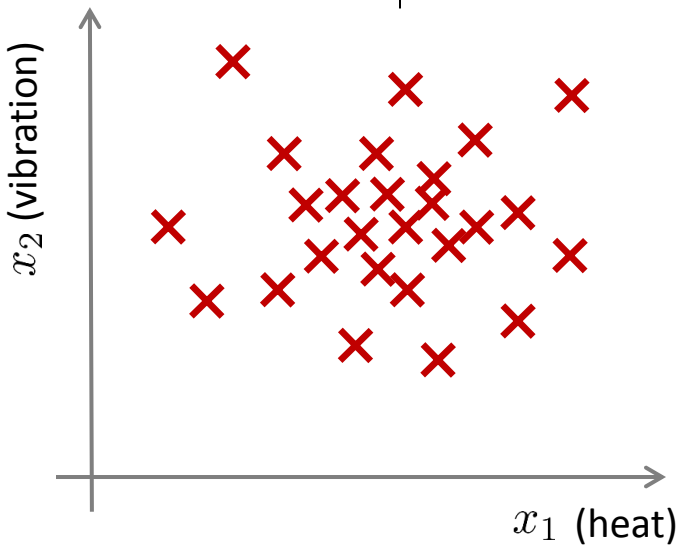
x_1 = heat generated

x_2 = vibration intensity

…

Dataset: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

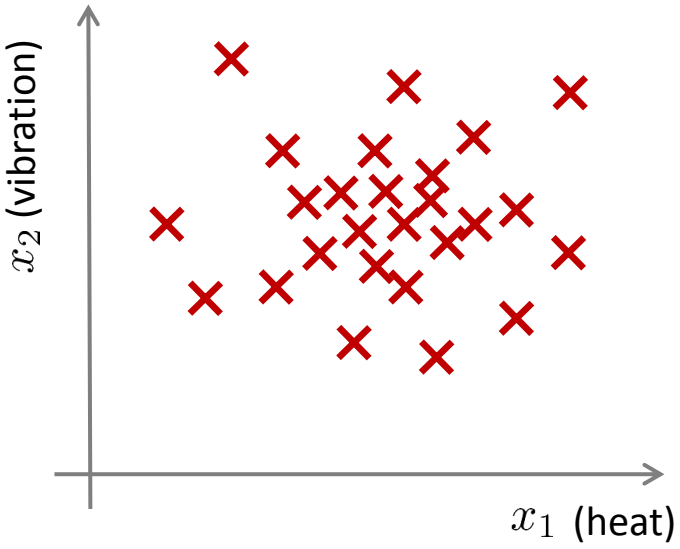
New engine: x_{test}



Density Estimation

Dataset: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

Is x_{test} anomalous?



Anomaly detection example

Fraud detection:

$x^{(i)}$ = features of user i's activities

Model $p(x)$ from data.

Identify unusual users by checking which have $p(x) < \varepsilon$

Manufacturing:

Monitoring computers in a data center.

$x^{(i)}$ = features of machine i

x_1 = memory use, x_2 = number of disk accesses/sec,

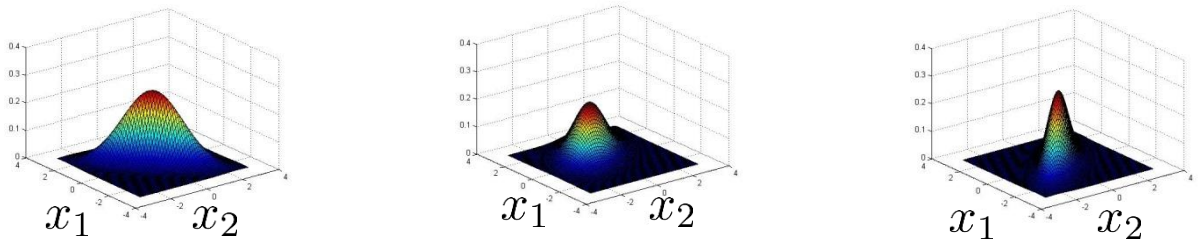
x_3 = CPU load, x_4 = CPU load/network traffic.

…

Example Density Estimation Method: Multivariate Gaussian (Normal) distribution

Parameters μ, Σ

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$



Parameter fitting:

Given training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

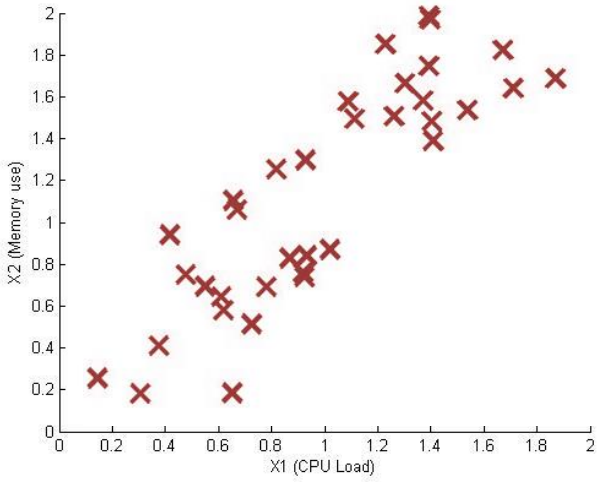
$$\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} \qquad \Sigma = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu)(x^{(i)} - \mu)^T$$

Anomaly detection with the Multivariate Gaussian

1. Fit model $p(x)$ by setting

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

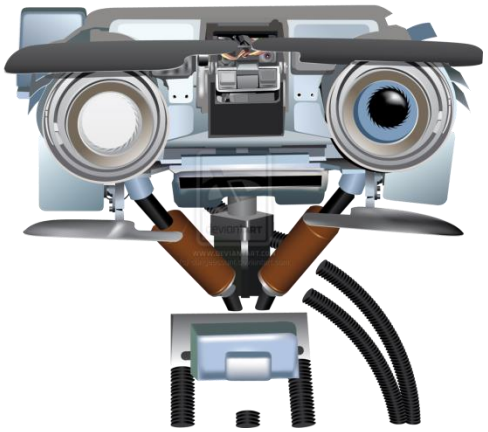
$$\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \mu)(x^{(i)} - \mu)^T$$



2. Given a new example x, compute

$$p(x) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

Flag an anomaly if $p(x) < \varepsilon$



Evaluation

Anomaly Detection

Evaluating an anomaly detection model

When developing a learning algorithm (choosing features, etc.), making decisions is much easier if we have a way of evaluating our learning algorithm.

Assume we have some labeled data, of anomalous and non-anomalous examples. ($y = 0$ if normal, $y = 1$ if anomalous).

Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ (assume normal examples/not anomalous)

Cross validation set: $(x_{cv}^{(1)}, y_{cv}^{(1)}), \ldots, (x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$
Test set: $(x_{test}^{(1)}, y_{test}^{(1)}), \ldots, (x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

Aircraft engines motivating example

10000 good (normal) engines
20 flawed engines (anomalous)

Training set: 6000 good engines
CV: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)
Test: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)

Alternative:
Training set: 6000 good engines
CV: 4000 good engines ($y = 0$), 10 anomalous ($y = 1$)
Test: 4000 good engines ($y = 0$), 10 anomalous ($y = 1$)

Algorithm evaluation

Fit model $p(x)$ on training set $\{x^{(1)}, \ldots, x^{(m)}\}$

On a cross validation/test example x, predict

$$y = \begin{cases} 1 & \text{if } p(x) < \varepsilon \text{ (anomaly)} \\ 0 & \text{if } p(x) \geq \varepsilon \text{ (normal)} \end{cases}$$

Possible evaluation metrics:

- Precision/Recall

$$precision = \frac{true\ positives}{predicted\ positives} = \frac{TP}{TP + FP}$$

$$recall = \frac{true\ positives}{actual\ positives} = \frac{TP}{TP + FN}$$

Can also use cross validation set to choose parameter ε

Anomaly detection *vs.* Supervised learning

- Very small number of positive examples (y=1)
- Large number of negative (y=0) examples

- Many different “types” of anomalies. Hard for any algorithm to learn from positive examples what the anomalies look like; future anomalies may look nothing like any of the anomalous examples we’ve seen so far.

- Large number of positive and negative examples.

- Enough positive examples for algorithm to get a sense of what positive examples are like, future positive examples likely to be similar to ones in training set.

Anomaly detection

- Fraud detection
- Manufacturing (e.g. aircraft engines)
- Monitoring machines in a data center

⋮

vs.

Supervised learning

- Email spam classification
- Weather prediction (sunny/rainy/etc).
- Cancer classification

⋮

Online Detection of Unusual Events in Videos via Dynamic Sparse Coding

Bin Zhao, Li Fei-Fei, Eric Xing

Proceedings of the International Conference on Computer Vision and Pattern Recognition (*CVPR 2011*), Colorado Springs, CO, USA, June 2011

Goal: Detect Unusual Events in Videos

- Example unusual event: entering subway via exit
- Videos are described as spatio-temporal features

Figure 2. Example spatio-temporal interest points

Dictionary-based Anomaly Detection

- Learn a dictionary of bases corresponding to usual events:
  - a usual event should be reconstructible from a small number of such bases, and
  - the reconstruction weights should change smoothly over space/time across actions in such events.
  - an unusual event is either not reconstructible from the dictionary of usual events with small error, or,
  - Needs a large number of bases, in a temporal-spatially non-smooth fashion.
- Must: Learn a good dictionary of bases representing usual events
- Must: Update the dictionary online to adapt to changing content of the video

Algorithm: Look for High Reconstruction Error

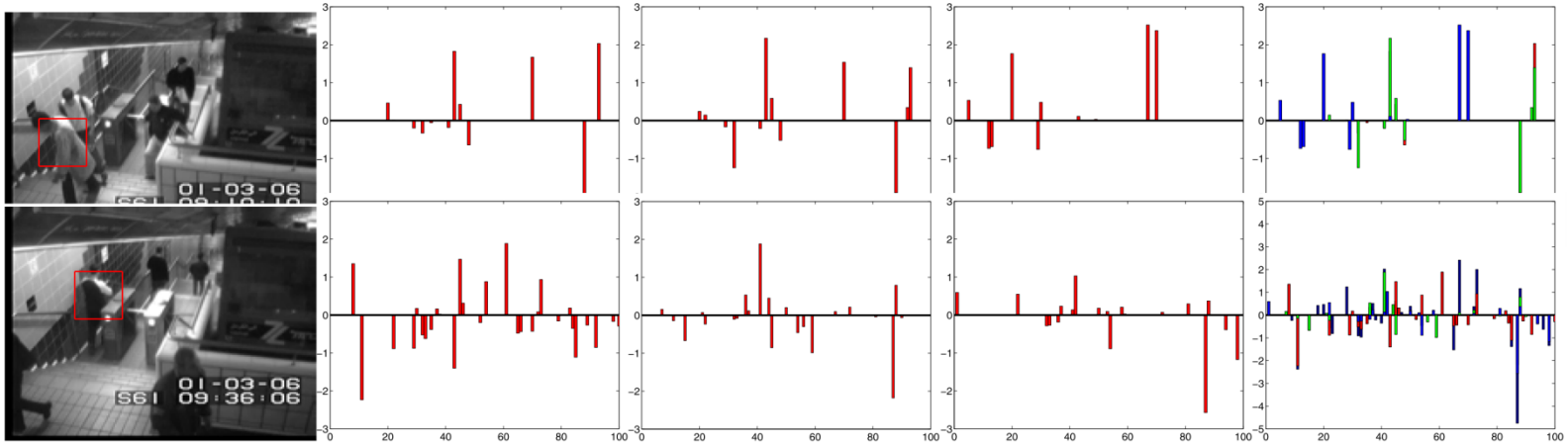
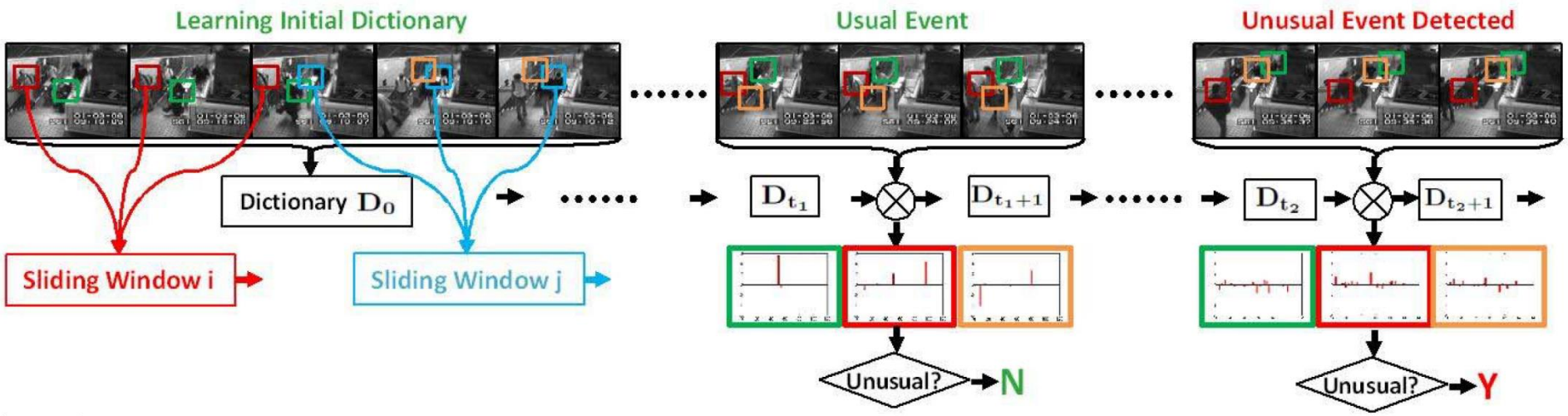
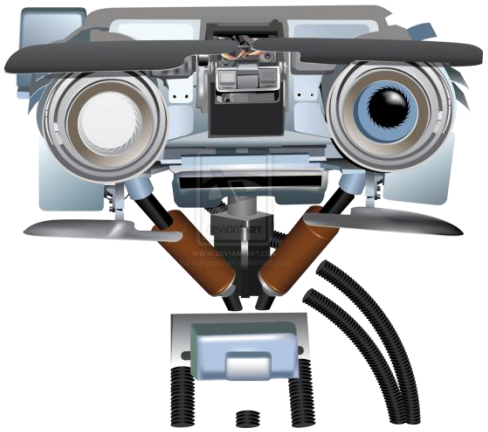


Figure 3. First row: usual event (leaving subway exit); second row: unusual event (entering subway exit). From left to right: example frame and sliding window, reconstruction vectors for 3 cuboids, plot all 3 reconstruction vectors on the same figure.

Algorithm


Learning Initial Dictionary
Usual Event
Unusual Event Detected
Dictionary D_0
Sliding Window i
Sliding Window j
D_{t_1}
D_{t_1+1}
D_{t_2}
D_{t_2+1}
Unusual?
N
Unusual?
Y



Semi-Supervised Learning

Slides credit: Jerry Zhu, Aarti Singh

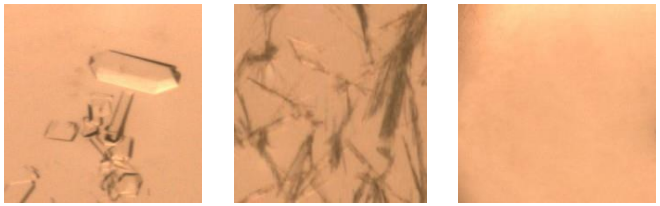
Supervised Learning

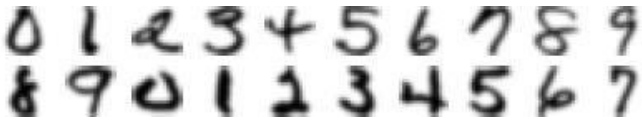
Feature Space $\mathcal{X}$ **Label** Space $\mathcal{Y}$

Goal: Construct a **predictor** $f : \mathcal{X} \to \mathcal{Y}$ to minimize

$\text{loss}(Y, f(X))$

Labeled and Unlabeled data





"Crystal" "Needle" "Empty"

"0" "1" "2" …

"Sports"
"News"
"Science"
…

Human expert/
Special equipment/
Experiment

Unlabeled data, X_i

Labeled data, Y_i

Cheap and abundant !

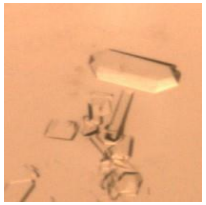
Expensive and scarce !

Semi-Supervised learning

Training data $\{(X_i, Y_i)\}_{i=1}^n$ $\{X_i\}_{i=1}^m$ ⇨ **Learning algorithm** ⇨ Prediction rule $\widehat{f}_{n,m}$

Supervised learning (SL)

Labeled data $\{X_i, Y_i\}_{i=1}^n$



X_i

"Crystal"

Y_i

Semi-Supervised learning (SSL)

Labeled data $\{X_i, Y_i\}_{i=1}^n$ **and** Unlabeled data $\{X_i\}_{i=1}^m$ $m \gg n$

Goal: Learn a better prediction rule than based on labeled data alone.

Semi-Supervised learning in Humans

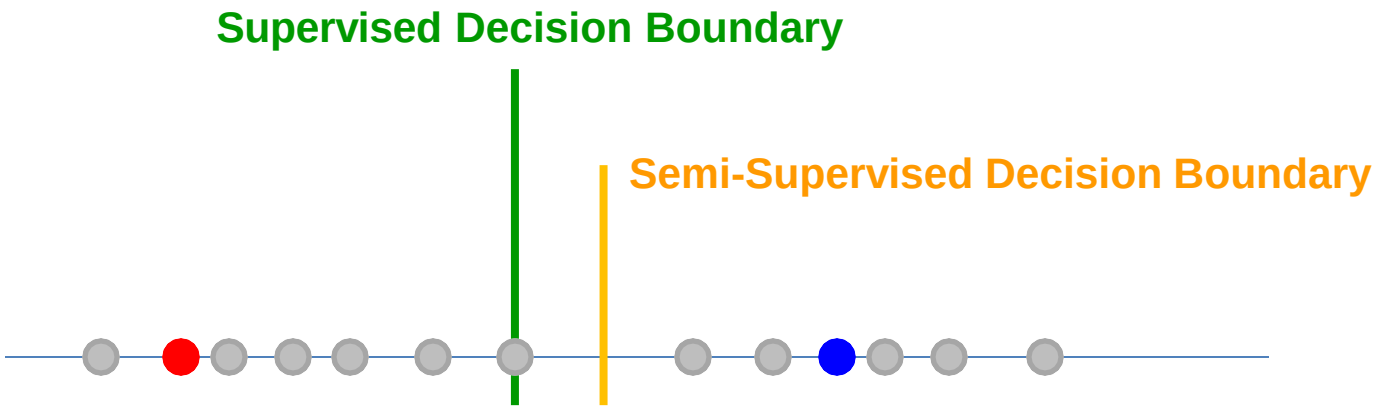
Cognitive science

Computational model of how humans learn from labeled and unlabeled data.

- concept learning in children: $\mathbf{x}$=animal, y=concept (e.g., dog)
- Daddy points to a brown animal and says "dog!"
- Children also observe animals by themselves

Can unlabeled data help?

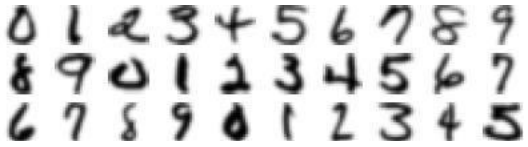
- Positive labeled data
- Negative labeled data
- Unlabeled data



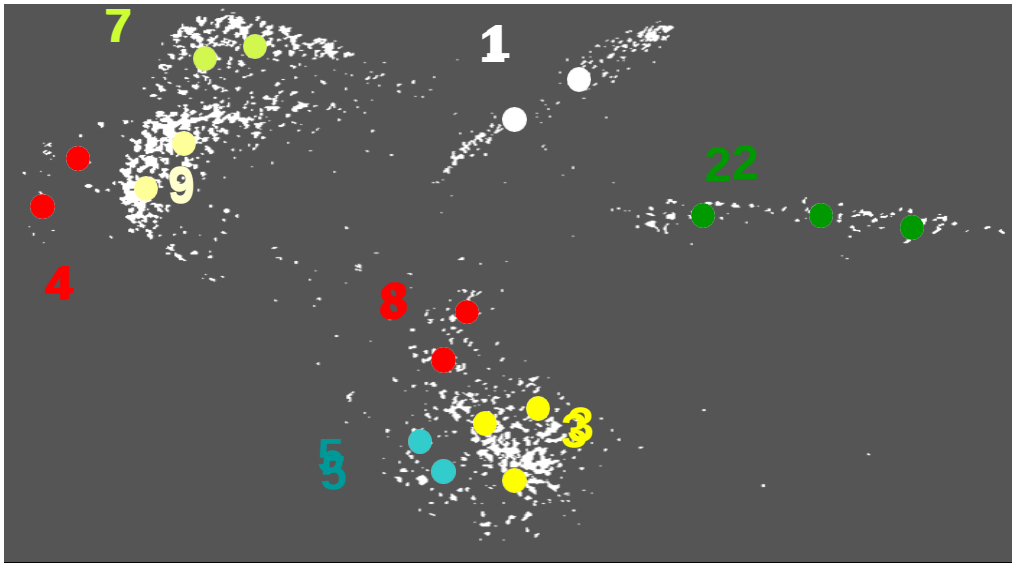
Assume each class is a coherent group (e.g. Gaussian)

Then unlabeled data can help identify the boundary more accurately.

Can unlabeled data help?

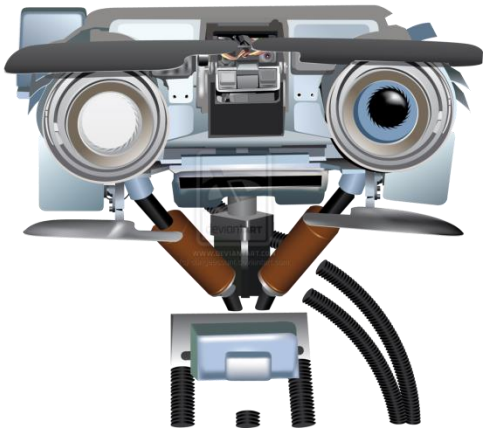
Unlabeled Images 

Labels “0” “1” “2” …



This embedding can be done by manifold learning algorithms, e.g. t-SNE

“Similar” data points have “similar” labels



Algorithms

Semi-Supervised Learning

Slides credit: Jerry Zhu, Aarti Singh

Some SSL Algorithms

- Self-Training
- Generative methods, mixture models
- Graph-based methods
- Co-Training
- Semi-supervised SVM
- Many others

Notation

- instance $\mathbf{x}$, label y
- learner $f : \mathcal{X} \mapsto \mathcal{Y}$
- labeled data $(X_l, Y_l) = \{(x_{1:l}, y_{1:l})\}$
- unlabeled data $X_u = \{\mathbf{x}_{l+1:l+u}\}$, available during training. Usually $l \ll u$. Let $n = l + u$
- test data $\{(x_{n+1\ldots}, y_{n+1\ldots})\}$, not available during training

Self-training

Our first SSL algorithm:

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
1. Initially, let $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ and $U = \{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
2. Repeat:
3. Train f from L using supervised learning.
4. Apply f to the unlabeled instances in U.
5. Remove a subset S from U; add $\{(\mathbf{x}, f(\mathbf{x}))|\mathbf{x} \in S\}$ to L.

Self-training is a *wrapper* method

- the choice of learner for f in step 3 is left completely open
- good for many real world tasks like natural language processing
- but mistake by f can reinforce itself

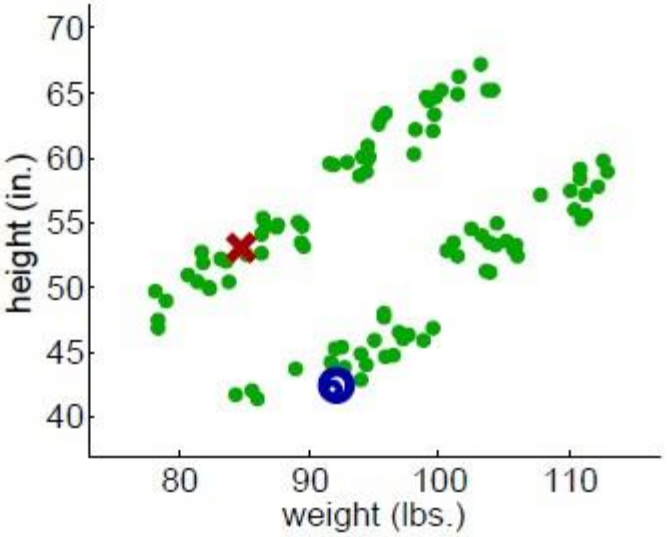
Self-training Example

Propagating 1-NN

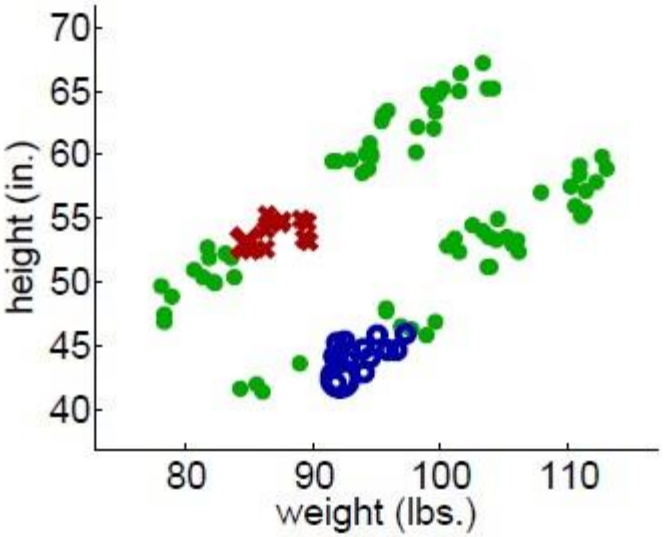
Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$, distance function $d()$.

1. Initially, let $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ and $U = \{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
2. Repeat until U is empty:
3. Select $\mathbf{x} = \text{argmin}_{\mathbf{x} \in U} \min_{\mathbf{x}' \in L} d(\mathbf{x}, \mathbf{x}')$.
4. Set $f(\mathbf{x})$ to the label of $\mathbf{x}$'s nearest instance in L. Break ties randomly.
5. Remove $\mathbf{x}$ from U; add $(\mathbf{x}, f(\mathbf{x}))$ to L.

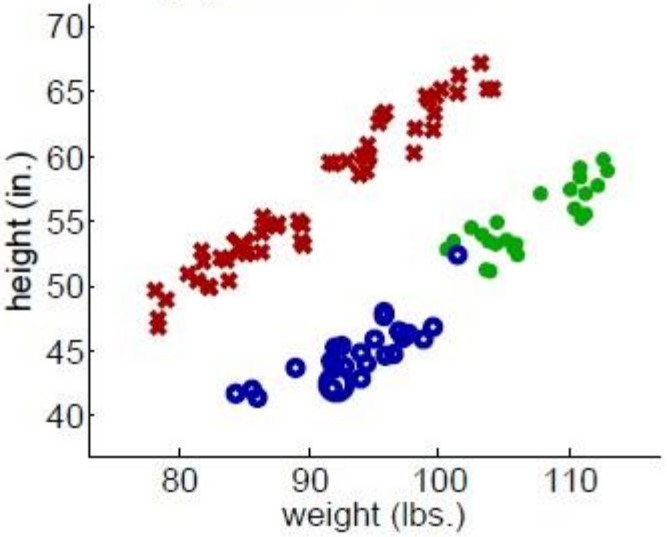
Propagating 1-Nearest-Neighbor: now it works



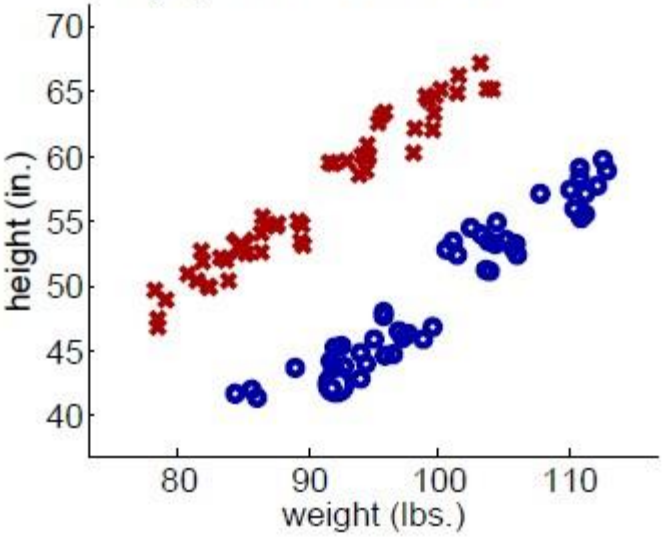
(a) Iteration 1



(b) Iteration 25



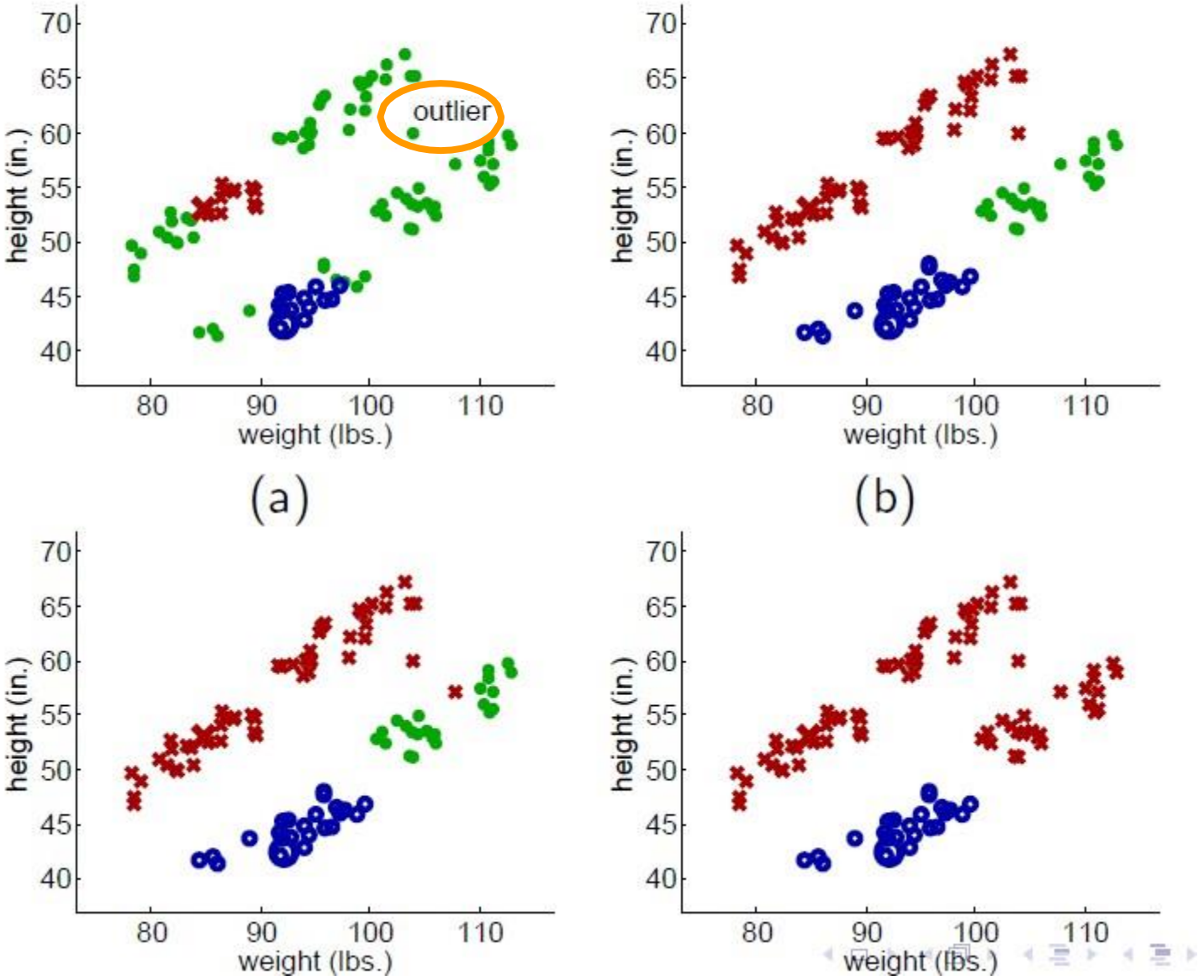
(c) Iteration 74



(d) Final labeling of all instances

Propagating 1-Nearest-Neighbor: now it doesn't

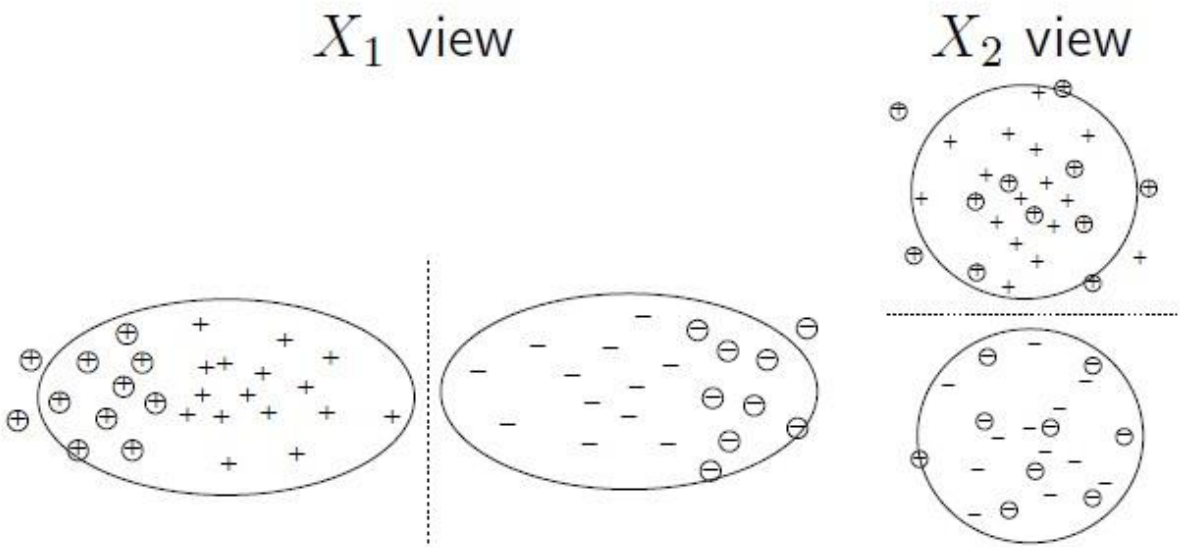
But with a single outlier...



Co-training

Assumptions

- feature split $x = [x^{(1)}; x^{(2)}]$ exists
- $x^{(1)}$ or $x^{(2)}$ alone is sufficient to train a good classifier



Co-training Algorithm

- Co-training (Blum & Mitchell, 1998) (Mitchell, 1999) assumes that

  (i) features can be split into two sets;
  (ii) each sub-feature set is sufficient to train a good classifier.

  - Initially two separate classifiers are trained with the labeled data, on the two sub-feature sets respectively.

  - Each classifier then classifies the unlabeled data, and 'teaches' the other classifier with the few unlabeled examples (and the predicted labels) they feel most confident.

  - Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats.

Co-training Algorithm

Blum & Mitchell'98

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$
each instance has two views $\mathbf{x}_i = [\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}]$,
and a learning speed k.

1. let $L_1 = L_2 = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$.
2. Repeat until unlabeled data is used up:
3. Train view-1 $f^{(1)}$ from L_1, view-2 $f^{(2)}$ from L_2.
4. Classify unlabeled data with $f^{(1)}$ and $f^{(2)}$ separately.
5. Add $f^{(1)}$'s top k most-confident predictions $(\mathbf{x}, f^{(1)}(\mathbf{x}))$ to L_2.
   Add $f^{(2)}$'s top k most-confident predictions $(\mathbf{x}, f^{(2)}(\mathbf{x}))$ to L_1.
   Remove these from the unlabeled data.