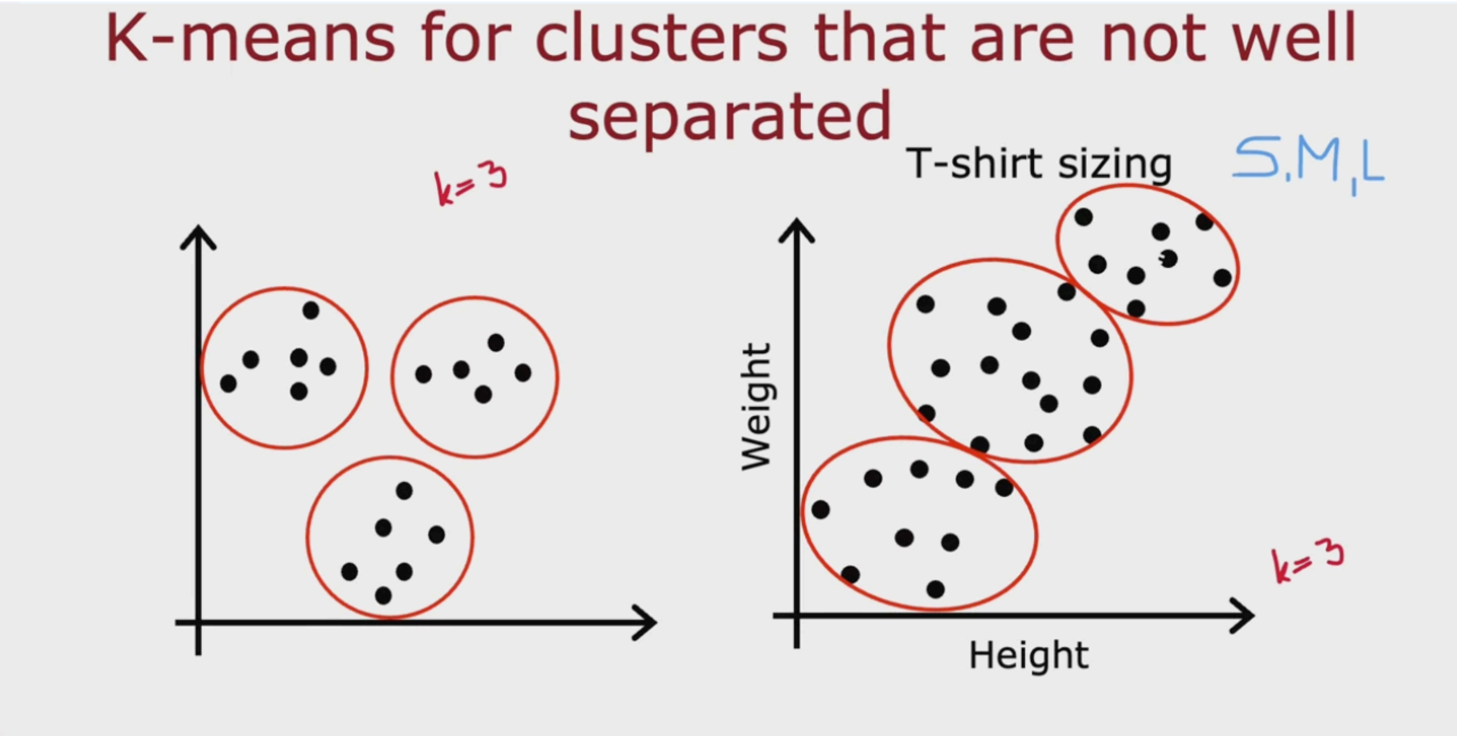
Unsupervised Learning, Recommender Systems, Reinforcement Learning

Supervised Learning

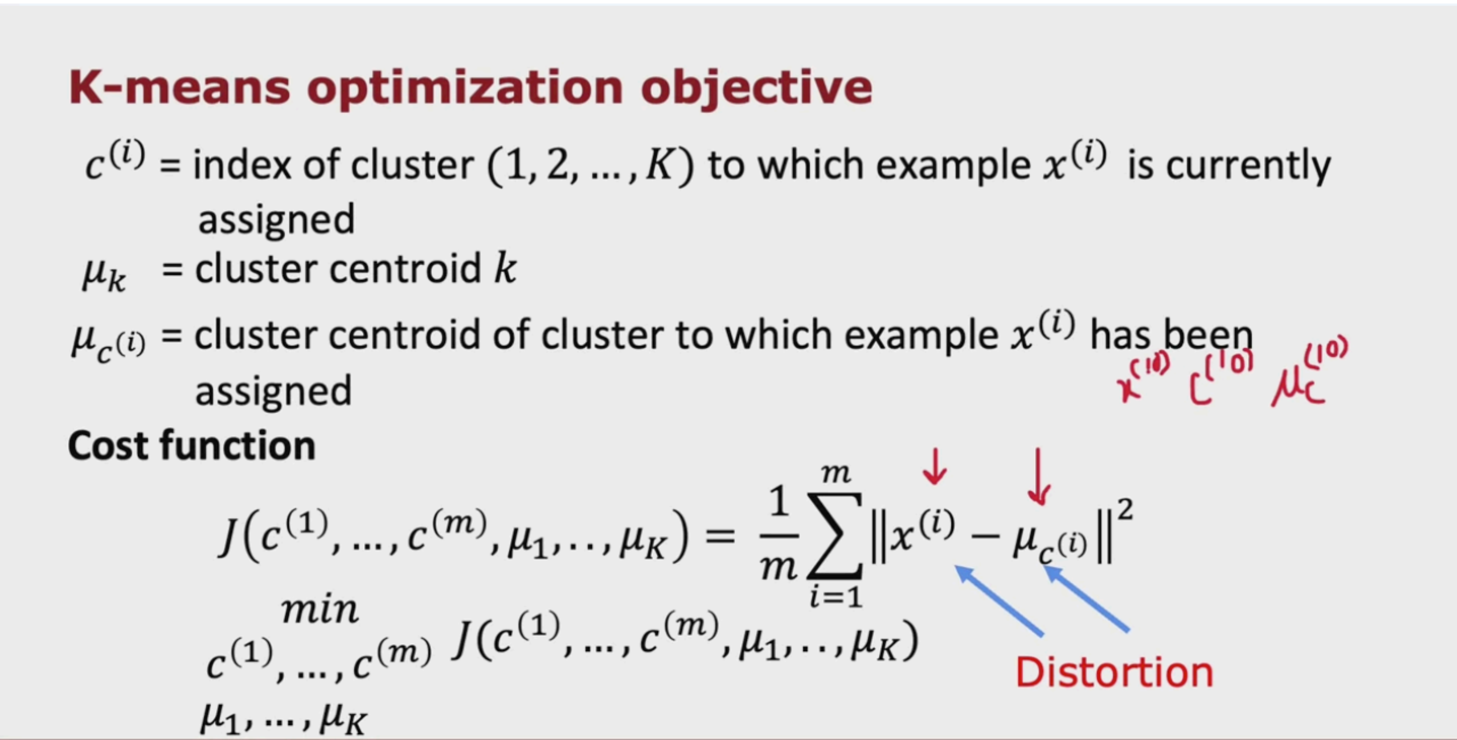
Clustering

First - takes random guess, looks how much close these random *cluster centroids* are.

K-means Algorithm



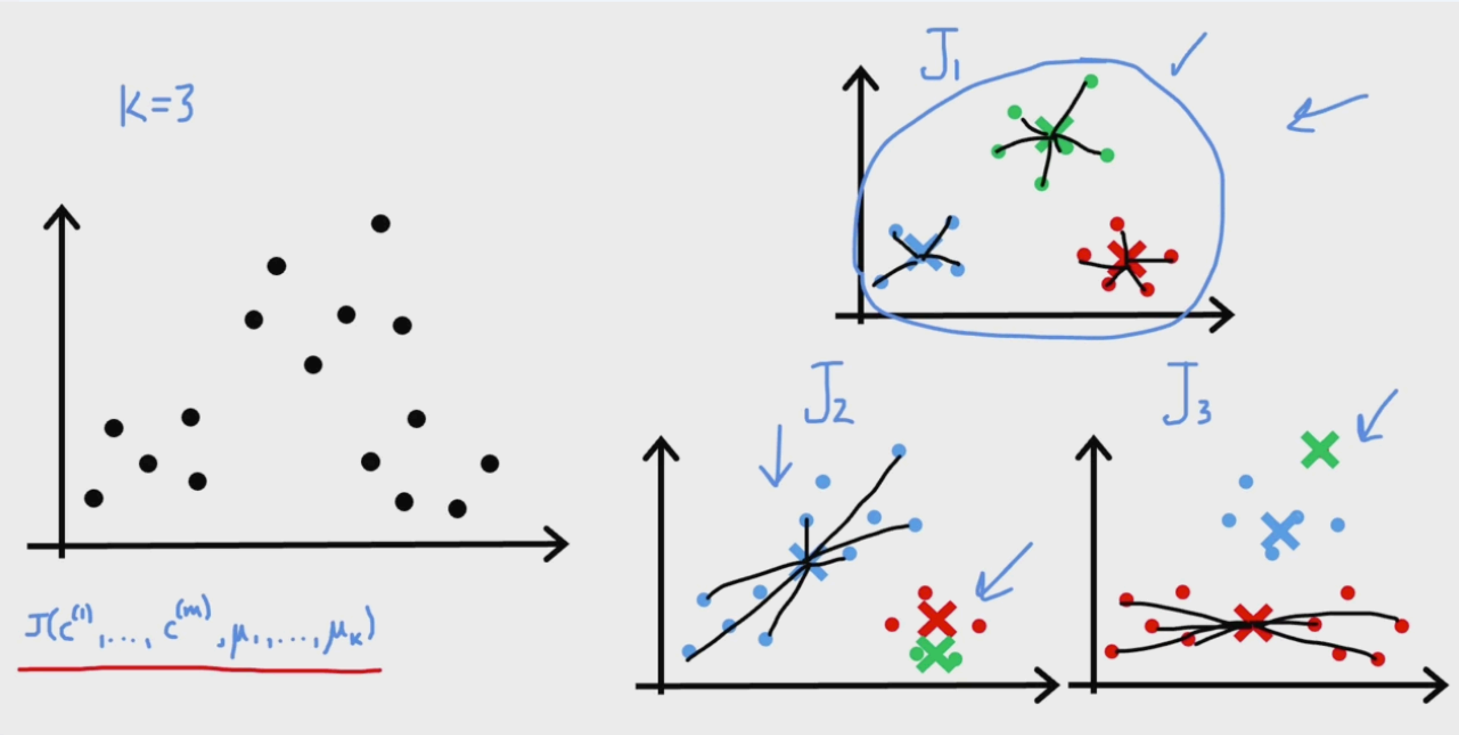
Optimizing a specific cost function.



Random initialization of *K*

K < *m*

Local Optima's can be a problem.



Randomly initialize *K-means* -> *50-1000 different initializations*
Pick set of clusters that gave lowest cost *J*

Right Value of *K*

Elbow method - not practical

Anomaly Detection

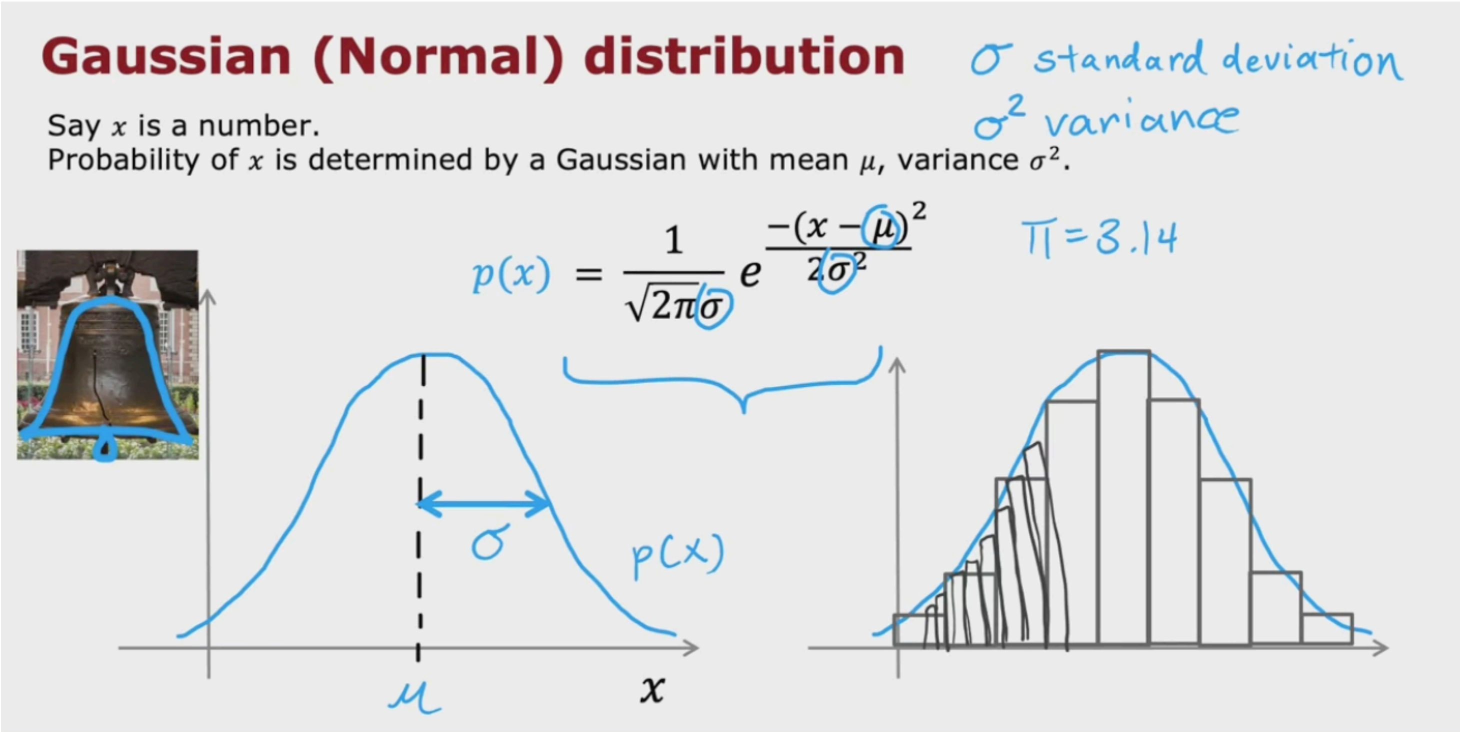
feature vector

Density Estimation

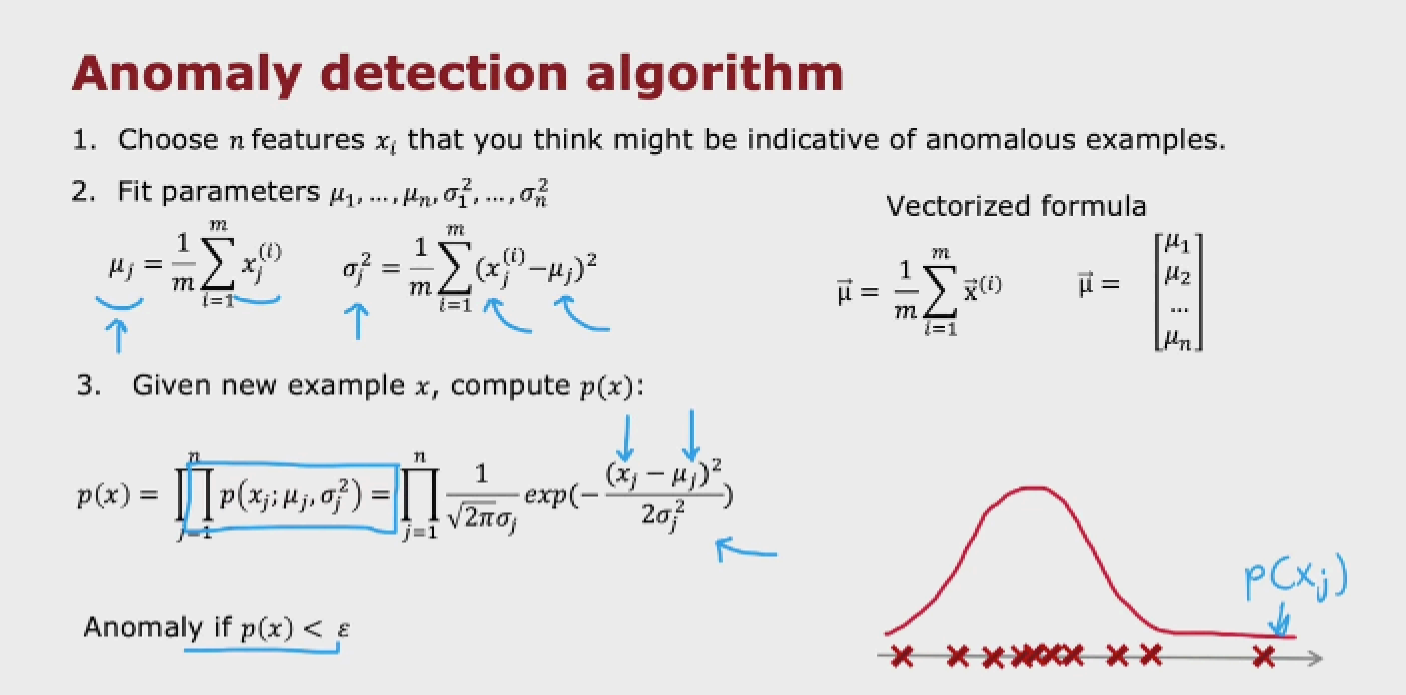
Probability of x being seen in dataset. areas with different probabilities.
epsilon as the limit. if less then epsilon => anomaly.
where there are many features. (# of pressed keys by user in minute, transaction #, clicks, visits, CPU load, CPU load / network activities)

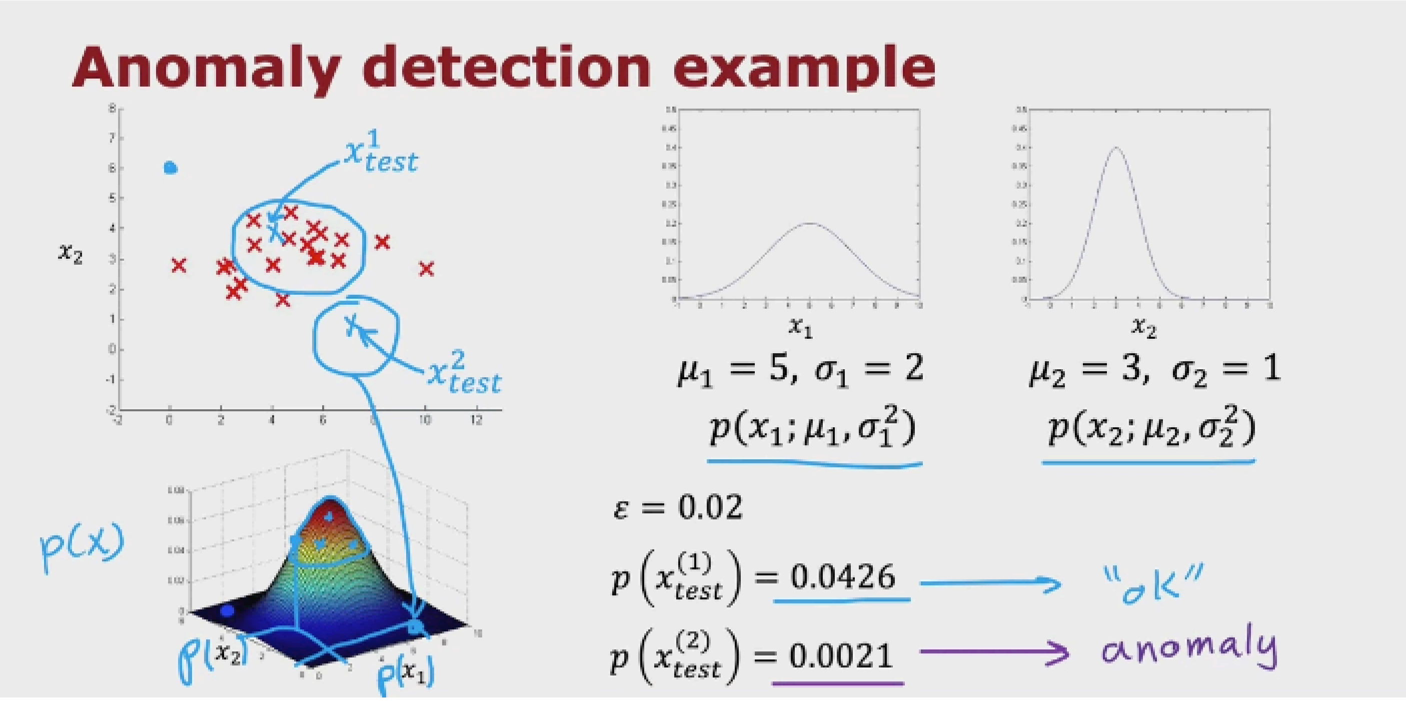
Gaussian - Normal - Bell Shape Distribution

std dev, (std dev)^2 = variance



Anomaly Detection:





How to evaluate anomaly detection systems?

Real-number Evaluation:

Making decisions if much easier if we have a way of evaluating learning algorithm.
Some labeled data, anomalous and non-anomalous.

Aircraft engines monitoring example

10000 good (normal) engines — 2 to 50

~~20~~ 2 flawed engines (anomalous) — $y=1$

Training set: 6000 good engines — $y=0$ — train algorithm on training set

CV: 2000 good engines ($y = 0$) 10 anomalous ($y = 1$)

use cross validation set — tune ε — tune x_j

Test: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)

Alternative: No test set — Use if very few labeled anomalous examples — higher risk of overfitting

Training set: 6000 good engines

CV: 4000 good engines ($y = 0$), ~~20~~ 2 anomalous ($y = 1$)

tune ε tune x_j

Algorithm evaluation

course 2 week 3

skewed datasets

Fit model $p(x)$ on training set $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

On a cross validation/test example x, predict

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

10

2000

Possible evaluation metrics:

- True positive, false positive, false negative, true negative
- Precision/Recall
- F_1-score

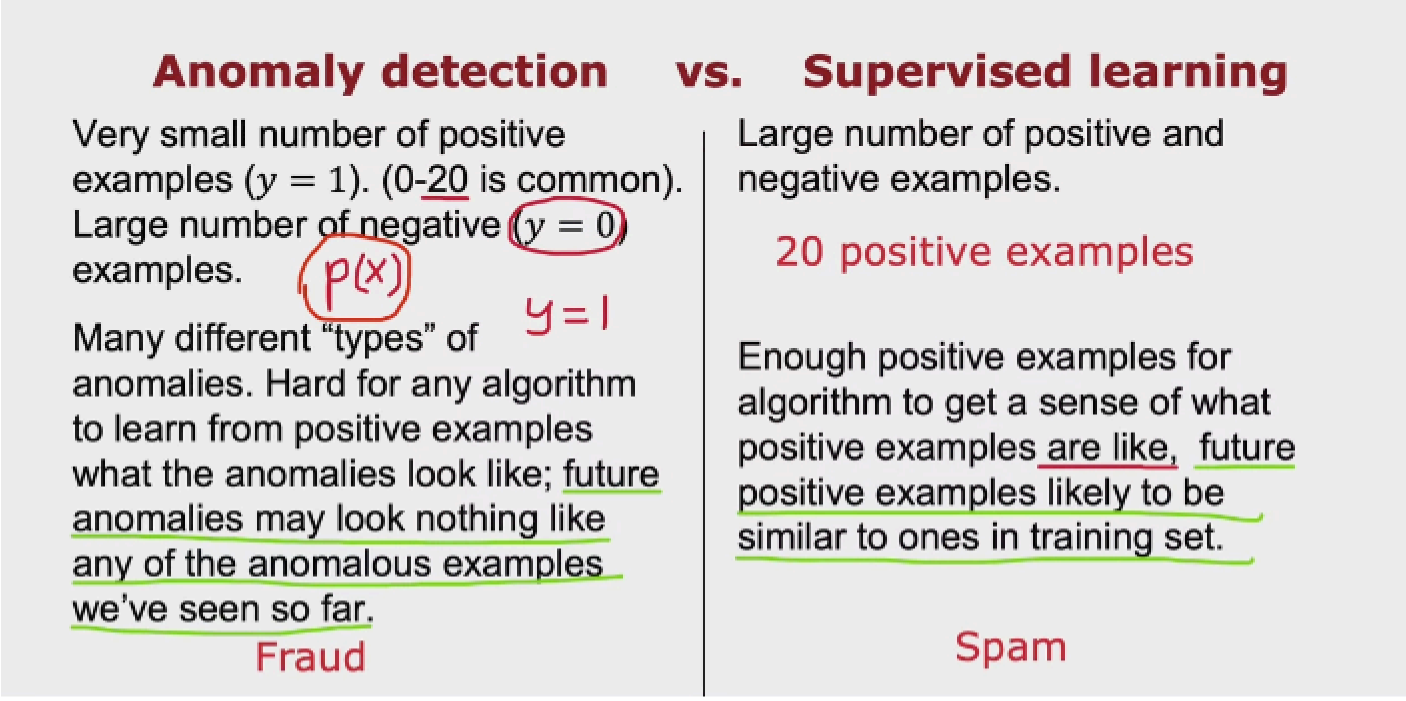
Use cross validation set to choose parameter ε

Anomaly Detection vs Supervised Learning:

Number of examples is the key factor in between.

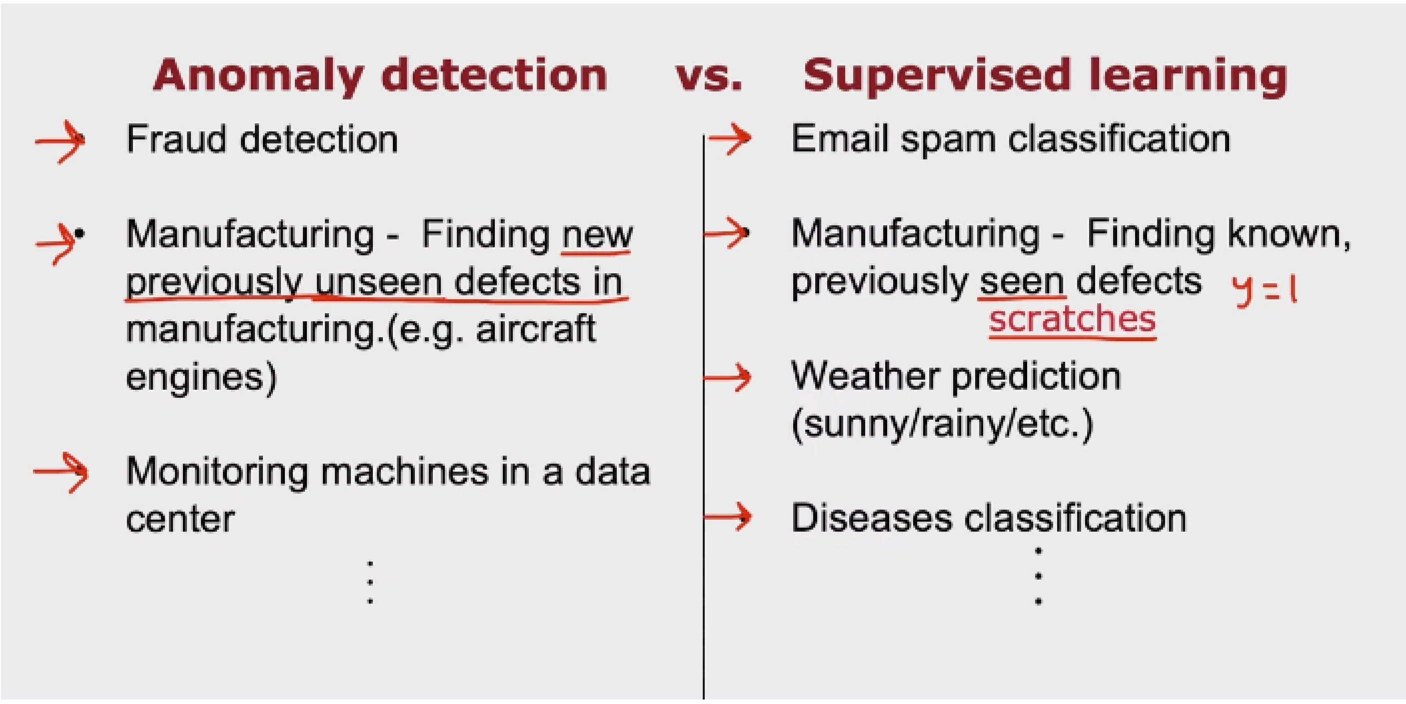
Anomaly Detection: Learns what to accept as normal to some extend.

Supervised Learning: Learns what looks like an acceptable or unacceptable example.



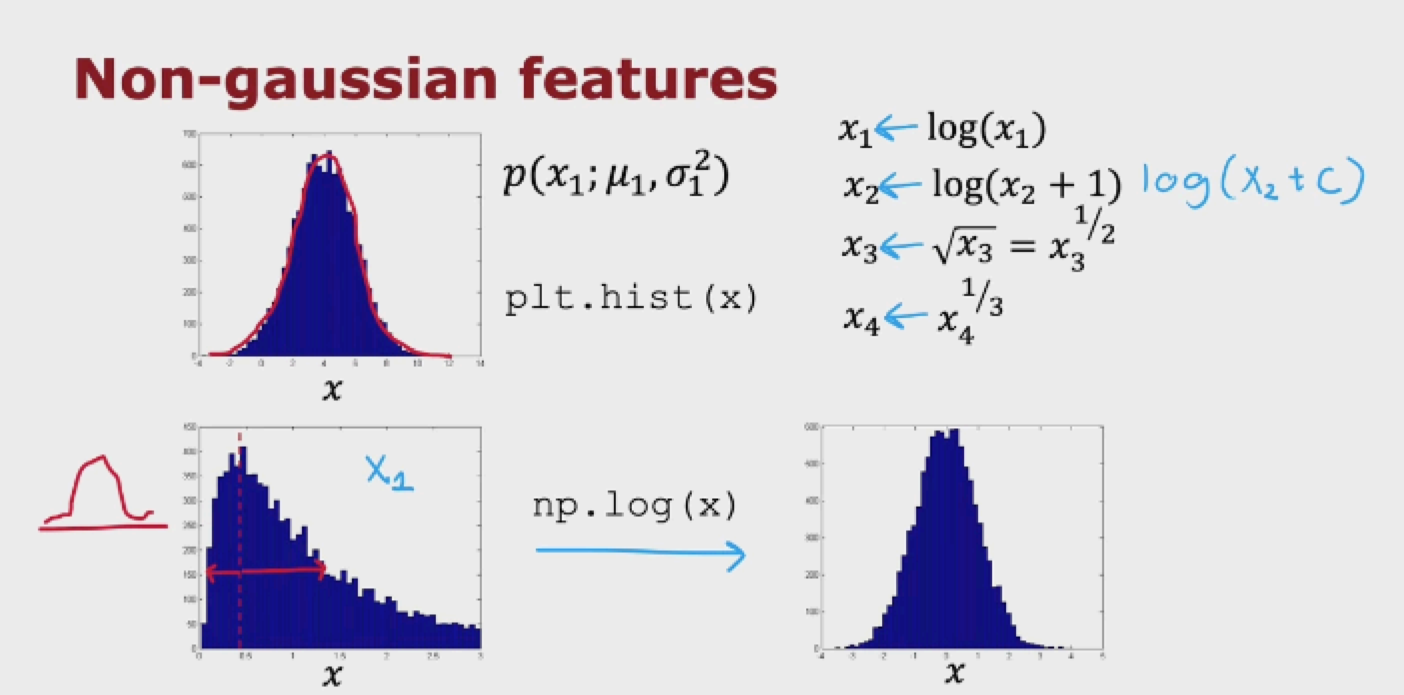
Spam -> probably you seen similar before

Fraud -> oncoming examples may be completely different from what we have seen.



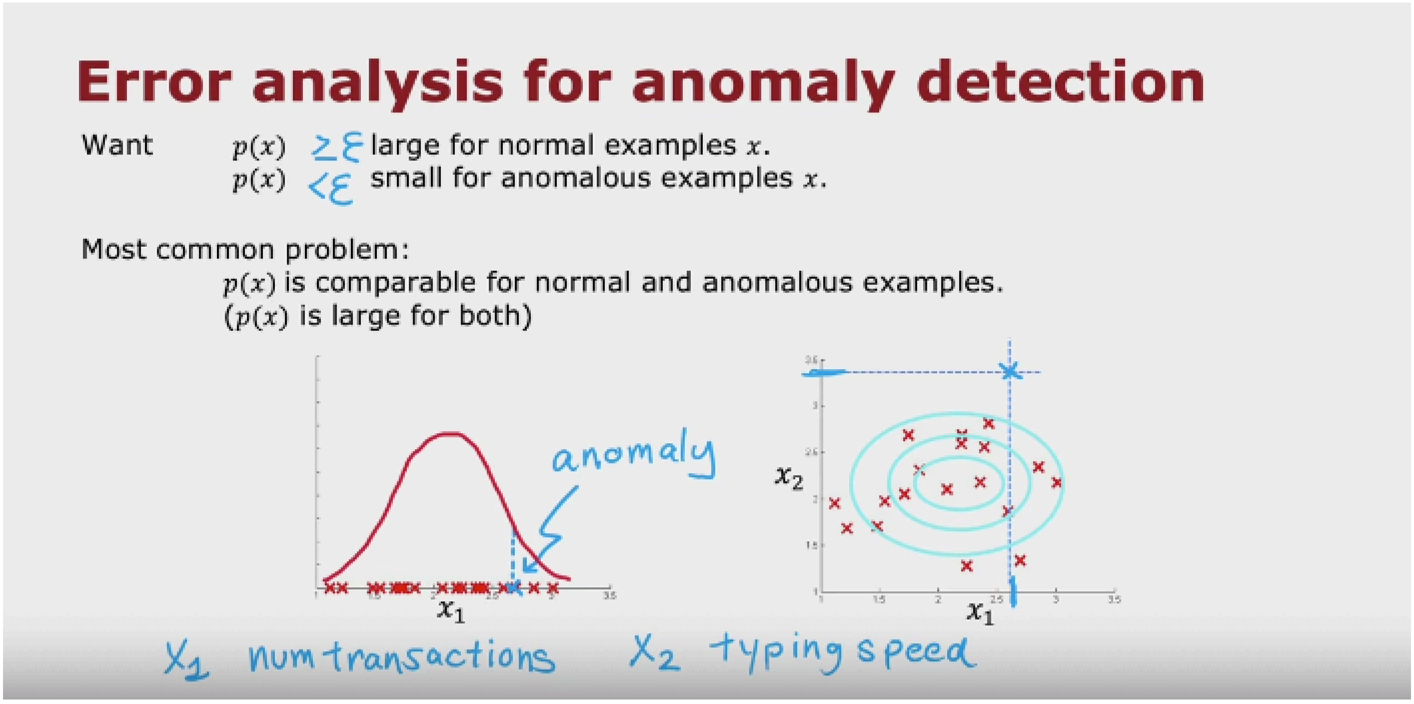
when sufficient number of both normal and anomaly examples, use supervised learning (1:1).

Choosing/Manipulating Features For User:



Play with features/values.

Error Analysis for Anomaly Detection



Find additional features that when combined gives reasonable outs, when some features alone fail to explain/detect an anomaly.