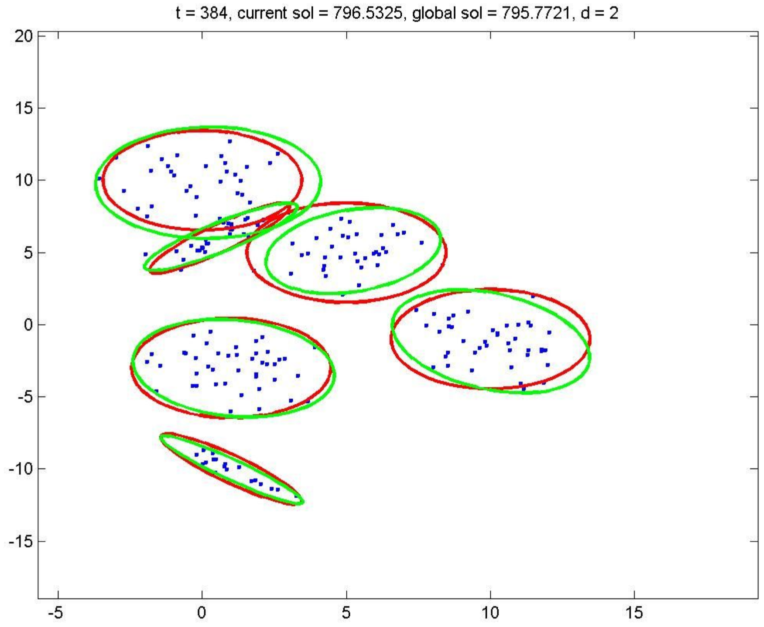
VISUALIZATION AND EVALUATION

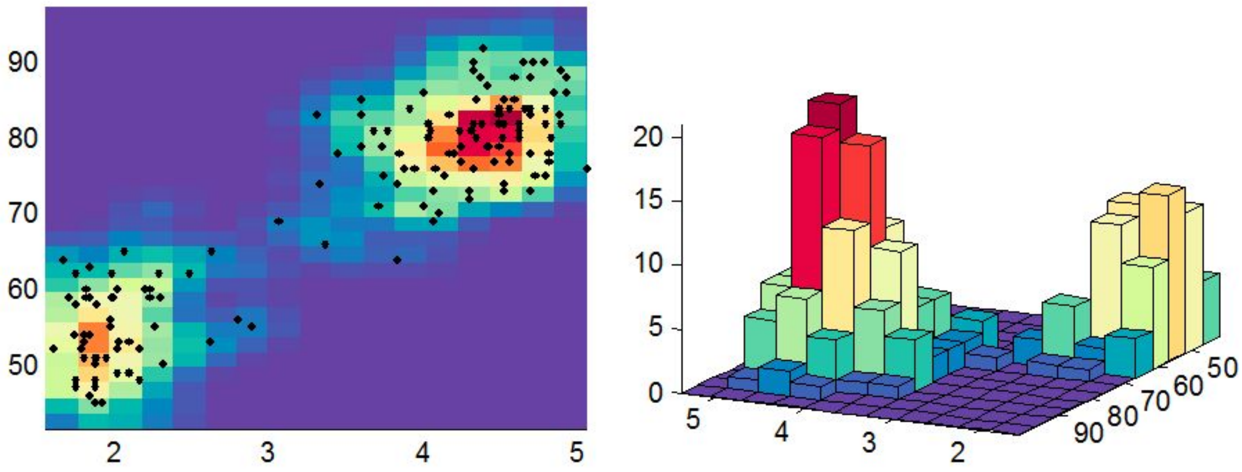
Clustering

- Easy to visualize in low dimension but hard to do so in high dimension



Histogram estimation in N-dimension

- Cut the space into N-dimensional cube
  - How many cubes are there?
  - Assume I want around 10 samples per cube to be able to estimate a nice distribution without overfitting. How many more samples do I need per one additional dimension?



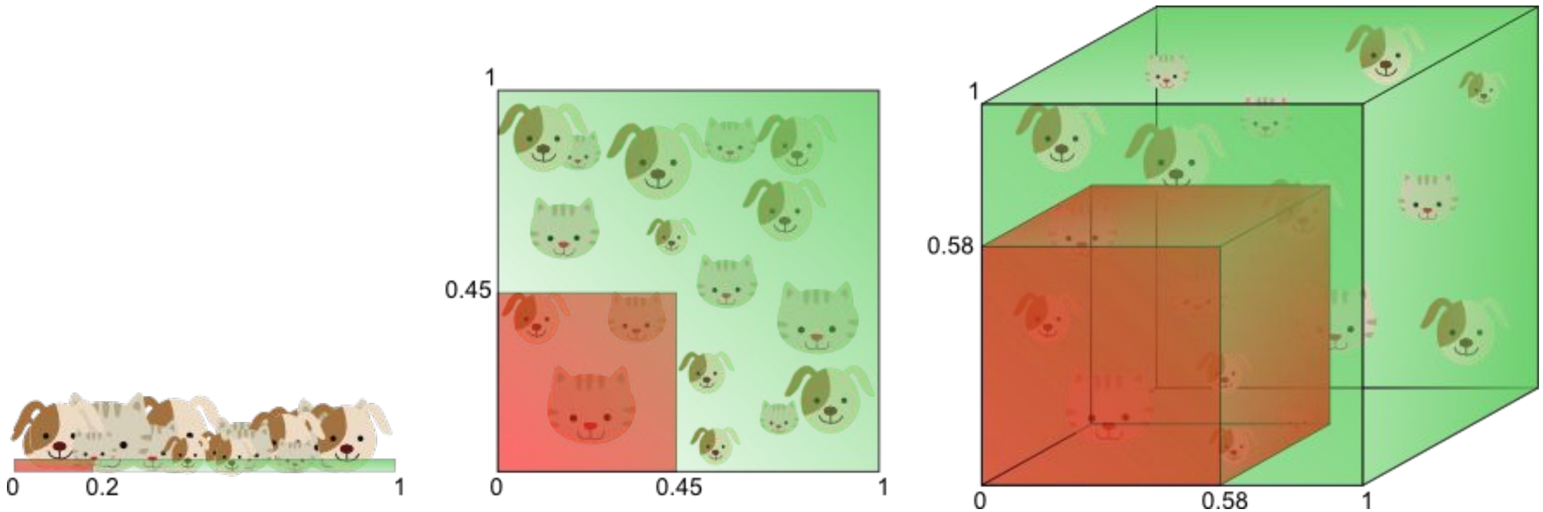
https://www.mathworks.com/matlabcentral/fileexchange/45325-efficient-2d-histogram--no-toolboxes-needed

The curse of dimensionality

https://erikbern.com/2015/10/20/nearest-neighbors-and-vector-models-epilogue-curse-of-dimensionality.html

The Curse of Dimensionality

- Harder to visualize or see structure of
  - Verifying that data come from a straight line/plane needs n+1 data points
- Hard to search in high dimension – More runtime
- Need more data to get a get a good estimation of the data



http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/

Combating the curse of dimensionality

- Feature selection
  - Keep only “Good” features
- Feature transformation (Feature extraction)
  - Transform the original features into a smaller set of features

Feature selection

- Proper methods
  - Algorithm that handles high dimension well and do selection as a by product
  - Tree-based classifiers
    - Random forest
  - Adaboost
  - Genetic Algorithm

Feature transformation

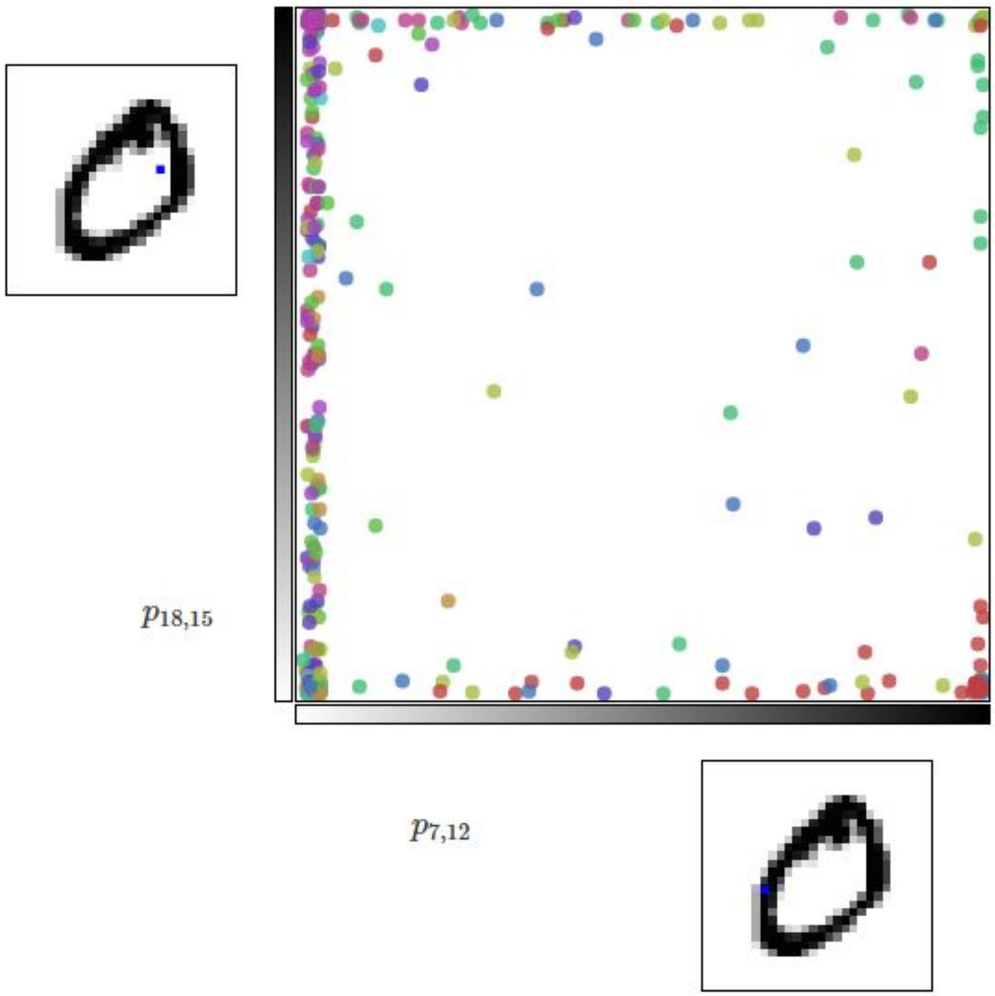
- Principal Component Analysis
- Linear Discriminant Analysis (NOT Latent Dirichlet Allocation)
- Random Projections

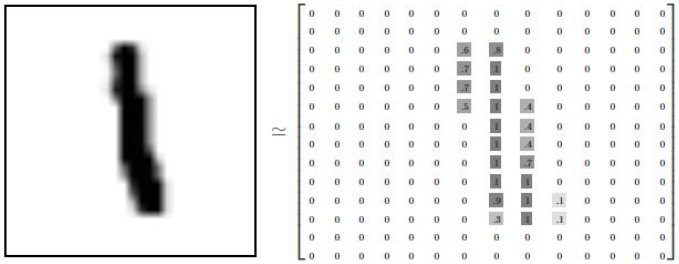
Visualization

But what if we, as humans, want to get a sense of our data?

- Interpretability (in some sense)

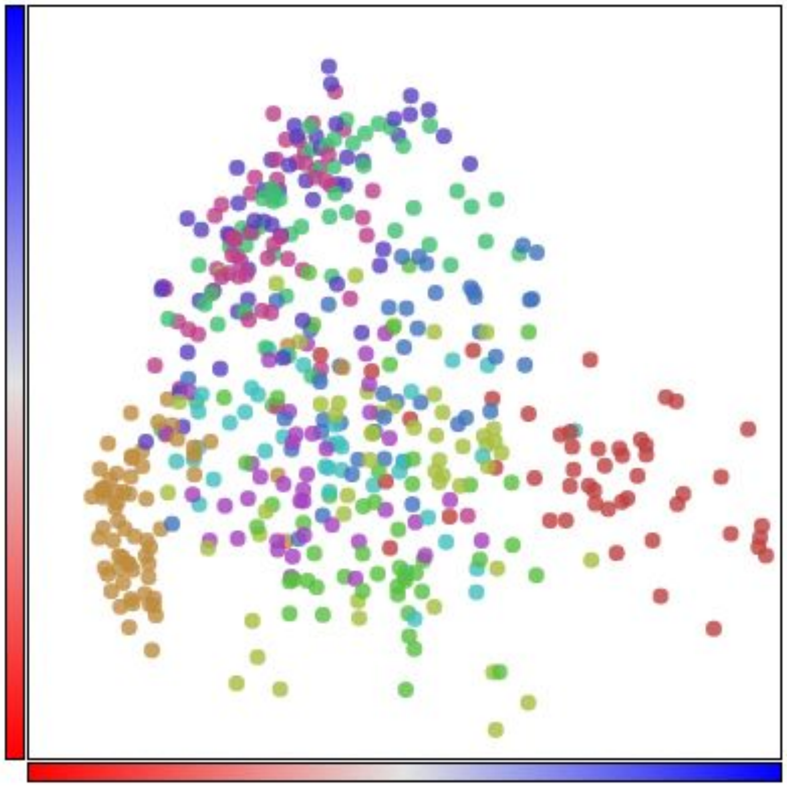
Visualizing MNIST

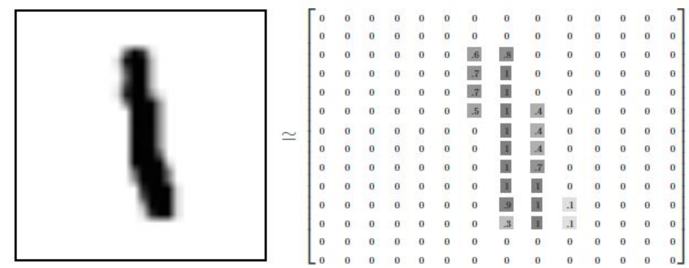


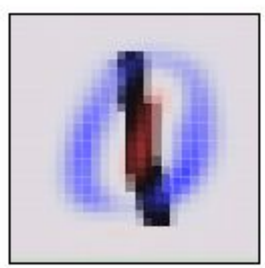


http://colah.github.io/posts/2014-10-Visualizing-MNIST/

PCA with MNIST







http://colah.github.io/posts/2014-10-Visualizing-MNIST/

t-distributed Stochastic Neighbor Embedding (t-SNE)

Preserves neighbor (preserves local distance).

- Things close together should be close together in the projected space
- Prefer using few projected dimensions (2-3)

Defining neighbors

Define $P_{j|i}$ probability that *i* would pick *j* as its neighbor

- Assume *i* picks proportional to Gaussian centered at *i*

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2)/2\sigma_i^2}{\sum_{k \neq i} exp(-||x_i - x_k||^2)/2\sigma_i^2}$$

$P_{i|i} = 0$ since we don't want to have it pick itself

The variance is fixed to some value.

Defining neighbors

Define $q_{j|i}$ probability that *i* would pick *j* as its neighbor

Assume *i* picks proportional to Gaussian centered at *i*

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2)/2\sigma_i^2}{\sum_{k \neq i} exp(-||x_i - x_k||^2)/2\sigma_i^2}$$

When projected to set of points $\{y_i\}$, define $q_{j|i}$ the probability that *i* would pick *j* in embedding/latent space

$$q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq i} exp(-||y_i - y_k||^2)}$$

We set the variance in the y space to be 1/sqrt(2)

Defining neighbors

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2)/2\sigma_i^2}{\sum_{k \neq i} exp(-||x_i - x_k||^2)/2\sigma_i^2}$$

$$q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq i} exp(-||y_i - y_k||^2)}$$

We expect p and q to be the same -> small distance

How to measure distance between probability functions?

Kullback-Leibler (KL) divergence

KL divergence

Distance between two distributions

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)} = -\sum_i P(i) log \frac{Q(i)}{P(i)}$$

Note $D_{KL}(P||Q) \neq D_{KL}(Q||P)$ (Not a real distance)

Always positive. Equals 0 iff Q = P at every point.

P(head) = 0.5 P(tail) = 0.5

Q(head) = 0.7 Q(tail) = 0.3

$D_{KL}(P||Q)$ = 0.5*ln 0.5/0.7 + 0.5*ln 0.5/0.3 = 0.087

$D_{KL}(Q||P)$ = 0.7*ln 0.7/0.5 + 0.3*ln 0.3/0.5 = 0.082

Loss function

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2)/2\sigma_i^2}{\sum_{k \neq i} exp(-||x_i - x_k||^2)/2\sigma_i^2} \qquad q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq i} exp(-||y_i - y_k||^2)}$$

We expect p and q to be the same -> small distance

Loss function

All points i

KL computes over j

$$\sum_i D_{KL}(p_i || q_i)$$

$$D_{KL}(P||Q) = \sum_i \boxed{P(j)} log \frac{P(i)}{Q(j)}$$

Note P can be considered as the weight for the distance
Where p is large but q is small -> large penalty
q is small but p is large -> small penalty
D(p||q) focuses on local structure in p

What are we minimizing wrt?
How to minimize loss?

Variance

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2)/2\sigma_i^2}{\sum_{k \neq i} exp(-||x_i - x_k||^2)/2\sigma_i^2}$$

How to set the variance of our original space?

A single variance for all points is not ideal.

- Want small variance for dense parts
- Want big variance for sparse parts

Set variance by amount of neighbors you want!

How to quantify amount of neighbors?

Perplexity

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2)/2\sigma_i^2}{\sum_{k \neq i} exp(-||x_i - x_k||^2)/2\sigma_i^2}$$

$$Perp(P_i) = 2^{H(P_i)}$$

$$H(P_i) = -\sum_j p_{j|i} log_2 p_{j|i}$$ Entropy

Perplexity of P_i represents effective amount of neighbors for the point i

Set Perp(P_i) then t-SNE algorithm searches for the corresponding variance

Typical values for perplexity 5 to 50

t-SNE summary

Goal: preserves local neighbors

Gradient-based -> need multiple runs to see the best

Two parameters: #iteration, perplexity

EVALUATION

Evaluating a detection problem

- 4 possible scenarios

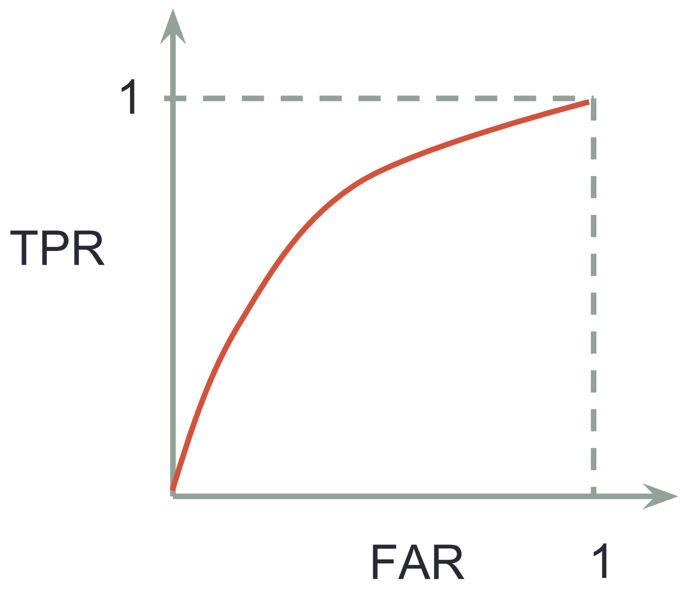
		Detector	
		Yes	No
Actual	Yes	True positive	False negative (Type II error)
	No	False Alarm (Type I error)	True negative

True positive + False negative = # of actual yes
False alarm + True negative = # of actual no

- False alarm and True positive carries all the information of the performance.

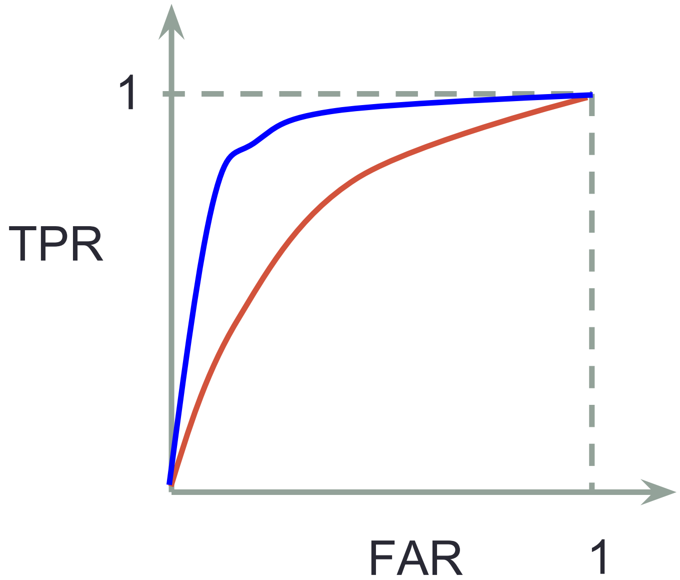
Receiver operating Characteristic (RoC) curve

- What if we change the threshold
- FA TP is a tradeoff
- Plot FA rate and TP rate as threshold changes



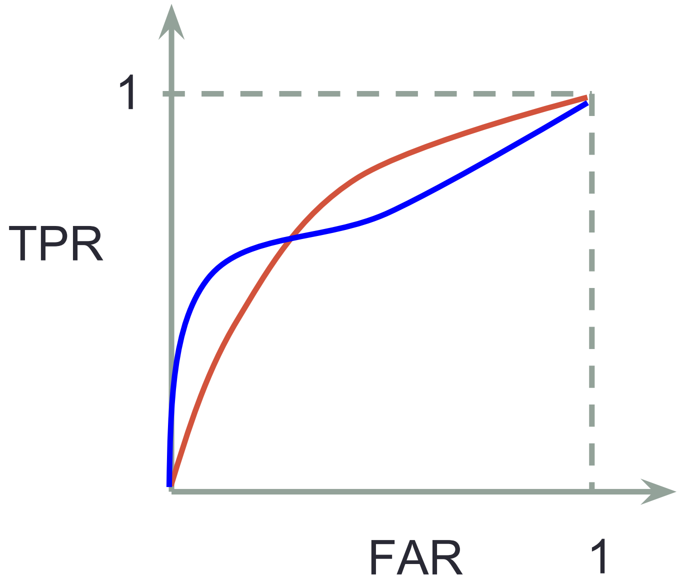
Comparing detectors

- Which is better?



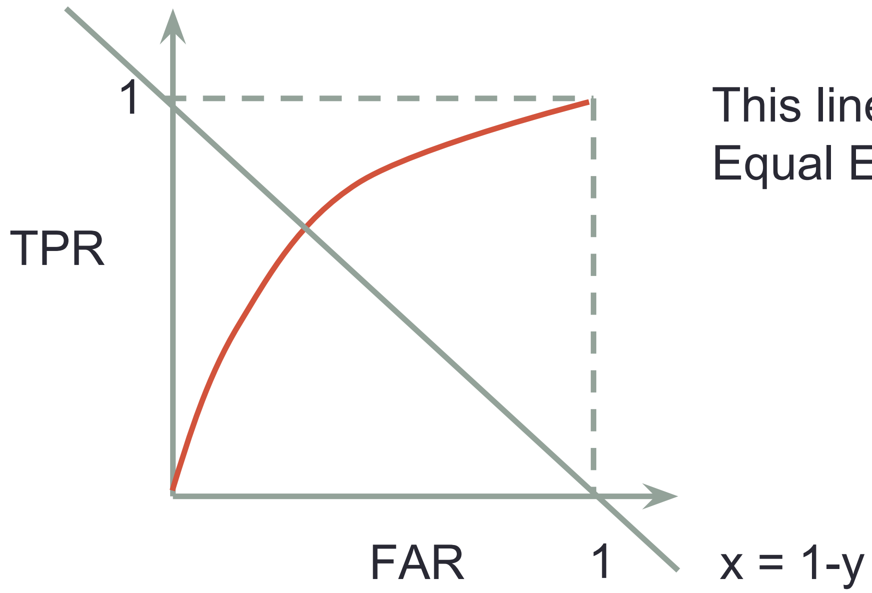
Comparing detectors

- Which is better?



Selecting the threshold

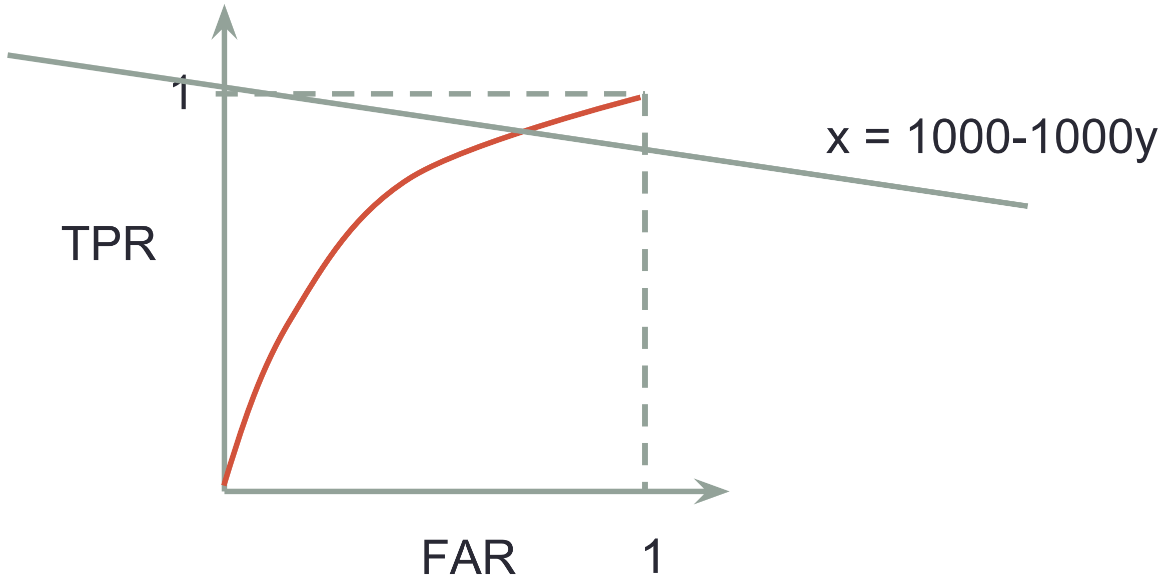
- Select based on the application
- Trade off between TP and FA. Know your application, know your users.
  - A miss is as bad as a false alarm   FAR = 1-TPR => x = 1-y



This line has a special name
Equal Error Rate (EER)

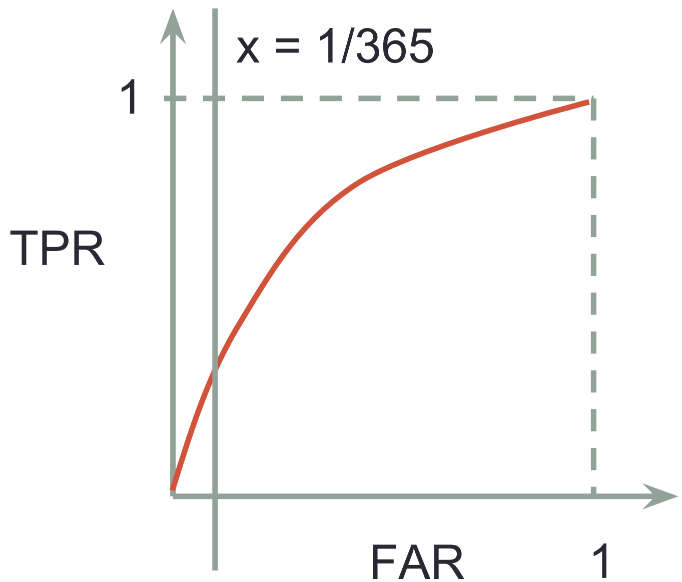
Selecting the threshold

- Select based on the application
- Trade off between TP and FA. Know your application, know your users. Is the application about safety?
  - A miss is 1000 times more costly than false alarm.
    - FAR = 1000(1-TPR) => x = 1000-1000y

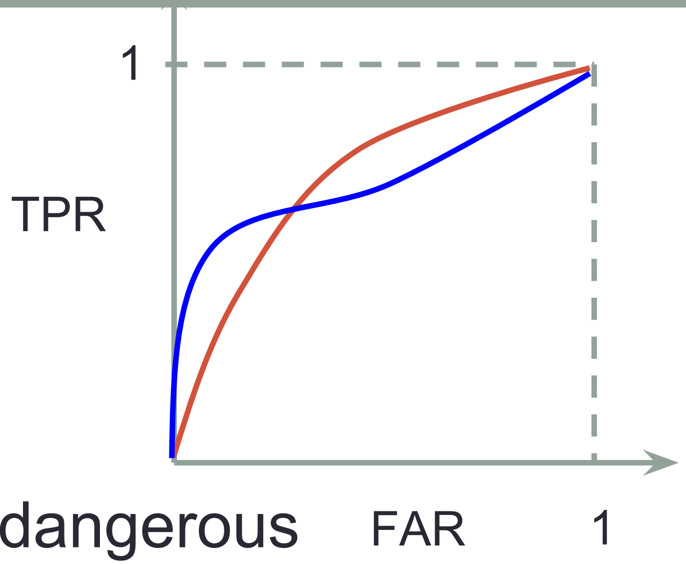


Selecting the threshold

- Select based on the application
- Trade off between TP and FA.
  - Regulation or hard threshold
  - Cannot exceed 1 False alarm per year
    - If 1 decision is made everyday, FAR = 1/365



Comparing detectors



- Which is better?
- You want to know whether a very dangerous virus is in a blood sample

Notes about RoC

- Ways to compress RoC to just a number for easier comparison -- use with care!!
  - EER
  - Area under the curve
  - F score
- Other similar curve - Detection Error Tradeoff (DET) curve
  - Plot False alarm vs Miss rate
- Can plot on log scale for clarity