DIMENSIONALITY REDUCTION AND VISUALIZATION

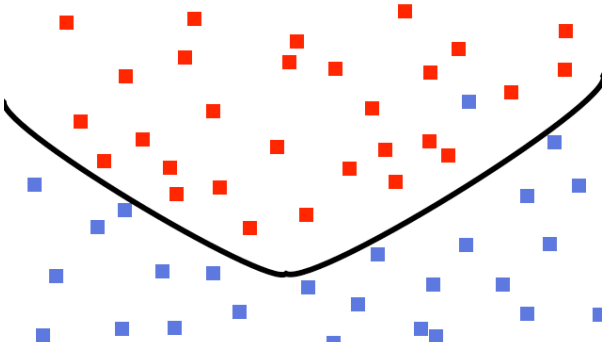
Loose ends from HW2

- Hyperparameters, bin size = 1000, 500, … ?
  - Tune on test set error rate
- Variance of a recognizer
  - Accuracy 100%? 98? 90? 80?
  - What's the mean and variance of the accuracy?
- A majority class baseline
  - Powerful if one class dominates
  - Recognizer becomes biased towards the majority class (the prior term)
  - Often happens in real life
  - How to deal with this?

Loose ends from HW2

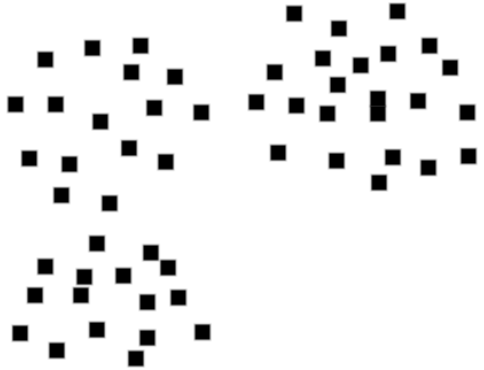
- Supervised learning
  - Learning with labels
  - Easy to use but hard to acquire
  - 10-15x to transcribe speech. 60x to label a self driving car training
- Unsupervised learning learning without labels
  - Usually we have a lot of these kinds of data
  - Hard to make use of them
- Reinforcement learning??

Three main types of learning

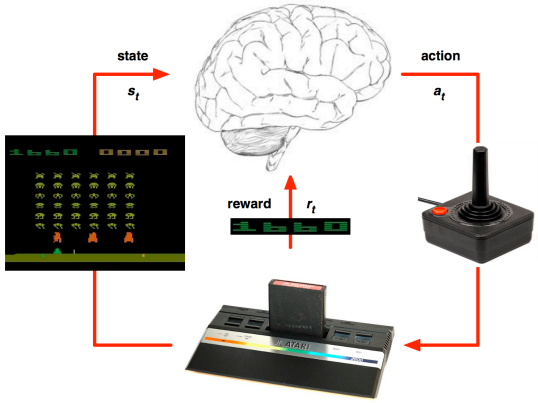


Supervised Learning

Reinforcement Learning



Unsupervised Learning



Loose ends from HW2

- What happens to P(x | hk), if there's no hk in the bin?
  - MLE estimates says P(a < x < b | hk) = 0
  - 0 probability for the entire term
  - Is this due to a bad sampling of the training set?
  - Can solve with MAP

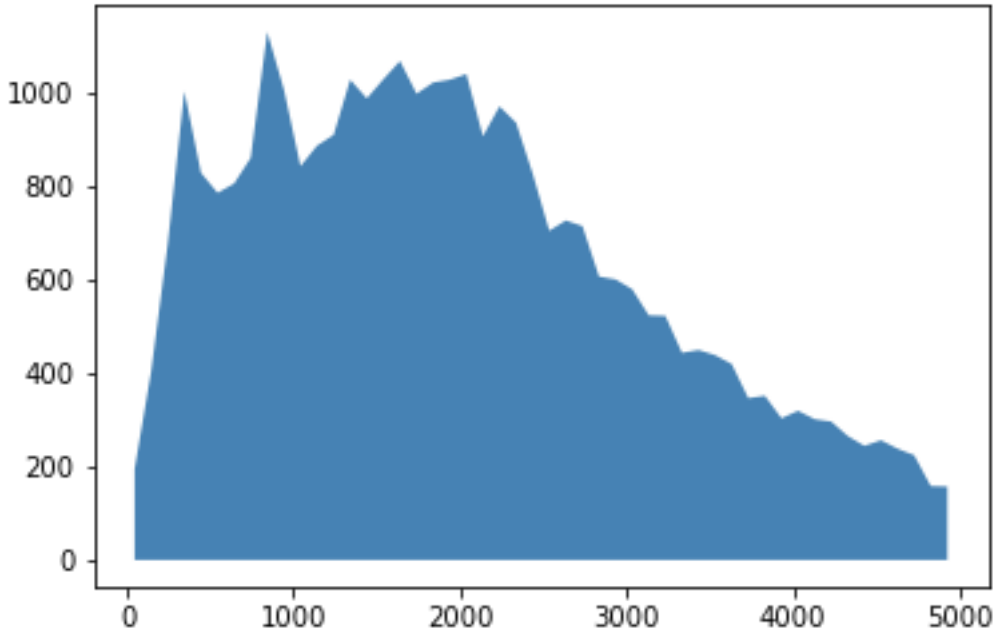
Map of a coin toss
β, α are prior hyperparameters

$$\theta = \frac{k + \alpha - 1}{n + \alpha + \beta - 2}$$

  - Use unsupervised data for the priors?

Loose ends from HW2

- Another method to combat zero counts is to use Gaussian mixture models
  - How to select the number of mixtures?
    - Maybe all these can be a course project



Loose ends from HW2

- Re-train using the full set for deployment (using the hyperparameters tuned on test)

Congratulations on your first attempt on re-implementing a research paper!

- Master thesis work

RESEARCH ARTICLE | OPEN ACCESS

Mining housekeeping genes with a Naive Bayes classifier

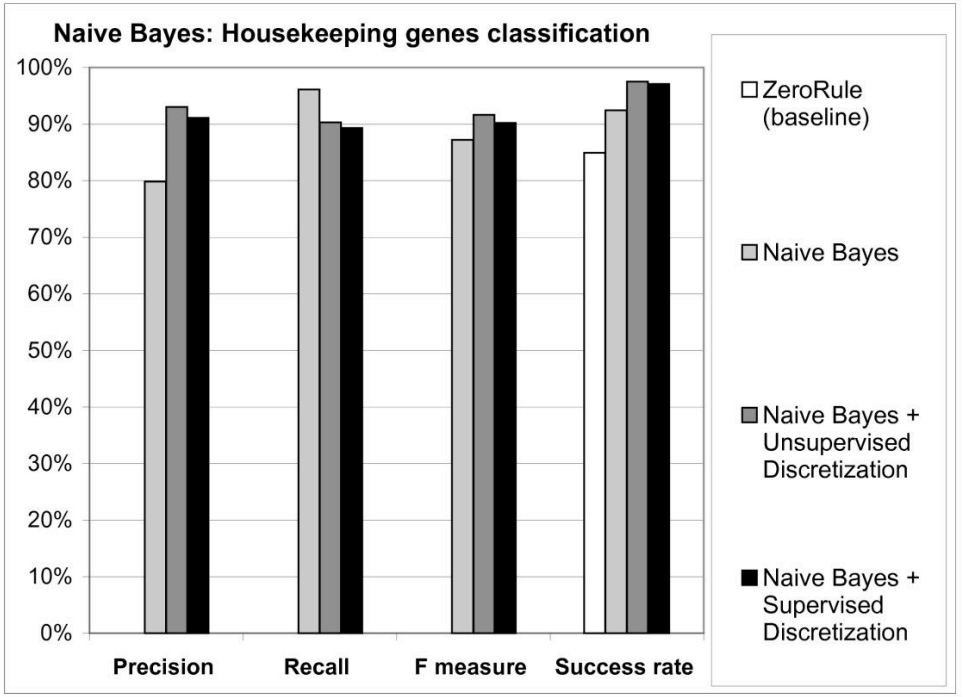
Luna De Ferrari and Stuart Aitken

BMC Genomics 2006 7:277 | https://doi.org/10.1186/1471-2164-7-277

Received: 28 February 2006 | Accepted: 30 October 2006 | Published: 30 October 2006

- Note that most of the hard work is on creating the dataset and feature engineering

(A)


Naive Bayes: Housekeeping genes classification
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Precision
Recall
F measure
Success rate
ZeroRule (baseline)
Naive Bayes
Naive Bayes + Unsupervised Discretization
Naive Bayes + Supervised Discretization

(B)

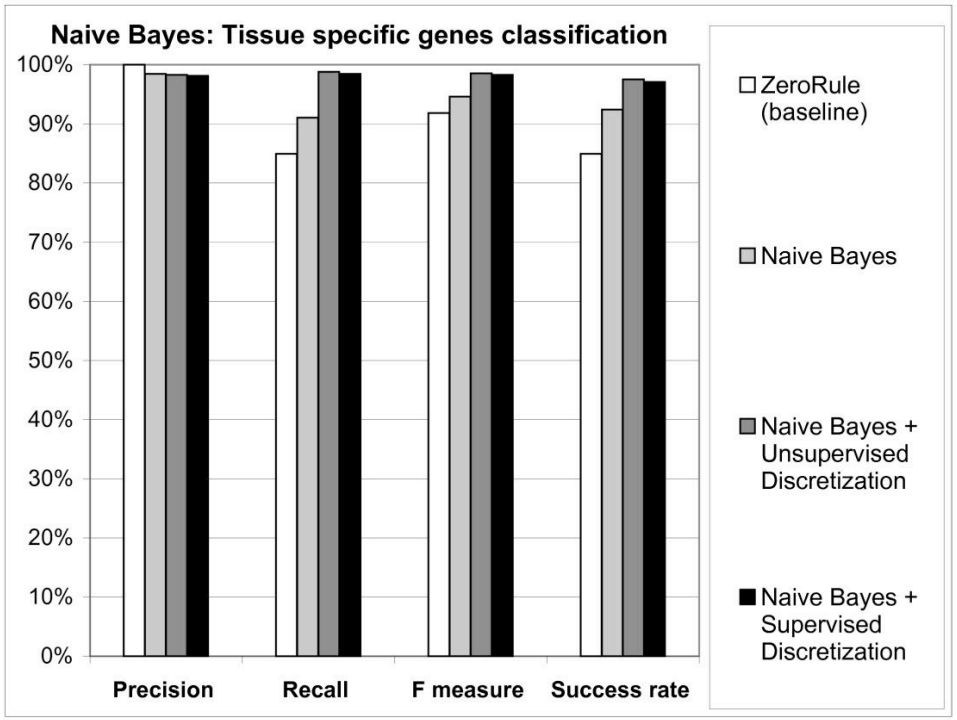
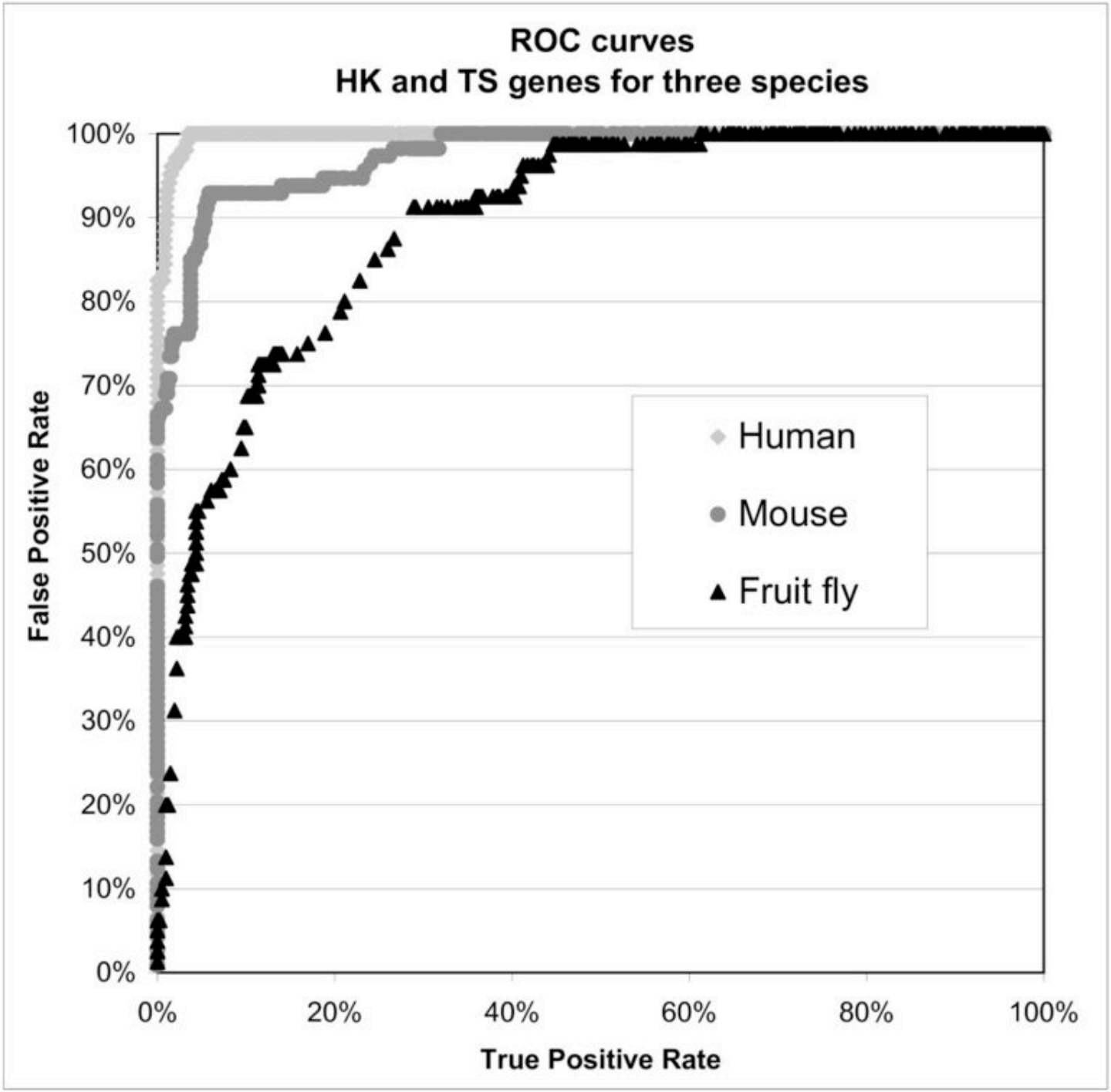

Naive Bayes: Tissue specific genes classification
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Precision
Recall
F measure
Success rate
ZeroRule (baseline)
Naive Bayes
Naive Bayes + Unsupervised Discretization
Naive Bayes + Supervised Discretization

Table 6
Naive Bayes classifier performance with unsupervised discretisation

		Precision	Recall	F Measure	Success Rate	Root Mean Squared Error
Human	HK	93.0	90.3	91.6	97.49 ± 0.14	0.13 ± 0.01
	TS	98.3	98.8	98.5		
Mouse	HK	83.3	79.6	81.4	92.57 ± 0.2	0.24 ± 0.01
	TS	94.7	95.8	95.2		
Fruit fly	HK	63.2	60.0	61.5	87.46 ± 0.34	0.32 ± 0.01
	TS	92.3	93.2	92.8		


ROC curves
HK and TS genes for three species
False Positive Rate
True Positive Rate
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
0%
20%
40%
60%
80%
100%
Human
Mouse
Fruit fly

Evaluating a detection problem

- 4 possible scenarios

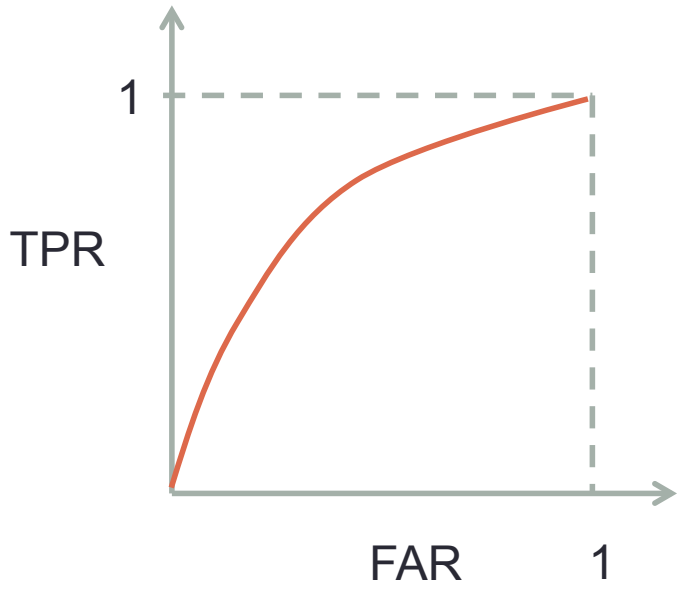
		Detector	
		Yes	No
Actual	Yes	True positive	False negative (Type II error)
	No	False Alarm (Type I error)	True negative

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.

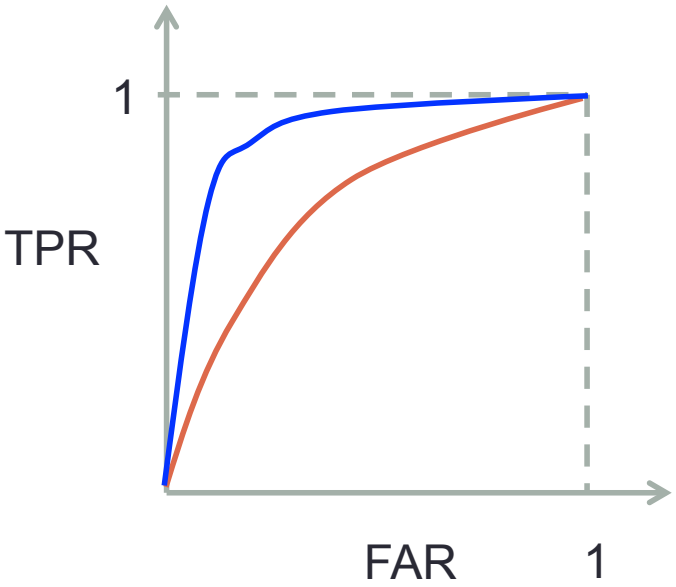
Receiver operation Characteristic (RoC) curve

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



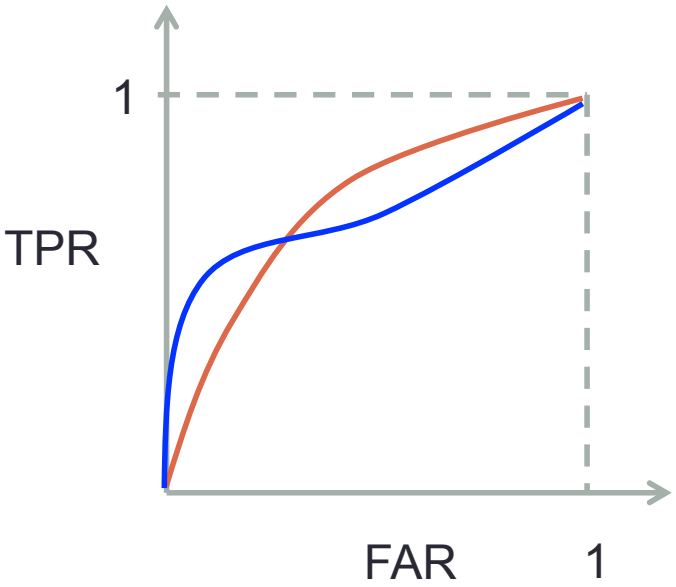
Comparing detectors

- Which is better?



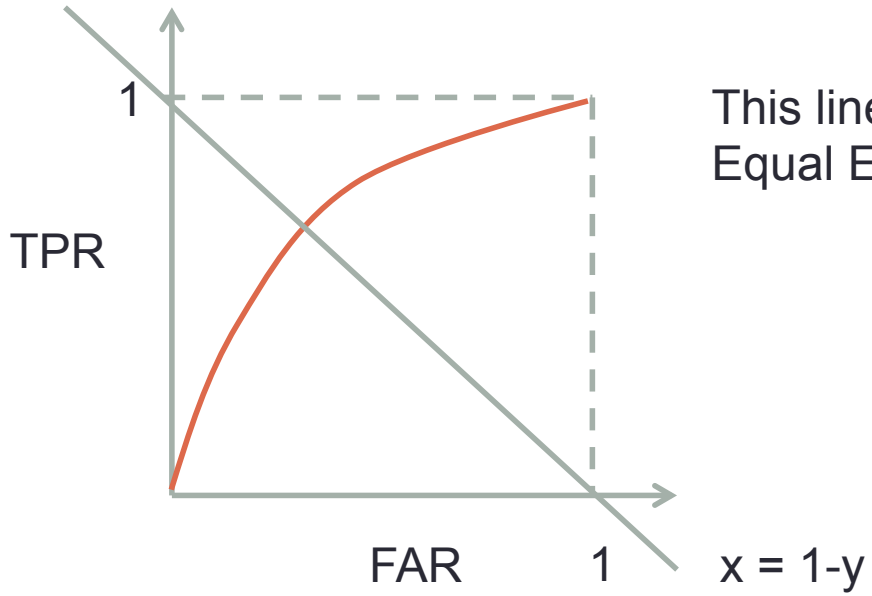
Comparing detectors

- Which is better?



Selecting the threshold

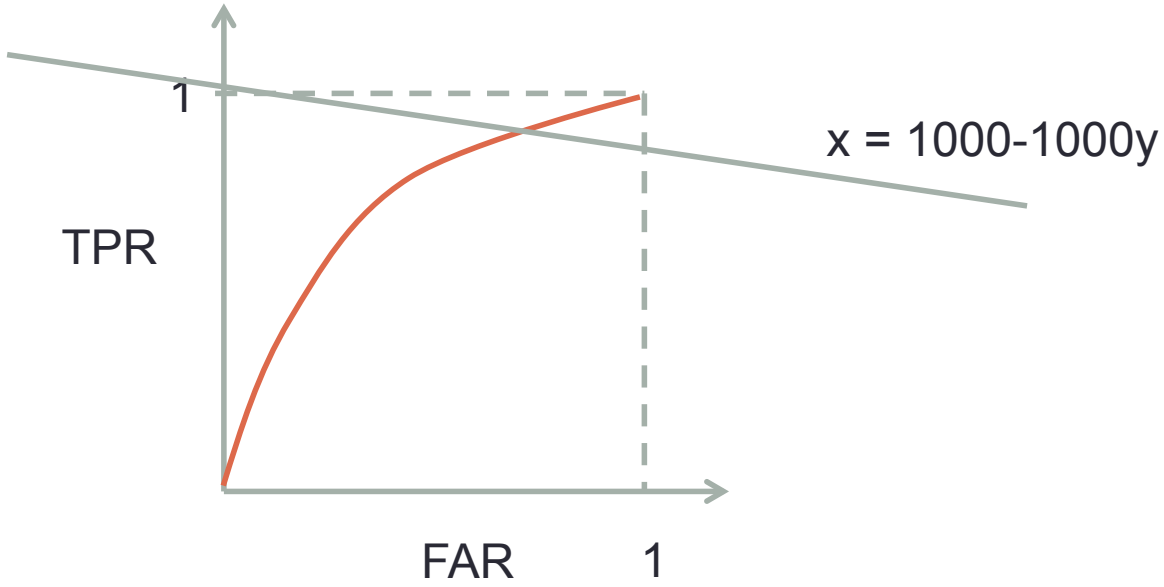
- Select based on the application
- Trade off between TP and FA. Know your application, know your users.
  - A miss is as bad as a false alarm FAR = 1-TPR => x = 1-y



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

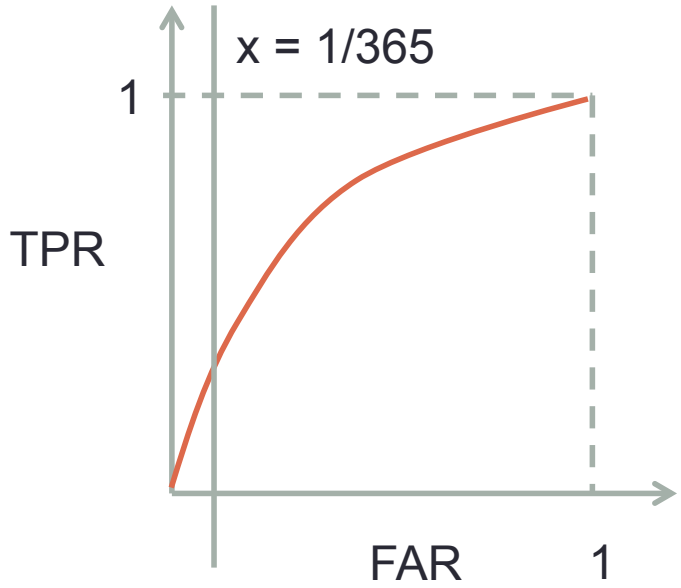
Selecting the threshold

- 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

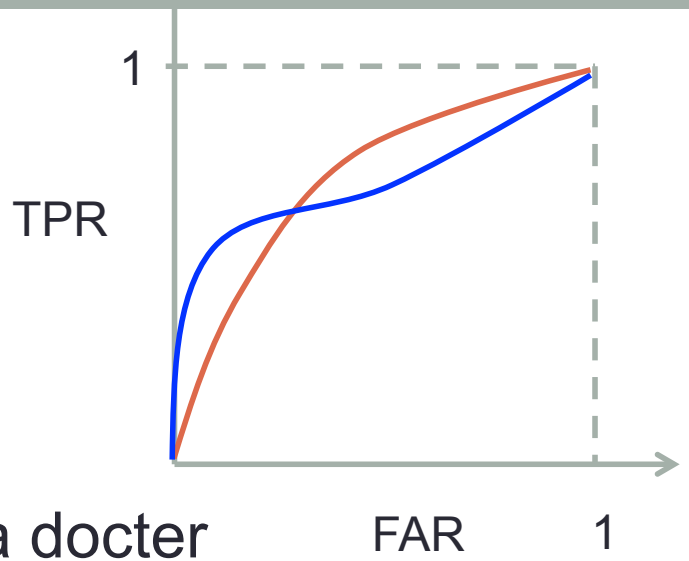


Selecting the threshold

- 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



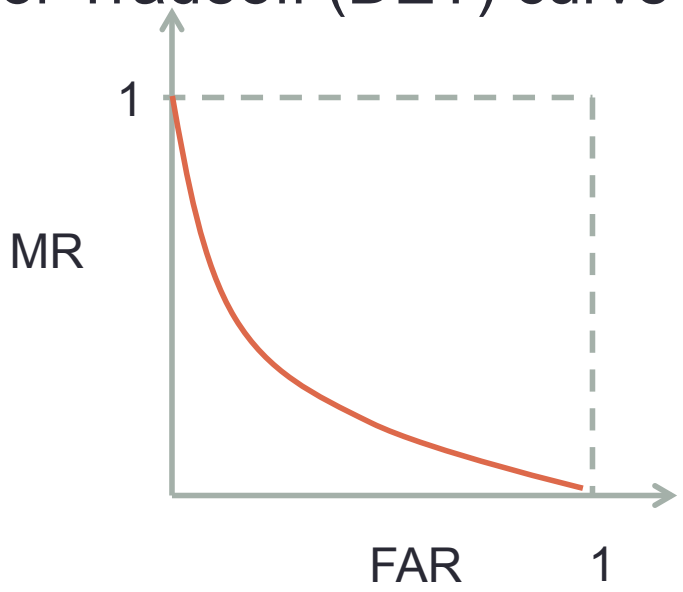
Comparing detectors



- Which is better?
- You want to give your findings to a docter to perform experiments to confirm that gene X is a housekeeping gene. You only want to identify a few new genes for your new drug.

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



Housekeeping genes data 10 years later

- ~30000 more genes experimented to be hk/not hk
- New hks
  - ENST00000209873
  - ENST00000248450
  - ENST00000320849
  - ENST00000261772
  - ENST00000230048
- New not hks
  - ENST00000352035
  - ENST00000301452
  - ENST00000330368
  - ENST00000355699
  - ENST00000315576

https://www.tau.ac.il/~elieis/HKG/

Housekeeping genes data 10 years later

- Some old training data got re-classified
  - hk -> not hk
  - ENST00000263574
  - ENST00000278756
  - ENST00000338167
- Importance of not trusting every data points
  - Noisy labels
  - overfitting

DIMENSIONALITY REDUCTION AND VISUALIZATION

Mixture models

$$p(x) = \sum_k p(k) p_k(x)$$

- A mixture of models from the same distributions (but with different parameters)
- Different mixtures can come from different sub-class
  - Cat class
    - Siamese cats
    - Persian cats
- p(k) is usually categorical (discrete classes)
- Usually the exact class for a sample point is unknown.
  - Latent variable

EM on GMM

- E-step
  - Set soft labels: $w_{n,j}$ = probability that nth sample comes from jth mixture p
  - Using Bayes rule
    - p(k|x ; μ, σ, ϕ) = p(x|k ; μ, σ, ϕ) p(k; μ, σ, ϕ) / p(x; μ, σ, ϕ)
    - p(k|x ; μ, σ, ϕ) α p(x|k ; μ, σ, ϕ) p(k; ϕ)

$$p(k_n = j|x_n; \phi, \mu, \Sigma) = \frac{p(x_n; \mu_j, \sigma_j)p(k_n = j; \phi)}{\Sigma_l p(x_n; \mu_l, \sigma_l)p(k_n = l; \phi)}$$

EM on GMM

- M-step (soft labels)

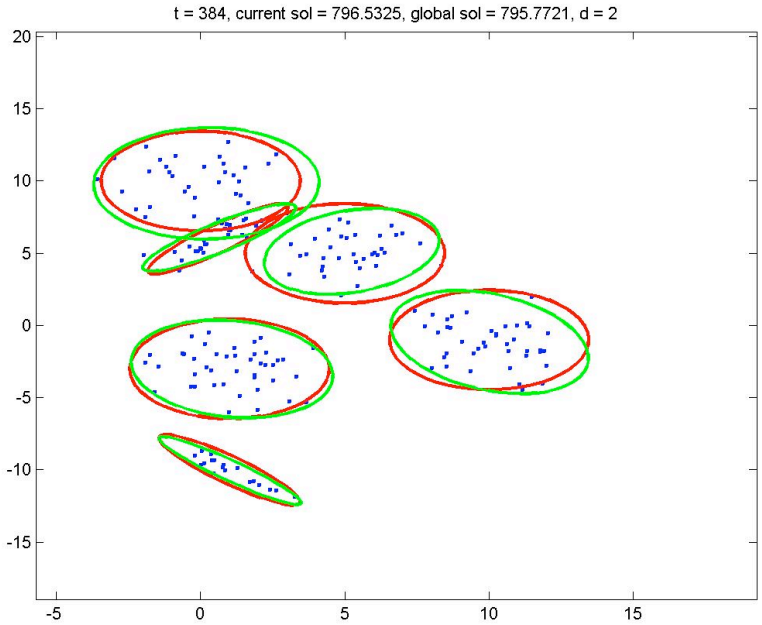
$$\phi_j = \frac{1}{N}\Sigma_{n=1}^{N} w_{n,j}$$

$$\mu_j = \frac{\Sigma_{n=1}^{N} w_{n,j} x_n}{\Sigma_{n=1}^{N} w_{n,j}}$$

$$\sigma_j^2 = \frac{\Sigma_{n=1}^{N} w_{n,j} (x_n - \mu_j)^2}{\Sigma_{n=1}^{N} w_{n,j}}$$

EM/GMM notes

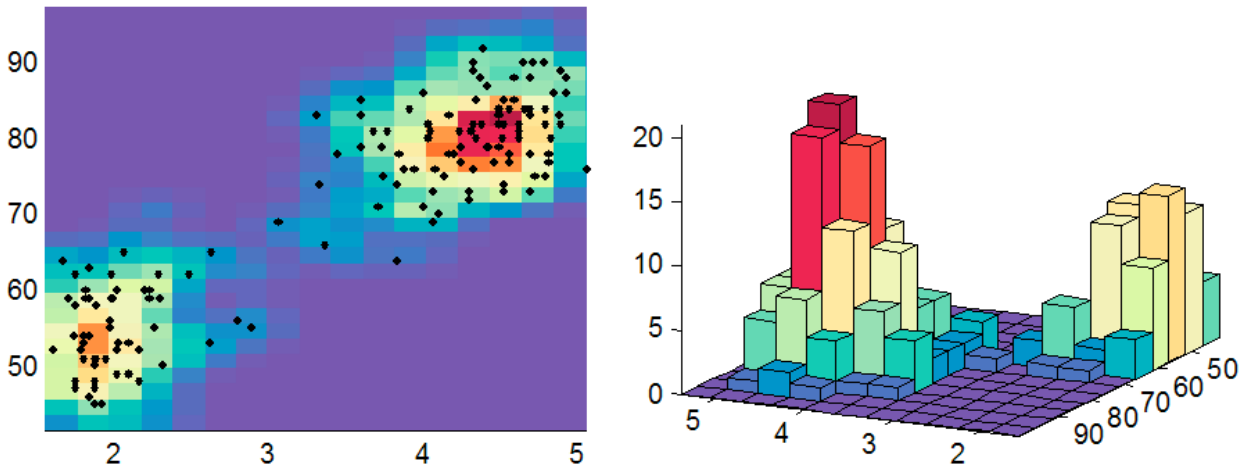
- Converges to local maxima (maximizing likelihood)
  - Just like k-means, need to try different initialization points
- What if it's a multivariate Gaussian?
  - The grid search gets harder as the number of number of dimension grows



https://www.mathworks.com/matlabcentral/fileexchange/7055-multivariate-gaussian-mixture-model-optimization-by-cross-entropy

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?



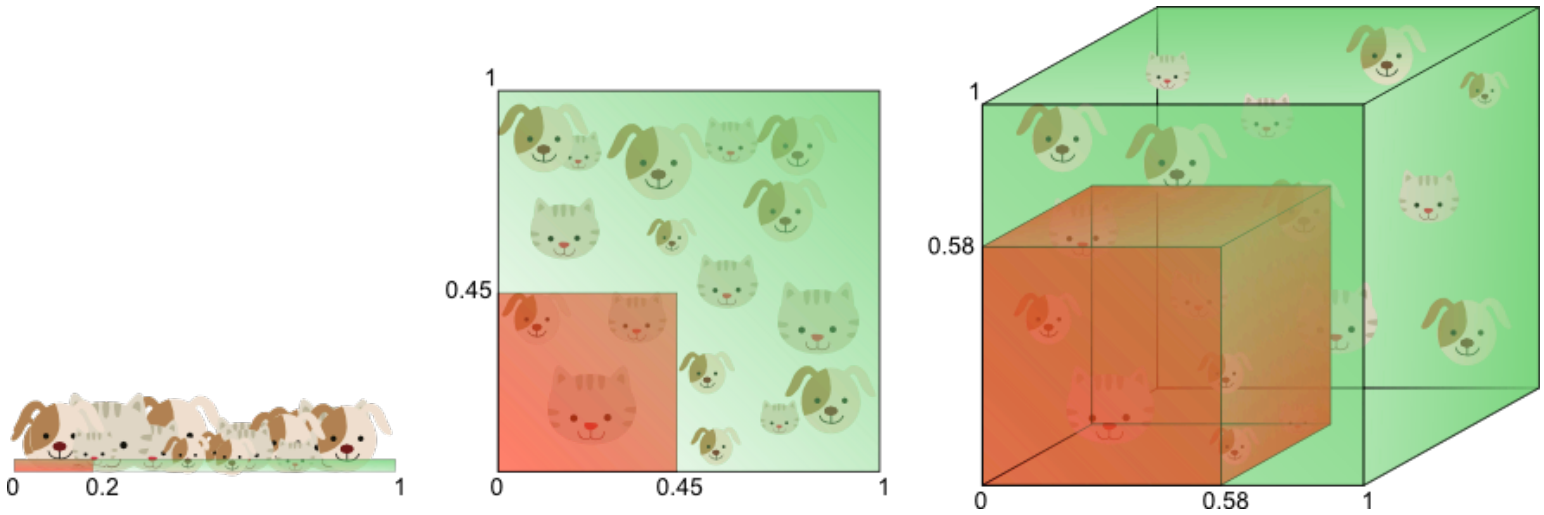
https://www.mathworks.com/matlabcentral/fileexchange/45325-efficient-2d-histogram--no-toolboxes-needed

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



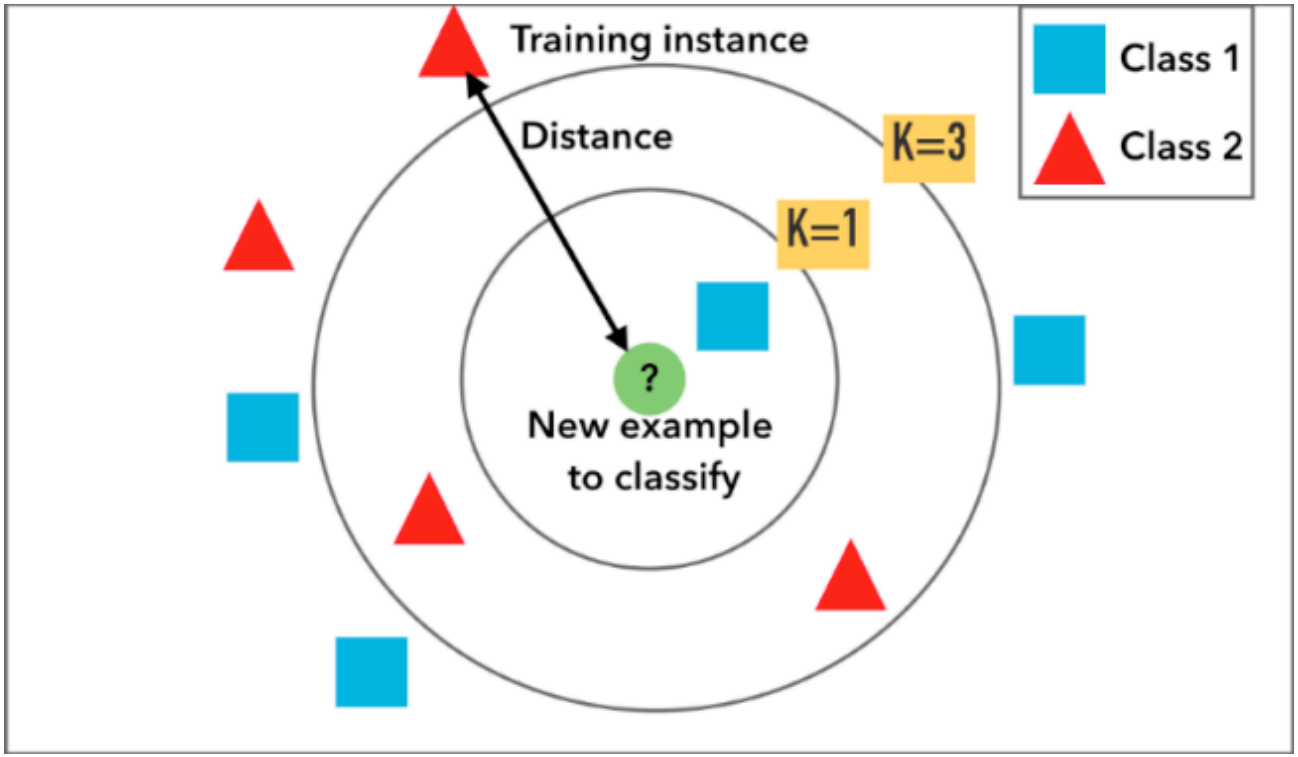
http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/

Nearest Neighbor Classifier

- The thing most similar to the test data must be of the same class

Find the nearest training data, and use that label

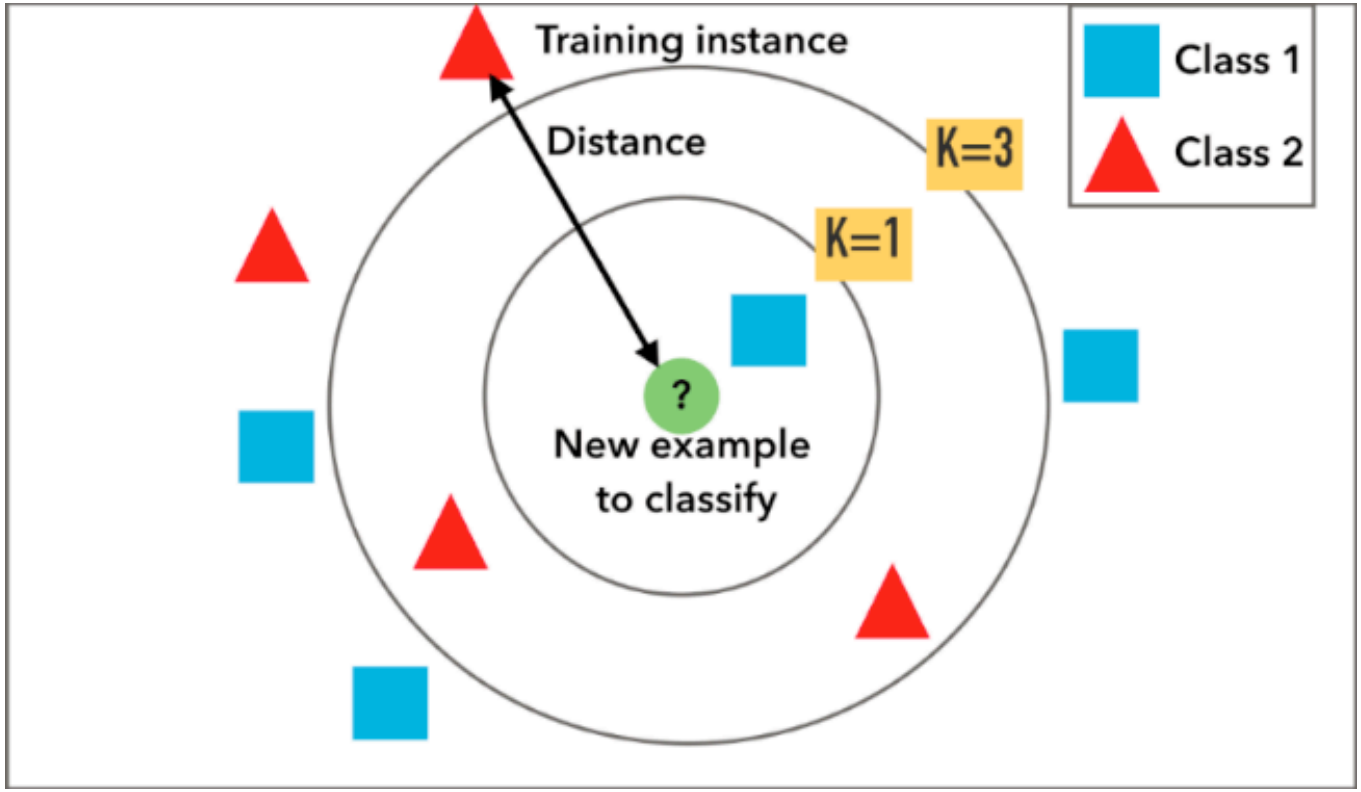
- Use “distance” as a measure of closeness.
- Can use other kind of distance besides Euclidean



https://arifuzzamanfaisal.com/k-nearest-neighbor-regression/

K-Nearest Neighbor Classifier

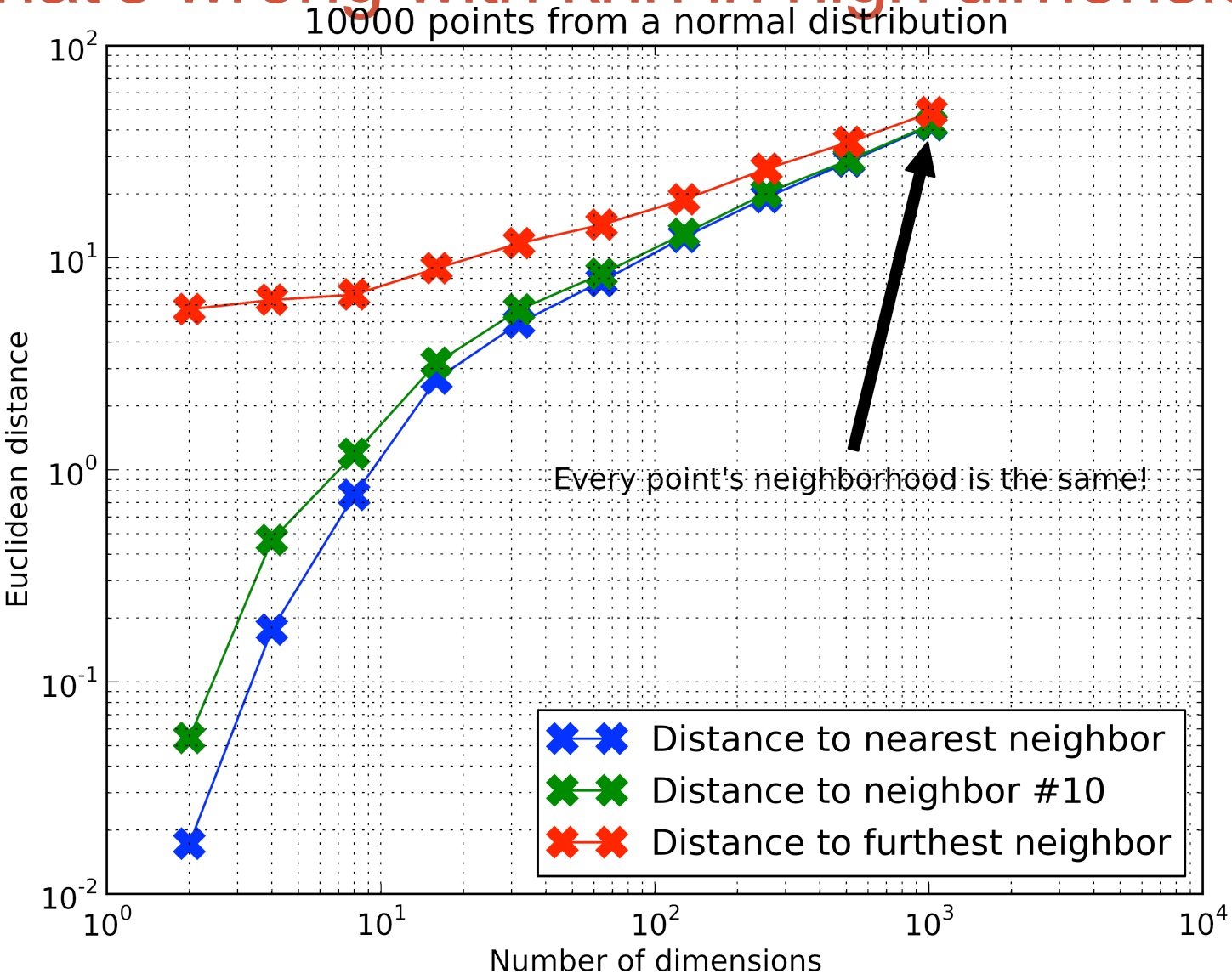
- Nearest neighbor is susceptible to label noise
- Use the k-nearest neighbors as the classification decision
  - Use majority vote



K-Nearest Neighbor Classifier

- It's actually VERY powerful!
  - Keeps all training data – Other methods usually smears the input together (to reduce complexity)
    - Cons: computing the nearest neighbor is costly with lots of data points and higher compute in higher dimensions
    - Workarounds: Locality sensitive hashing, kd trees
- Still useful even today
  - Finding the closest word to a vector representation

What's wrong with knn in high dimension?



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

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 vs Feature transform

- Keep original features
  - Useful for when the user wants to know which feature matters
    - But, correlation does not imply causation…
- New features (a combination of old features)
- Usually more powerful
  - Captures correlation between features

Feature selection

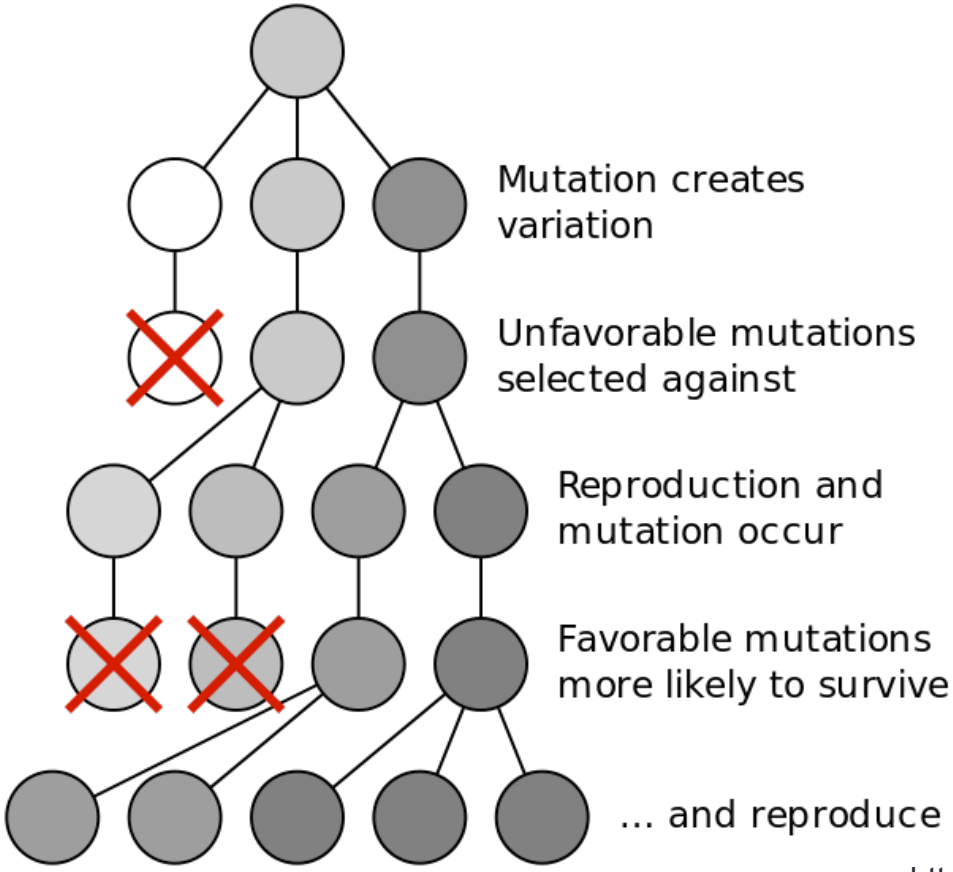
- Hackathon level (time limit days-a week)
  - Drop missing features
  - Low variance rows
    - A feature that is a constant is useless. Tricky in practice
  - Forward or backward feature elimination
    - Greedy algorithm: create a simple classifier with n-1 features, n times. Find which one has the best accuracy, drop that feature. Repeat.

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

Genetic Algorithm

- A method based inspired by natural selection
  - No theoretical guarantees but often work



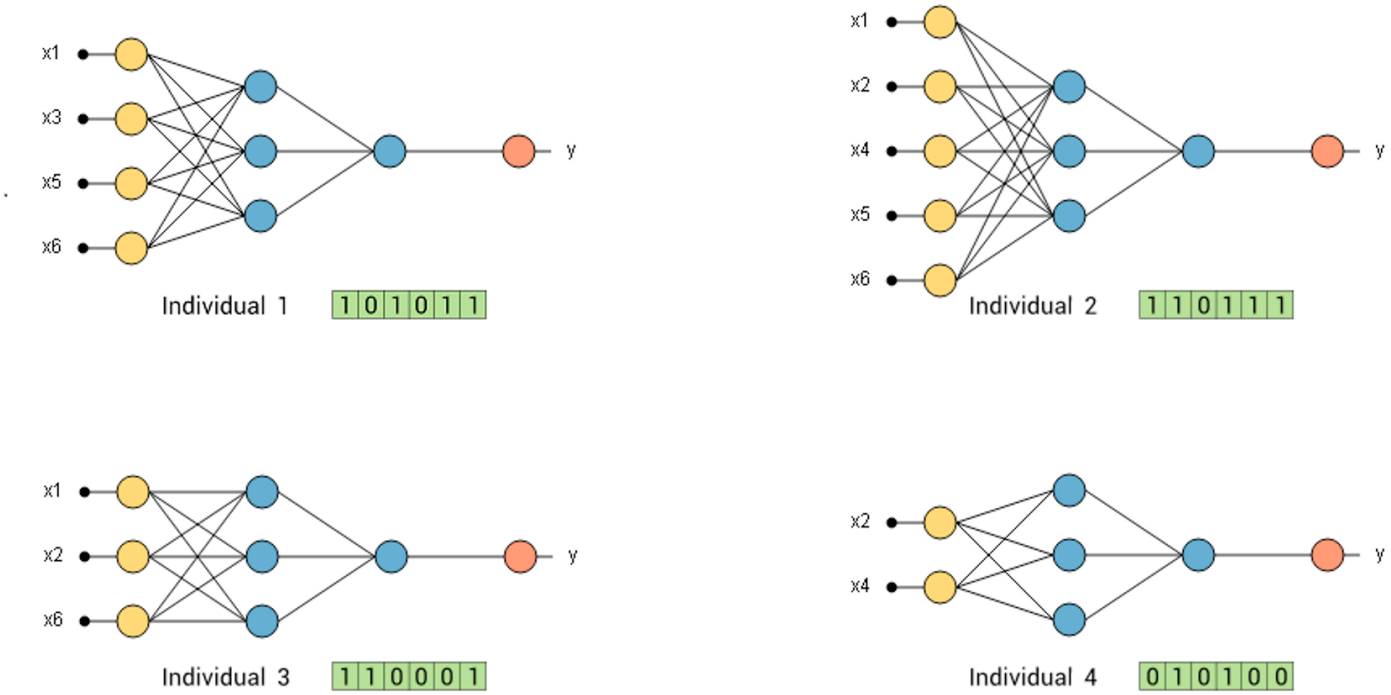
https://elitedatascience.com/dimensionality-reduction-algorithms

Genetic Algorithm

- Initialization
  - Create N classifiers, each using different subset of features
- Selection process
  - Rank the N classifiers according to some criterion, kill the lower half
- Crossover
  - The remaining classifier breeds offsprings by selecting traits from the parents
- Mutation
  - The offsprings can have mutations by random in order to generate diversity
- Repeat till satisfied

Initialization

- Create N classifiers
- Randomly select a subset of features to use



Examples from https://www.neuraldesigner.com/blog/genetic_algorithms_for_feature_selection

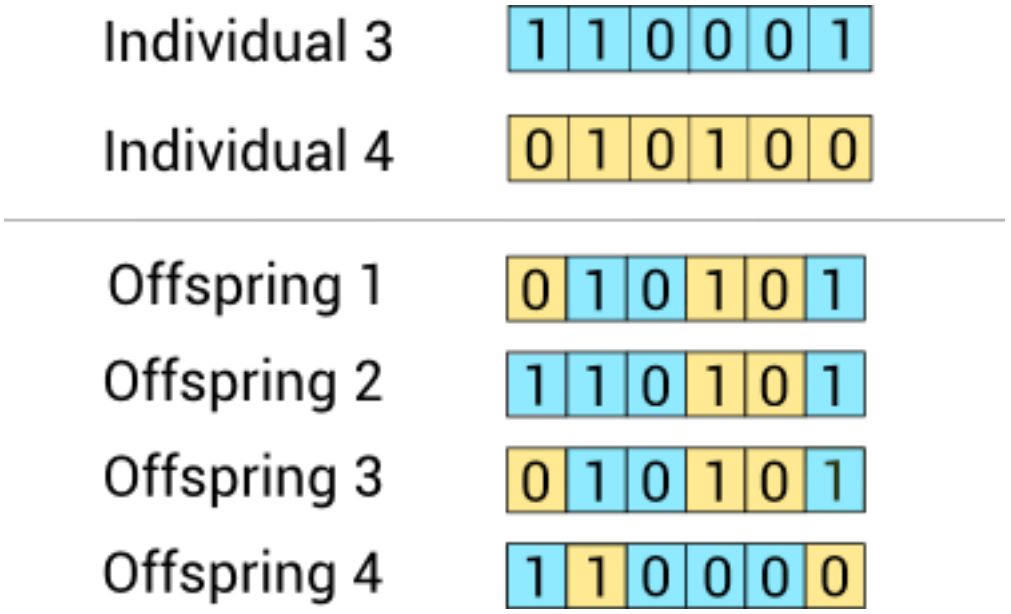
Selection process

- Score the classifiers and kill the lower half (the amount to kill is also a parameter)

	Selection error	Rank
~~Individual 1~~	0.9	1
Individual 2	0.6	3
~~Individual 3~~	0.7	2
Individual 4	0.5	4

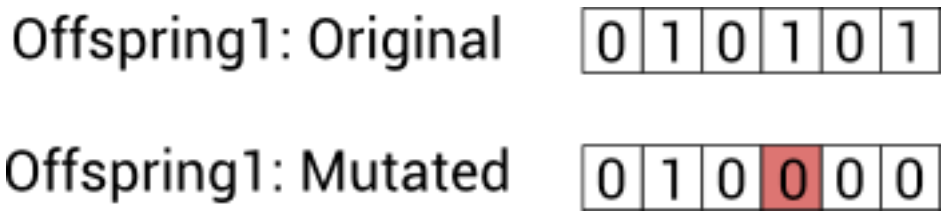
Crossover

- Breed offsprings by randomly select genes from parents



Mutation

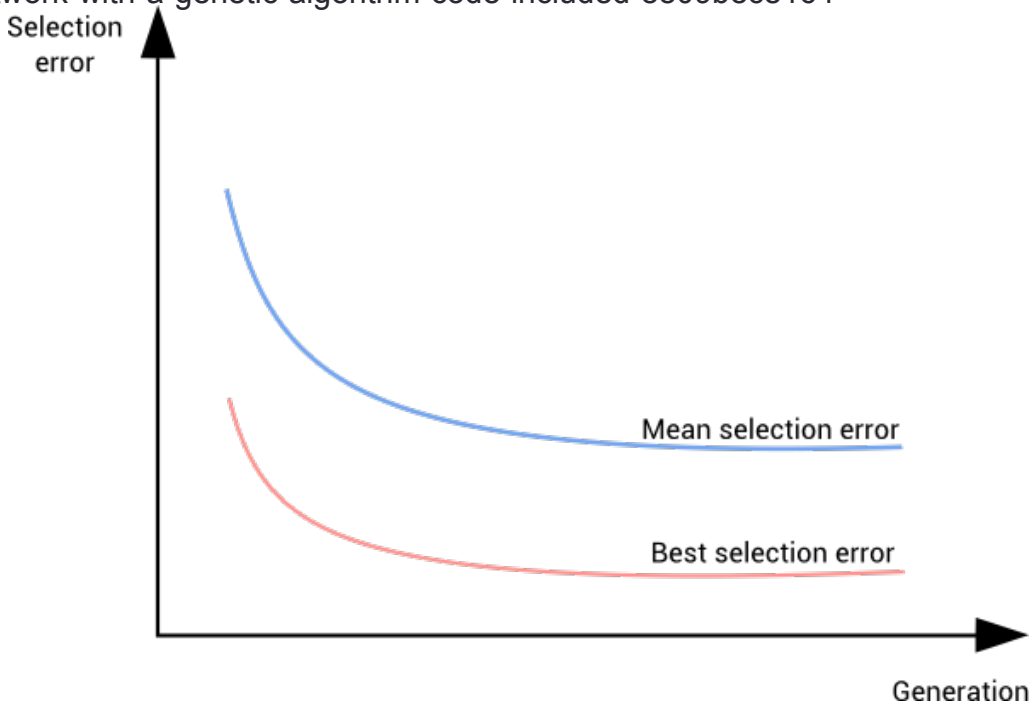
- Offspring can mutate with some probability to introduce diversity
- Mutation rate is usually 1/k where k is the number of features.
  - On average you mutate once per individual



Performance

- Usually performs well. The general population usually gets better (mean). The best performing (individual) also gets better after each generation
- Can be use to tune neural networks!

https://blog.coast.ai/lets-evolve-a-neural-network-with-a-genetic-algorithm-code-included-8809bece164

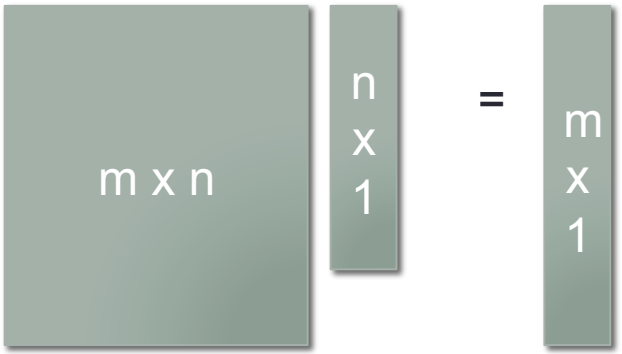


Feature transformation

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

Linear Algebra Review

- Think Sets and Functions, rather than manipulation of number arrays/rectangles



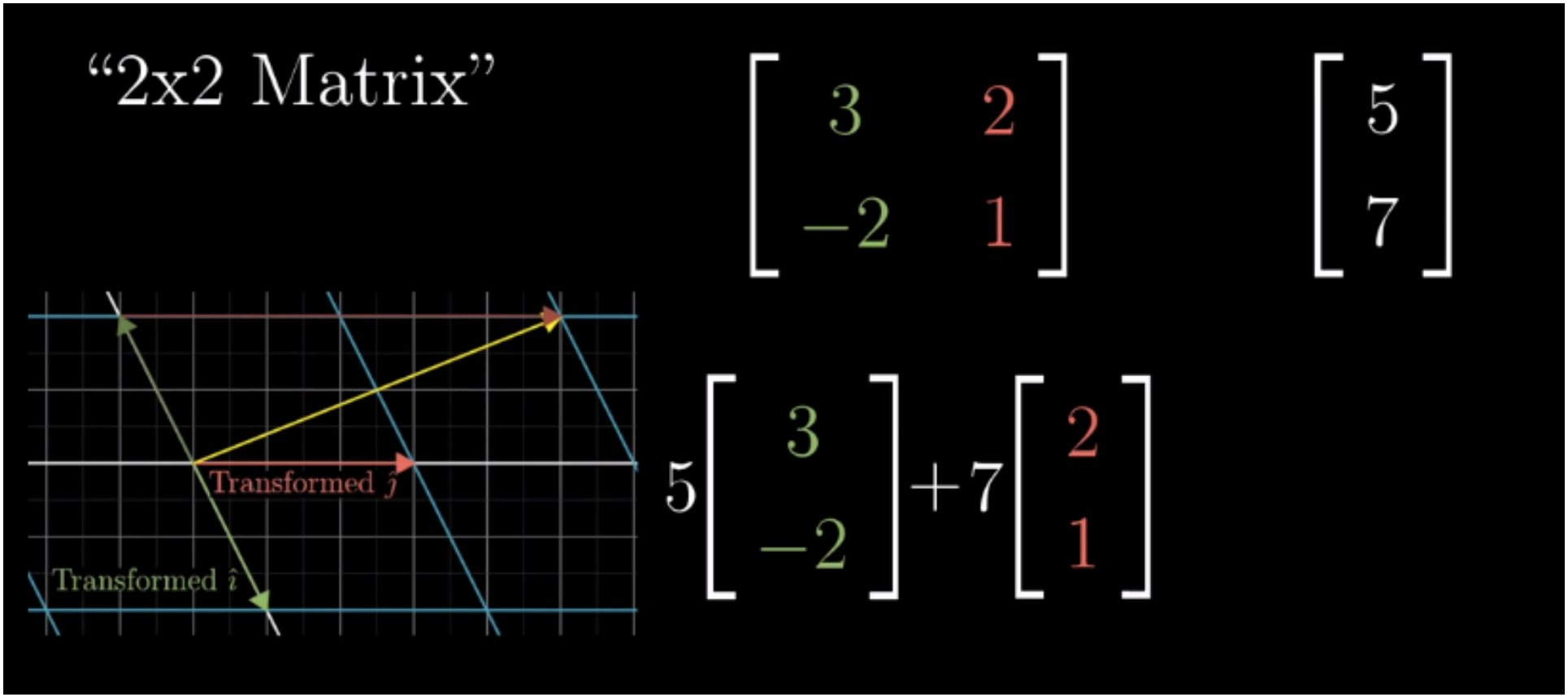
$f: R^n \rightarrow R^m$

$f(\mathbf{x})$

https://www.linkedin.com/pulse/key-machine-learning-prereq-viewing-linear-algebra-through-ashwin-rao

Linear Algebra Review

- Matrix as a sequence of column vectors



https://www.youtube.com/watch?v=kYB8IZa5AuE&list=PLZHQObOWTQDPD3MizzM2xVFitgF8hE_ab&index=4

Linear Algebra Review

- Understand Matrix Factorizations as Compositions of "Simple" Functions:

$$h(\mathbf{x}) = k(\, d(\mathbf{x})\,)$$

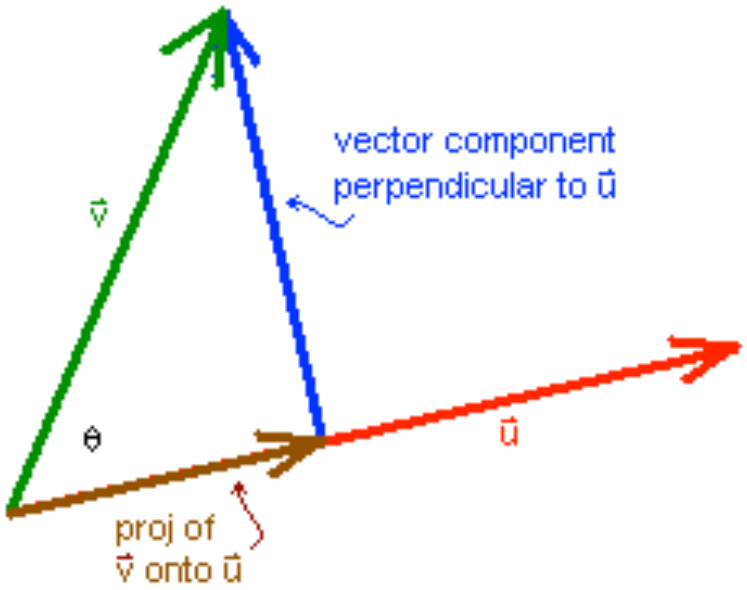
Linear Algebra Review

- View Eigendecomposition (ED) and Singular Value Decomposition (SVD) as rotations and stretches

D has eigenvalues on the diagonal
Q is a matrix where i^{th} column is the q^{th} eigenvector

Linear Algebra Review

- Projection as a change of basis
- Change basis from x,y coordinates to be on u



$u^t v$

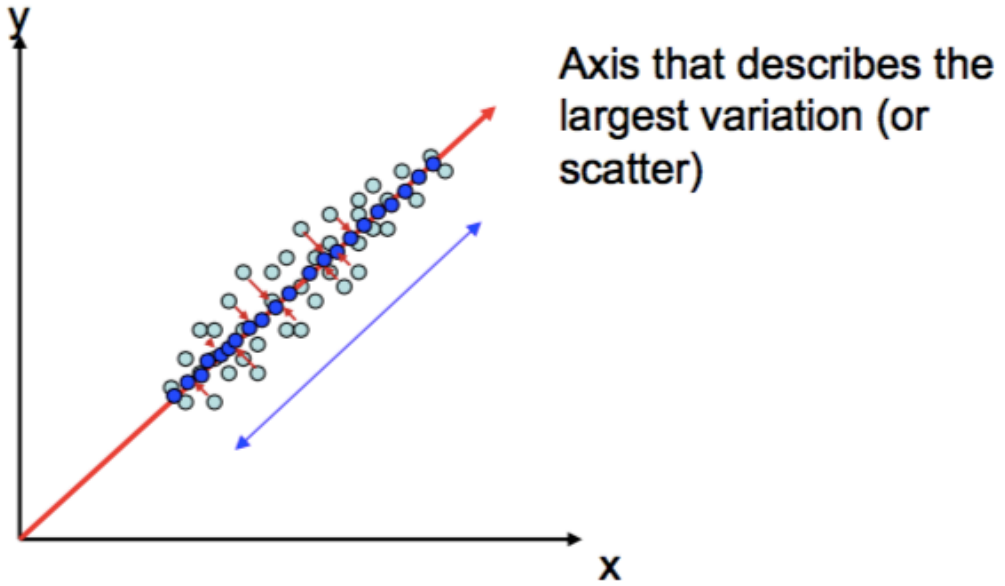
Positive semi-definite and eigendecomposition

- Covariance matrix is positive semi-definite and symmetric.
- Eigenvalues are always positive or null, eigenvalues and vectors are real values, eigenvectors are mutually orthogonal.

- $q_i^t q_j = 0$ for i != j

What is PCA?

- We want to reduce the dimensionality but keep useful information
  - What is useful information? Variation
- We want to find a projection (a transformation) that describe maximum variation



Formulation

- Maximize the variance after projection ie
  - argmax $Var(w^t x)$
- Subject to w is a unit vector

- $\Sigma w = \lambda w$ <- eigenvector

Trace properties

1. tr (a) = a
2. trA = trA^T
3. tr(A+B) = trA+trB
4. tr(aA) = atr(A)

5. $$\nabla_A trAB = B^T$$
6. $$\nabla_{A^T} f(A) = (\nabla_A f(A))^T$$
7. $$\nabla_A trABA^TC = CAB + C^TAB^T$$
8. $$\nabla_{A^T} trABA^TC = B^TA^TC^T + BA^TC$$

So we got to eigenvectors

- A dxd covariance matrix has d eigen vectors/values pair. Do we use all of them?
- Which pair to use?

Selecting eigenvectors

- Remember the variance of projected data is

$$\omega^T \Sigma \omega. \quad (1)$$

- And our solution yielded

$$\Sigma\omega = \lambda\omega \quad (2)$$

- Plug (2) in (1) and we get

$$\begin{aligned} \text{projected variance} &= \omega^T \Sigma \omega = \omega^T \lambda\omega \\ &= \lambda\, \omega^T \omega \qquad (\text{remember } \|\omega\|=1) \\ &= \lambda \end{aligned}$$

PCA

- The direction vector captures the variance corresponding to the eigenvalue
- So we want the higher eigenvalues
  - How many?

Matrix rank

- A square dxd matrix has full rank (e.g. rank d) if the columns are linearly independent.
- The number of linearly independent columns is the rank of the matrix

- A covariance matrix of size dxd will have have at most N-1 rank where N is the number of training samples
  - 640x640 images = ~400000 dimensions
  - 1000 training images
  - The covariance matrix will be at most rank 999. The missing rank is because of the mean.

PCA

- The direction vector captures the variance corresponding to the eigenvalue
- So we want the higher eigenvalues
- Take the eigenvalues with non-zero eigenvalues (at most N-1 non-zero eigenvalues)

Basis decomposition

- Let's consider our projection w_i which is the eigenvectors to be a basis vector v_i
- We can represent any vector as a sum of basis vectors as follows:

$$\mathbf{x} = \sum_{i=1}^{N} p_i \mathbf{v}_i = p_1 \begin{bmatrix} | \\ \mathbf{v}_1 \\ | \end{bmatrix} + p_2 \begin{bmatrix} | \\ \mathbf{v}_2 \\ | \end{bmatrix} + .. + p_n \begin{bmatrix} | \\ \mathbf{v}_n \\ | \end{bmatrix} = \mathbf{V}\mathbf{p}$$

Finding the weights

$$\mathbf{x} = \sum_{i=1}^{N} p_i \mathbf{v}_i = p_1 \begin{bmatrix} | \\ \mathbf{v}_1 \\ | \end{bmatrix} + p_2 \begin{bmatrix} | \\ \mathbf{v}_2 \\ | \end{bmatrix} + .. + p_n \begin{bmatrix} | \\ \mathbf{v}_n \\ | \end{bmatrix} = \mathbf{V}\mathbf{p}$$

- If v_i are orthogonal, the projection of x onto v_i gives p_i

$$\mathbf{V}^{\mathbf{T}}\mathbf{x} = \begin{bmatrix} - & \mathbf{v}_1 & - \\ - & \mathbf{v}_2 & - \\ - & \mathbf{v}_3 & - \end{bmatrix} \begin{bmatrix} | \\ \mathbf{x} \\ | \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

Means

- In PCA, we model variance. (Variation around the mean)
- In our projection we need to remove the mean

$$\mathbf{p} = \mathbf{V}^{\mathbf{T}}(\mathbf{x} - \mathbf{m})$$

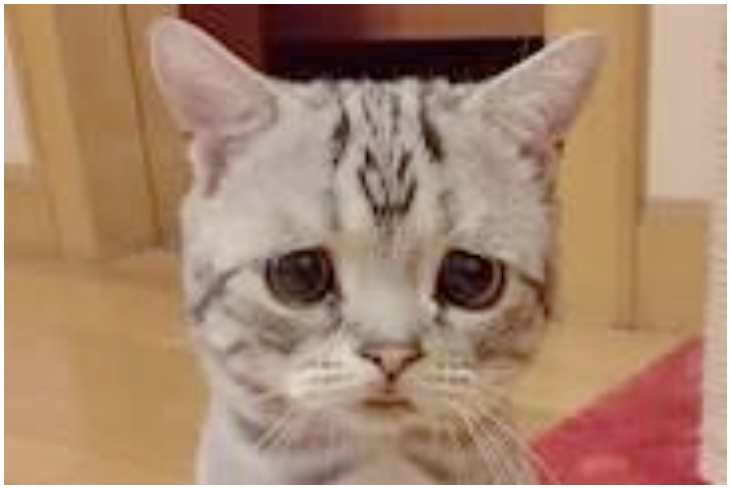
- The mean is the mean of all your training data
- If we want to reconstruct the data we need to add back the mean

$$\mathbf{x} = \sum_{i=1}^{N} p_i \mathbf{v}_i + \mathbf{m} = p_1 \begin{bmatrix} | \\ \mathbf{v}_1 \\ | \end{bmatrix} + p_2 \begin{bmatrix} | \\ \mathbf{v}_2 \\ | \end{bmatrix} + .. + p_n \begin{bmatrix} | \\ \mathbf{v}_n \\ | \end{bmatrix} + \mathbf{m} = \mathbf{V}\mathbf{p} + \mathbf{m}$$

Practical issues

- If your data has different magnitudes in different dimensions, normalize each dimension before PCA

- If we have 640x640 images = ~400000 dimensions.
- What is the size of the covariance matrix?



Practical issues

- You have N training examples.
- For the case where N << 400000, we only have N-1 eigen values we care about anyway

Gram Matrix

$$\Sigma = E(x - \mu)(x - \mu)^T = XX^T$$

Covariance matrix is the outer-product of the input matrix

Must solve $\Sigma v = \lambda v$

$XX^T v = \lambda v$ (pre-mult by X^T) (1)

$X^T XX^T v = \lambda X^T v$ ($v' = X^T v$) (2)

Solve eigenvalue problem $X^T X v' = \lambda v'$

- $X^T X$ is a gram of inner-product matrix. Its size is NxN where N is the number of data samples.

But how to get v from v'?

- From previous slide, equation (1) and (2)
  - $XX^Tv = \lambda v$ (1)
  - $v' = X^Tv$ (2)
- Substitute (2) into (1)
  - $Xv' = \lambda v$

- Thus, v = Xv'. We don't care about the scaling term because we will always scale the eigenvector so that it is orthonormal i.e. ||v|| = 1.

How many eigenvectors?

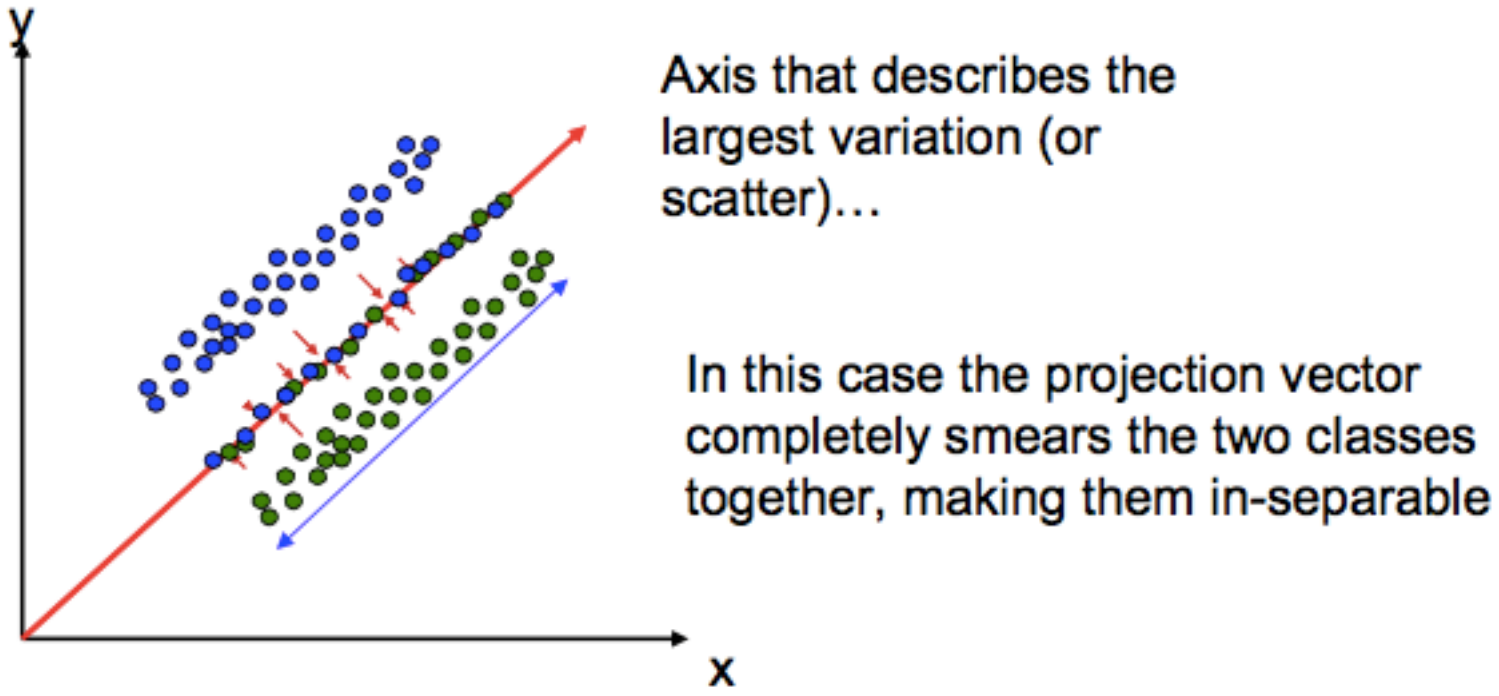
- Select based on amount of variance explained
  - Sum of eigenvalues exceeds some percent of total
- Reconstruction error

$$\mathbf{x} = \sum_{i=1}^{N} p_i \mathbf{v}_i = p_1 \begin{bmatrix} | \\ \mathbf{v}_1 \\ | \end{bmatrix} + p_2 \begin{bmatrix} | \\ \mathbf{v}_2 \\ | \end{bmatrix} + .. + p_n \begin{bmatrix} | \\ \mathbf{v}_n \\ | \end{bmatrix} = \mathbf{V}\mathbf{p}$$

  - Select enough v so that the difference between original x and reconstructed x is small

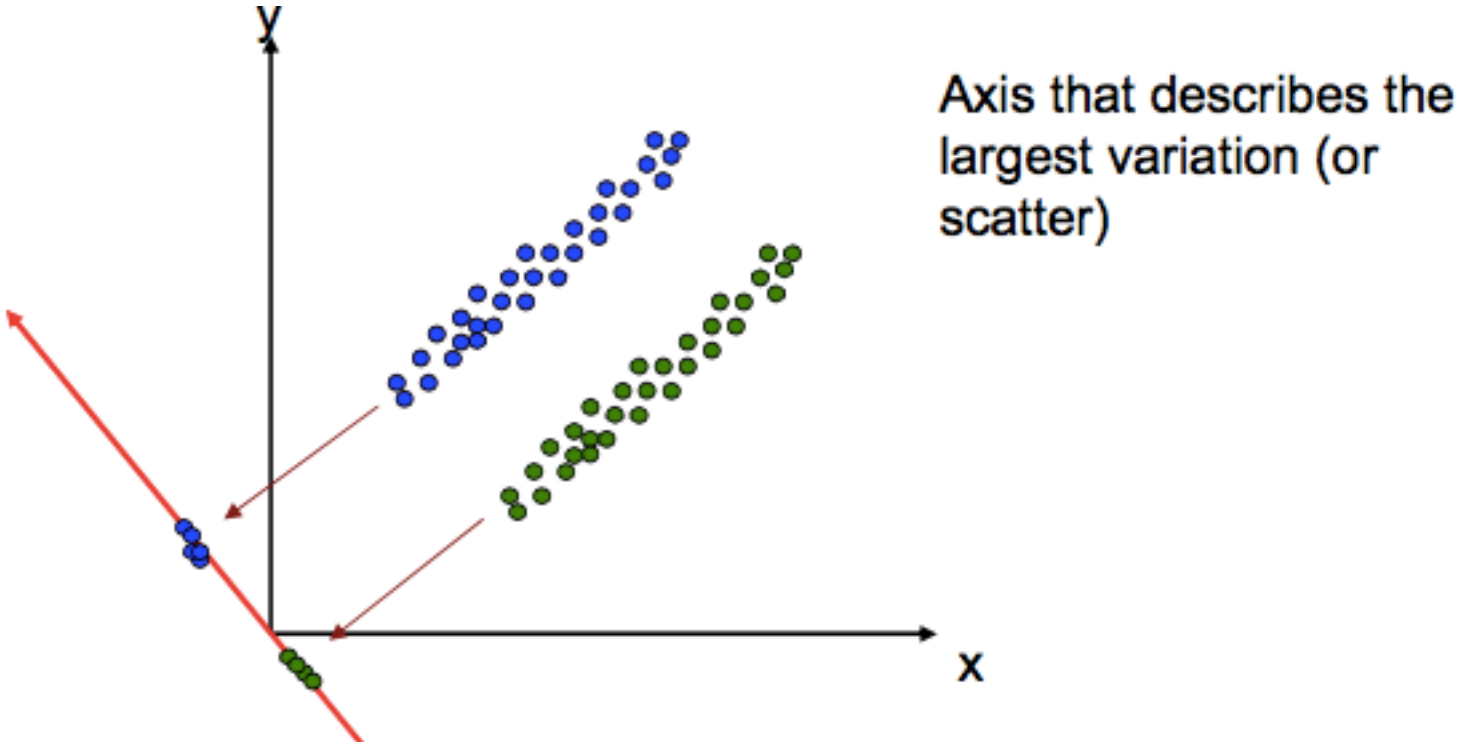
PCA for classification

- PCA does not cares about the class labels



What is LDA

- Find the projections that separate the classes.
- Assumes unimodal Gaussian model for each class
  - Maximize the distance between the means and minimize the variance of each class -> best classification performance



Simple 2 class case

- We want to maximize the distance between the projected means:

  e.g. maximize $|(\tilde{\mu}_1 - \tilde{\mu}_2)|^2$

Where $\tilde{\mu}_1$ is the projected mean μ_1 of class onto LDA direction vector $\mathbf{w}$, i.e.

$$\tilde{\mu}_1 = \mathbf{w}^T \boldsymbol{\mu}_1$$

and for class 2: $\tilde{\mu}_2 = \mathbf{w}^T \boldsymbol{\mu}_2$ thus

$$|(\tilde{\mu}_1 - \tilde{\mu}_2)|^2 = |(\mathbf{w}^T \boldsymbol{\mu}_1 - \mathbf{w}^T \boldsymbol{\mu}_2)|^2$$

$$= \mathbf{w}^T (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \mathbf{w}$$

$$= \mathbf{w}^T \mathbf{S}_B \mathbf{w}$$

Between class scatter matrix S_B

$$(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = (\mathbf{w}^T\boldsymbol{\mu}_1 - \mathbf{w}^T\boldsymbol{\mu}_2)^2$$
$$= \mathbf{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T\mathbf{w}$$
$$= \mathbf{w}^T\mathbf{S}_B\mathbf{w}$$

We want to maximize w^TS_Bw where S_B is the between class scatter matrix defined as:

$$\mathbf{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$$

We also want to minimize within class scatter

- The variance or scatter of each class. We also want to minimize them.

$$\tilde{s}_1^{\,2} = \sum_{i=1}^{N_1} (\tilde{x}_i - \tilde{\mu}_1)^2$$

Minimize the total scatter $\tilde{s}_1^{\,2} + \tilde{s}_2^{\,2}$

Within class scatter

- Lets expand on scatter s_1, s_2.

$$\tilde{s}_1^2 = \sum_{i=1}^{N_1} (\tilde{x}_i - \tilde{\mu}_1)^2$$

$$= \sum_{i=1}^{N_1} (\mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \boldsymbol{\mu}_1)^2$$

$$= \sum_{i=1}^{N_1} \mathbf{w}^T (\mathbf{x}_i - \boldsymbol{\mu}_1)(\mathbf{x}_i - \boldsymbol{\mu}_1)^T \mathbf{w}$$

$$= \mathbf{w}^T \mathbf{S}_1 \mathbf{w}$$

Total within class scatter

- We want to minimize $\tilde{s}_1^2 + \tilde{s}_2^2$

- This is the same as $\mathbf{w}^T \mathbf{S}_w \mathbf{w}$

$$\mathbf{S}_w = \sum_{i=1}^{C} \sum_{j=1}^{Ni} (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T$$

C number of classes, Ni number of images from class i

Fisher Linear Discriminant Criterion

- We want to maximize between class scatter
- We want to minimize within class scatter

- We have an objective function as a ratio so we can achieve both!

$$J(\mathbf{w}) = \frac{|(\tilde{\mu}_1 - \tilde{\mu}_2)|^2}{\tilde{s}_1^{\,2} + \tilde{s}_2^{\,2}}$$

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathbf{B}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathbf{W}}\mathbf{w}}$$

LDA solution

- If you do calculus

$$\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w}$$

$$\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{w} = \lambda\mathbf{w}$$

If S_w is non-singular and invertible.

- Generalized eigenvalue problem. The number of solutions is min(rankS_B, rankS_W) = C-1 or N-C
- For 2 class this simplifies to
- Note this is only one projection direction

$$\mathbf{w} = \mathbf{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

LDA+PCA

- First do PCA to reduce dimension
- Then do LDA to maximize classification ability
- How many dimensions to PCA?
  - Do PCA to keep N-C eigenvectors -> Makes S_w full rank and invertible
  - Then, do LDA and compute C-1 projections in this N-C subspace
- PCA+LDA = Fisher project

Random projection

- Original d-dimensional data is project to k-dimensional subspace
- Using a random k x d matrix R with unit norm columns
  - Johnson-Lindenstrauss lemma: If points in a vector space are projected onto a randomly selected subspace of suitably high dimension, then the distances between the points are approximately preserved
- Elements of R are usually selected from Gaussians.
  - Generally any zero mean unit variance distribution would satisfy Johnson-Lindenstrauss lemma.

Random projection notes

- R is not generally orthogonal.
  - But in a substantially large subspace, random vectors might be close to orthogonal.
- Looks weird but works…

Summary

- PCA
- LDA
  - PCA+LDA
- Random projection
- Homework

- Next time tSNE and SVM