CS423: Data Mining

Dimensionality Reduction

Jakramate Bootkrajang

Department of Computer Science
Chiang Mai University

Outlines

- Principle Component Analysis
- Feature Subset Selection
- Random Projection

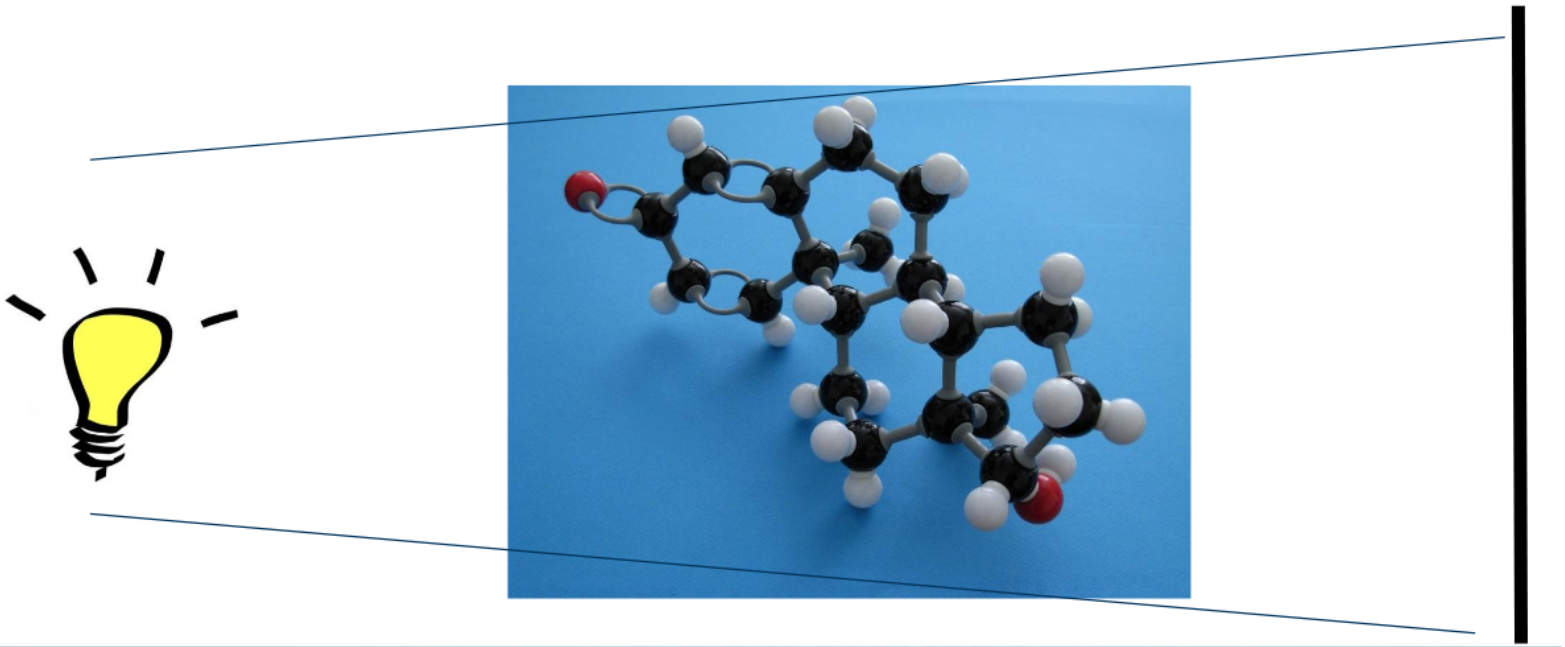
Principle Component Analysis: PCA

PCA: Goals

- Reduce dimensionality of the data while trying to preserve data structure.

PCA: Intuition

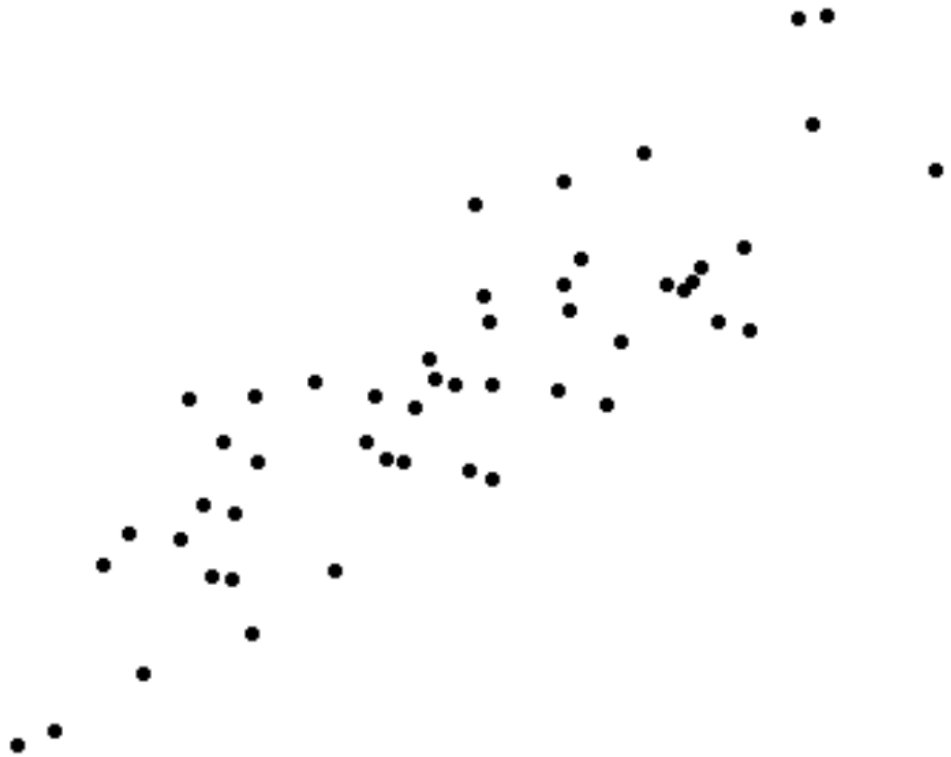
- Find low-dimensional projection with largest spread



Credit: Applied Multivariate Statistics: ETZ

PCA: Intuition

- Our data

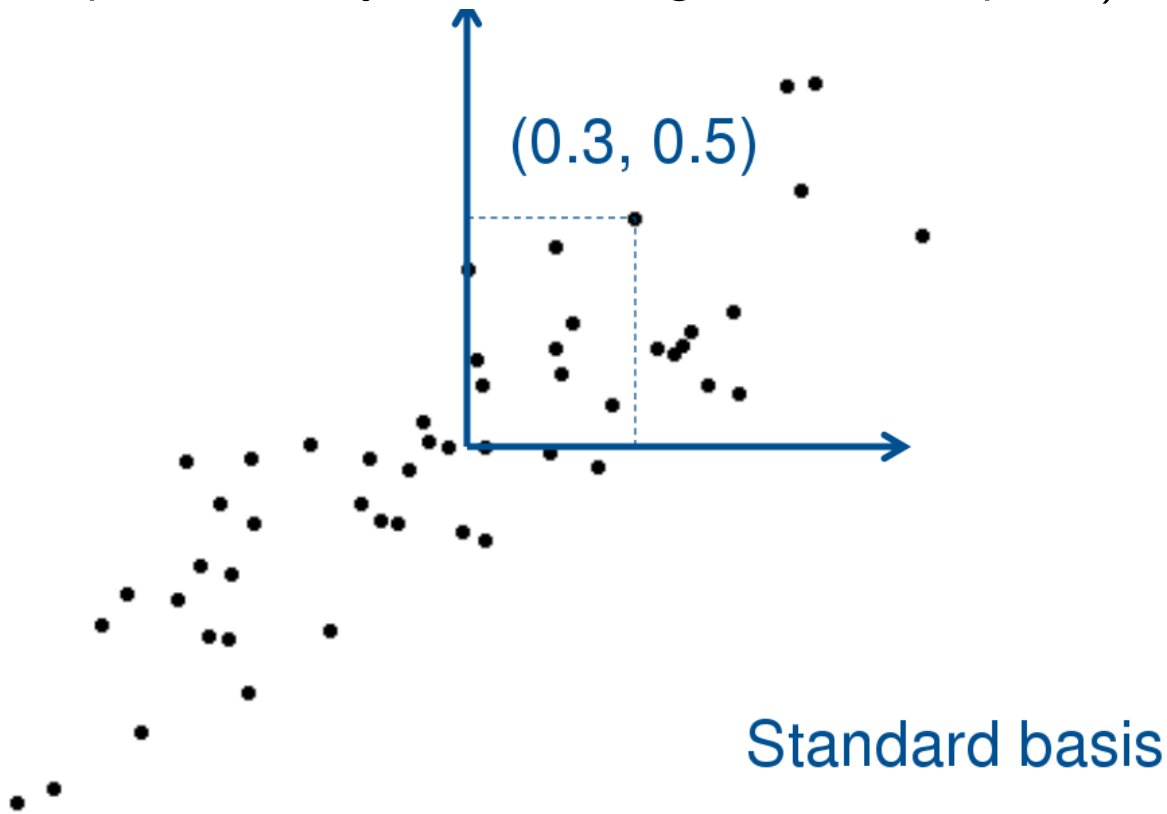


Credit: Applied Multivariate Statistics: ETZ

PCA: Intuition

- In the standard basis $\{[0 \quad 1], [1 \quad 0]\}$

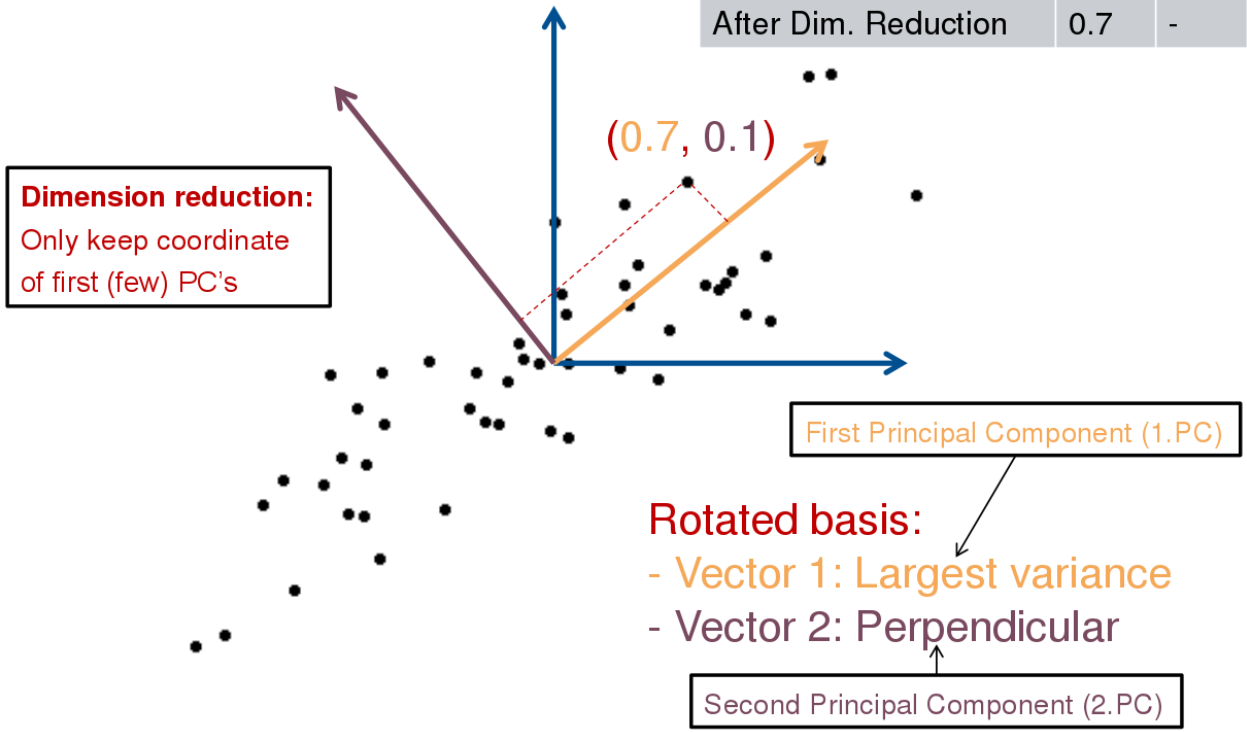
- (A basis is a set of linearly independent vectors that, in a linear combination, can represent every vector in a given vector space.)



PCA: Intuition

- In a new basis produced by PCA

	X_1	X_2
Std. Basis	0.3	0.5
PC Basis	0.7	0.1
After Dim. Reduction	0.7	-



PCA: in practice

- Find projection directions that maximise variance.
  - Subject to being uncorrelated with those already selected.
- These directions are known as principle components.
- They form a linear basis.
- Hopefully we can select a few of these PCs and project the data onto this new basis.

PCA: What is a projection ?

- Projection of a vector $\mathbf{x} = (x^1, x^2, \ldots, x^m)$ on to a *unit vector* $\mathbf{a} = (a^1, a^2, \ldots, a^m)$ is the linear combination

$$\mathbf{a} \cdot \mathbf{x} = \sum_{i=1}^{m} a^i x^i$$

PCA: What is a projection ?

- To generalise the above, the projection of a data point **x** onto some set of linearly independent vectors (*basis*) A is then

$$A\mathbf{x}^T = \begin{bmatrix} a^1 & a^2 & ... & a^m \\ b^1 & b^2 & ... & b^m \\ c^1 & c^2 & ... & c^m \end{bmatrix} \begin{bmatrix} x^1 \\ \vdots \\ x^m \end{bmatrix} = \begin{bmatrix} p^1 \\ p^2 \\ p^3 \end{bmatrix}$$

- $\mathbf{p}^T$ is an 1×3 vector, compared to the original **x** which was $1 \times m$ vector.

PCA: Theory

- Goal: find projection directions that maximise variance.
- Assumed data matrix X and its projection $\mathbf{a}X^T$ on some direction $\mathbf{a}$ are mean-centred.
- The variance of the projection is

$$\begin{aligned}\sigma_{\mathbf{a}}^2 &= (\mathbf{a}X^T)(\mathbf{a}X^T)^T \\ &= \mathbf{a}X^T X\mathbf{a}^T \\ &= \mathbf{a}V\mathbf{a}^T\end{aligned}$$

  where V is the $m \times m$ covariance matrix of the data

- We see that variance is a function of both the projection direction $\mathbf{a}$ and the covariance matrix V.

PCA: Maximising the variance

- Recall: variance of the projection $\sigma_{\mathbf{a}}^2 = \mathbf{a} V \mathbf{a}^T$
- Maximising the above makes no sense, as we can increase the variance by increasing **a**.
- We have to impose normalisation constraint on the vector **a** such that $\mathbf{a}\mathbf{a}^T = 1$
- We then instead optimise

$$u = \mathbf{a} V \mathbf{a}^T - \lambda(\mathbf{a}\mathbf{a}^T - 1)$$

- $\lambda > 0$ is a Lagrange multiplier imposing the needed constraint.

PCA: Maximising the variance (cont.)

- Differentiating with respect to **a** and equating to zero yields

$$\frac{\partial u}{\partial \mathbf{a}} = 2V\mathbf{a} - 2\lambda\mathbf{a} = 0 \tag{1}$$

$$V\mathbf{a} = \lambda\mathbf{a} \tag{2}$$

Eq.(5) is one type of Linear system of equations

PCA: Characteristic Equations, Eigenvectors and Eigenvalues

- $V\mathbf{a} = \lambda\mathbf{a}$ is called the Characteristic Equations.
- For an $m \times m$, real and symmetric matrix V, there are m possible solution vectors to this system.
- In our case, V is surely real and symmetric since it's the covariance matrix.
- Each of the solutions $\mathbf{a}_i$ is also known as eigenvector of the covariance matrix V.
- Each eigenvector is associated with an eigenvalue λ_i.

Side note on Eigenvector problem

Eigenvector and Eigenvalue

- Definition 1: A nonzero vector x is an eigenvector (or characteristic vector) of a square matrix A if there exists a scalar λ such that $Ax = \lambda x$.
- Then λ is an eigenvalue (or characteristic value) of A.
- Note: The zero vector can not be an eigenvector even though $A0 = \lambda 0$.
- But $\lambda = 0$ can be an eigenvalue.

Geometric interpretation

- An $n \times n$ matrix A multiplied by $n \times 1$ vector x results in another $n \times 1$ vector $y = Ax$. Thus A can be considered as a transformation matrix.
- In general, a matrix acts on a vector by changing both its magnitude and its direction.
- However, a matrix may act on certain vectors by changing only their magnitude, and leaving their direction unchanged (or possibly reversing it).
- These vectors are the eigenvectors of the matrix.

Example

- Show x $= \begin{bmatrix} 2 \\ 1 \end{bmatrix}$ is an eigenvector for A $= \begin{bmatrix} 2 & -4 \\ 3 & -6 \end{bmatrix}$
- Solution: Ax = ...

Finding Eigenvectors

- Given A $= \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$, find its eigenvectors and eigenvalues
- Solution:
  1. from definition $Ax = \lambda x$
  2. $(A - \lambda I)x = 0$
  3. Since x is nonzero, we know that $(A - \lambda I)$ is not invertible.
  4. So determinant of $(A - \lambda I)$ must be zero.
  5. $|A - \lambda I| = 0$

Properties of Eigenvectors

- All eigenvalues of a real symmetric matrix are real
- For symmetric matrices, eigenvectors for distinct eigenvalues are orthogonal
- An m-by-m square matrix can have at most m distinct eigenvectors

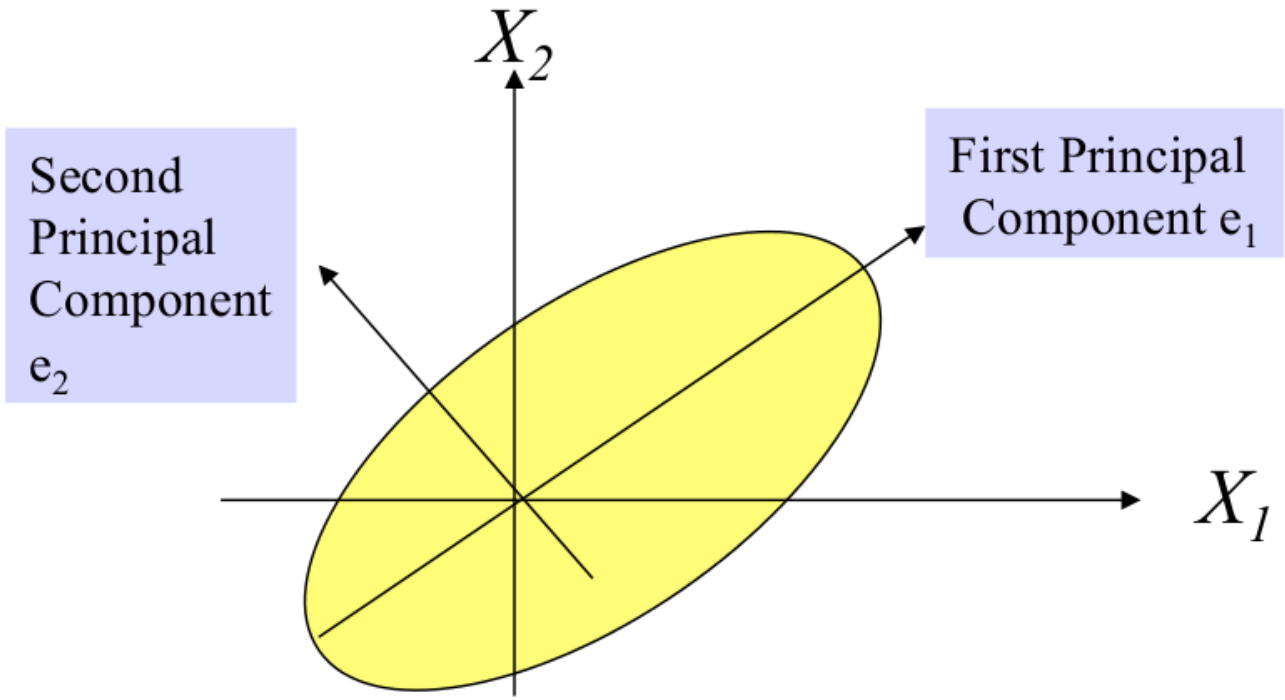
PCA: So...

- Principle Component is actually an eigenvector of the covariance matrix.
- The first principal component is the eigenvector associated with the largest eigenvalue.
- The second principal component is then the eigenvector associated with the second largest eigenvalue, and so on.
- Note that eigenvectors of V are all orthogonal to each other.

PCA: Practical usage

- Standardising X.
- Calculate V, a covariance matrix for X
- Find all the eigenvectors of V.
- Select k most important principle components put it in a matrix A.
- Project X onto A by calculating AX^T.

PCA: Example



Credit: Applied Multivariate Statistics: ETZ

PCA: How to choose dimensionality of the subspace?

- Recall that our eigenvector $\mathbf{a}_i$ is a unit vector.
- Recall also that eigenvalue is the amount of stretching that the covariance matrix V induced on $\mathbf{a}_i$.
- So from $V\mathbf{a}_i = \lambda_i \mathbf{a}_i$, we see that λ_i is actually the variance of data after projecting on $\mathbf{a}_i$.
- The variance error of selecting k PCs and omitting the rest is then equal

$$\frac{\sum_{i=k+1}^{m} \lambda_i}{\sum_{i=1}^{m} \lambda_i}$$

- We can stop adding more PCs once the above error exceeds some predefined threshold.

PCA: Computational Issue

- $O(nm^2)$ + Complexity of solving eigenvector.
- Calculating V takes $O(nm^2)$.
- Can be applied to large dataset but *does not scale well* with dimensionality m.

Feature Subset Selection

FSS: Goal

- Goal: Find the optimal feature subset.
- There are a number of methods.
  - Wrappers
  - Filters
  - Embedded methods

FSS: Key ideas

- Need a measure for assessing the goodness of a feature subset (scoring function).
- A strategy to search the space of possible feature subsets.
- Brute force is not applicable (NP-Hard).
- Involve ranking features based on some criteria.

FSS: Widely used ranking criteria

- Signal-2-Noise Ratio
- Information gain
- Mutual Information

FSS: Signal-2-Noise Ratio

- Define how well a feature discriminates two classes.

$$S2N = \frac{\mu_1 - \mu_2}{\sigma_1 + \sigma_2}$$

- μ_1 is the mean of the data points having positive class label.
- μ_2 is the mean of the data points having negative class label.
- σ_1 is the standard deviation of the data points having positive class label.
- σ_2 is the standard deviation of the data points having negative class label.

FSS: Information gain

- Measures the number of bits of information gained about the class label when knowing the feature.
- Measure the uncertainty about Y (the class label)
- Measure the uncertainty about Y given feature X (the class label with feature)
- The uncertainties can be measure using the entropy $H(Y)$ and $H(Y|X)$
- Information gain, $IG(X) = H(Y) - H(Y|X)$

FSS: Entropy

- A measure of uncertainties of information content.
- For example
  - An entropy of a fair coin toss is large since we cannot be certain about the outcome, $P(outcome = `head') = P(outcome = `tail') = 0.5$
  - However, an entropy of a biased coin toss is lower than that of the above, i.e., $P(outcome = `head')$ is much higher than $P(outcome = `tail')$.

FSS: Calculating an entropy

- Entropy of event Y

$$H(Y) = -\sum_{i=1}^{k} P(Y = y_i) \log_2(P(Y = y_i))$$

- Conditional entropy of Y given X

$$H(Y|X) = \sum_{j=1}^{r} P(X = x_j) H(Y|X = x_j)$$

where, $H(Y|X = x_j)$ refers to an entropy of Y among only those records in which X has value x_j

- Specifically,

$$H(Y|X = x_j) = -\sum_{i=1}^{k} P(Y = y_i|X = x_j) \log_2(P(Y = y_i|X = x_j))$$

FSS: Information gain (an example)

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

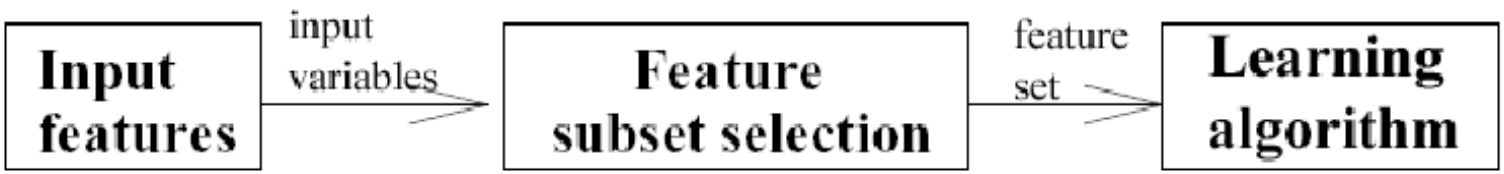
- $H(Y) =$
- $H(Y|X) =$
- $IG = H(Y) - H(Y|X)$

FSS: Mutual Information

- $I(x, y) =$ how much information do x and y share.
- It measures how much knowing one of these variables reduces uncertainty about the other.
- High MI example: Gender and name
- Low MI example: Gender and ???
- Definition: $I(x, y) = \sum_x \sum_y p(x, y) \log \frac{p(x,y)}{p(x)p(y)}$
- Can you show that there is a connection between the conditional entropy $H(Y|X)$ and mutual information?

FSS: Filter methods

- Select subsets of variables as a pre-processing step.
- Independently of the choice of classifier.



FSS: Pros/Cons of filter methods

Pros.

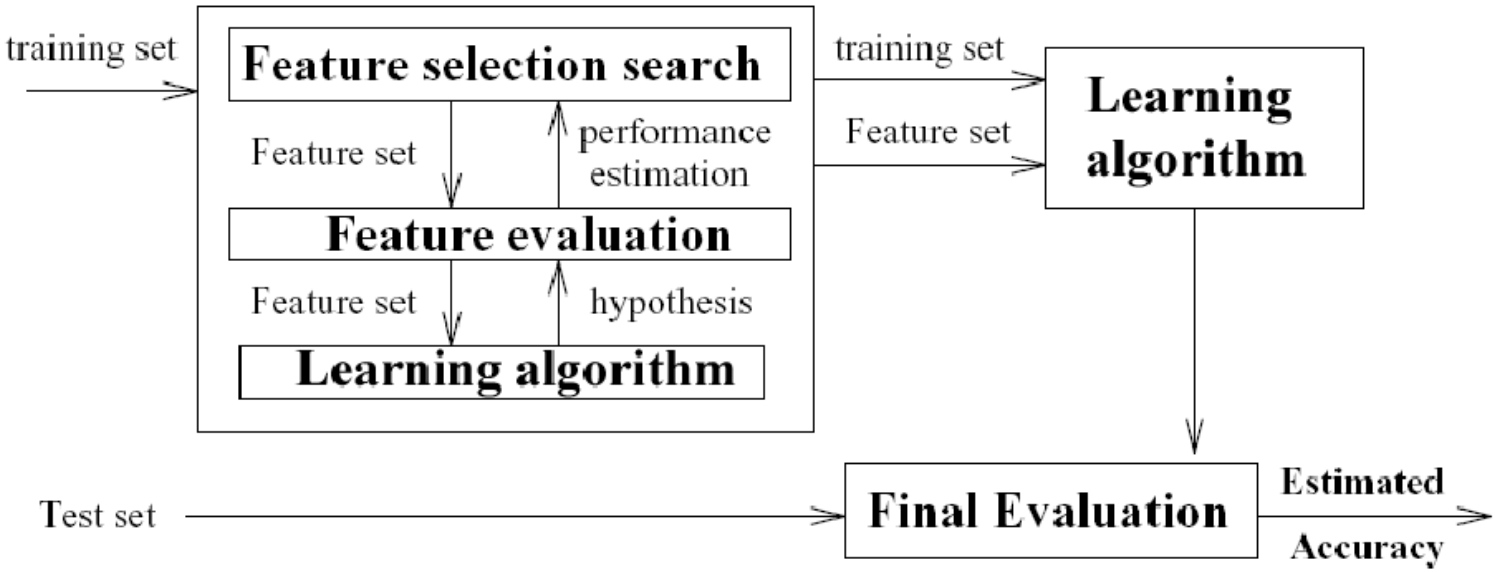
- Usually fast.
- Provide generic selection of features, not tuned by given learner.
- Can be used as a preprocessing step for other methods.

Cons

- Feature not optimised for the choice of classifier.

FSS: Wrapper methods

- Learner is considered a black-box, final subsets vary for different learners



- Need to define
  - How to search the space of all possible subsets?
  - How to assess the performance of a learner?

FSS: Wrapper methods

- How to search the space?
  - Brute force: Not a good idea, there are 2^m subsets to evaluate.
  - Forward selection (start with empty feature set and add features at each step)
  - Backward elimination (start with full feature set and discard features at each step)
- How to evaluate the learner?
  - For classification: Measure accuracy on hold-out validation set.

FSS: Embedded methods

- Specific to a given learning machine
- Performs variable selection in the process of training
- Optimise the regularised objective: *obj* + *regularisation*
- For example, $\text{argmin}_w \sum_{i=1}^{N} y_i(\mathbf{w}^T\mathbf{x}_i + b) - \lambda \sum_{j=1}^{M} |w_j|$

Random Projection

RP: Key Idea

- We have seen that the working of data mining algorithms depends in some way or other on the geometry of data – lengths of vectors, distances, angles.
- Hign dimensional geometry is very different from low dimensional geometry. It defeats our intuitions – we can see things in 2D and 3D but have no idea how things will look like in HD.

RP: So what happens in HD?

Concentration of norms: Generate points in 2D and up and measure their lengths.

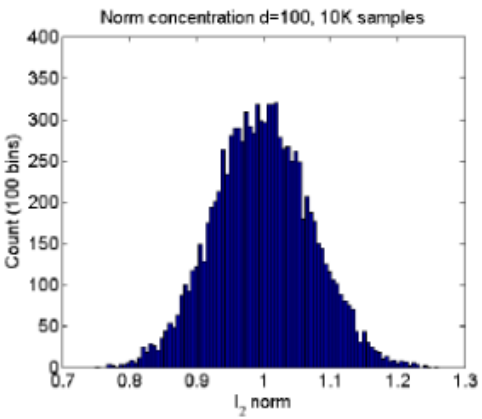


Figure: $d = 100$ norm concentration

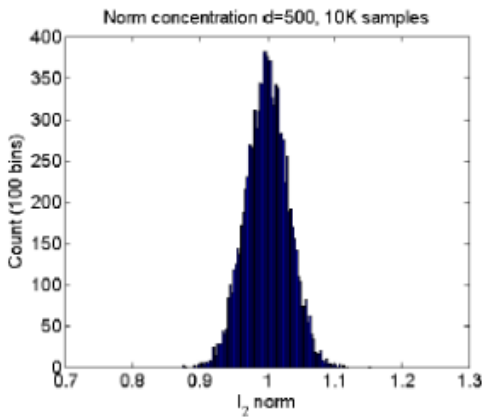


Figure: $d = 500$ norm concentration

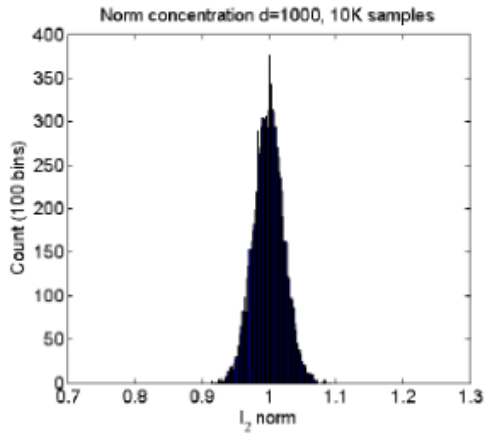


Figure: $d = 1000$ norm concentration

Figure: Credit: A.Kaban, CS-Bham

RP: So what happens in HD?

Near-ortholonality: Generate points in 2D and up and measure their angles.

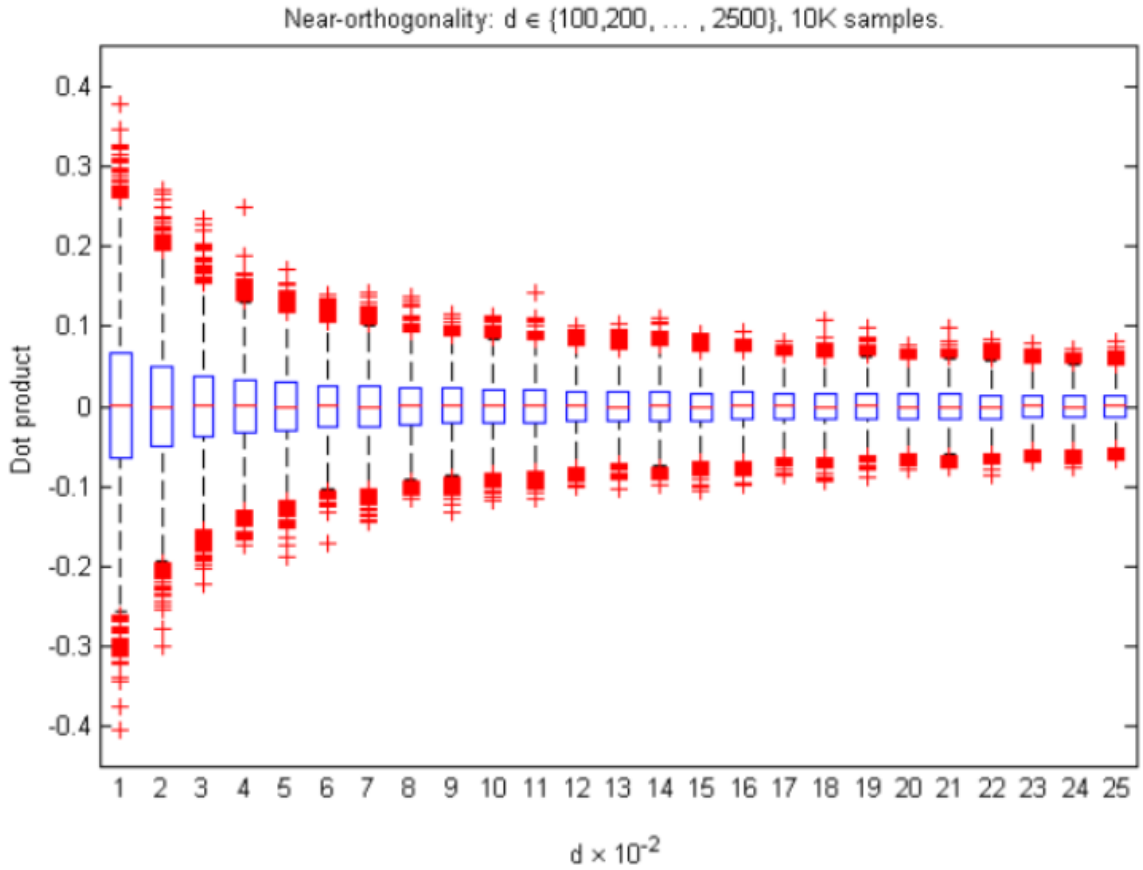


Figure: Credit: A.Kaban, CS-Bham

RP: What is happening as d $\rightarrow \infty$

- We can see from the plots that
  - As m increases, any two random vectors end up being orthogonal to each other.
  - As m increases, any random vectors ends up having about the same length.

RP: Consequently

- When data has little structure (many features are independent of each other), then the 'nearest neighbour' is at about the same distance as the furthest one.

- When data does have structure, we can use a small collection of random vectors to project our data to lower dimension without losing much of the structure.

RP: Random projection in practice

- Repeat T times
  1. Generate a $k \times m$ random matrix R_i where $k < m$.
  2. Matrix entries are sampled i.i.d from some distribution e.g., normal distribution
  3. Pre-multiply data matrix X with R_i to get k-dimensional data matrix P_i
- Final data matrix is

$$\frac{\sum_{i=1}^{T} P_i}{T}$$

RP: Why does it work ?

Theorem[Johnson and Lindenstrauss, 1984] Let $\epsilon \in (0,1)$. Let $N, k \in \mathbb{N}$ such that $k \geq C\epsilon^{-2} \log N$, for a large enough absolute constant C. Let $V \subseteq \mathbb{R}^d$ be a set of N points. Then there exists a linear mapping $R : \mathbb{R}^d \to \mathbb{R}^k$, such that for all $u, v \in V$:

$$(1-\epsilon)\|u-v\|_{\ell_2^d}^2 \leq \|Ru - Rv\|_{\ell_2^k}^2 \leq (1+\epsilon)\|u-v\|_{\ell_2^d}^2$$

- With high probability *random projection* satisfies JLL [Dasgupta & Gupta '02] (proof by Chernoff bounding).