04.02.2019

Statistical Methods in AI (CSE/ECE 471)

Lecture-9:

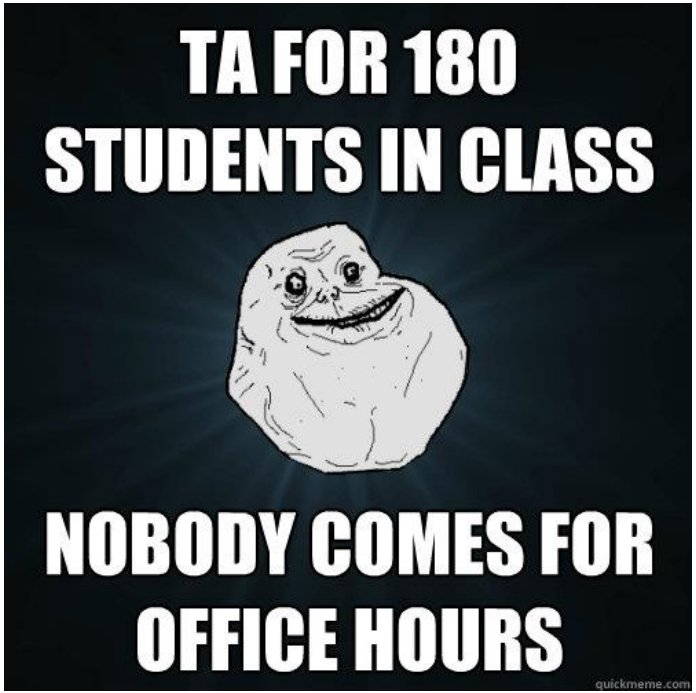
- Basic Feature Selection approaches
- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)

Ravi Kiran

Center for Visual Information Technology (CVIT), IIIT Hyderabad

Announcements

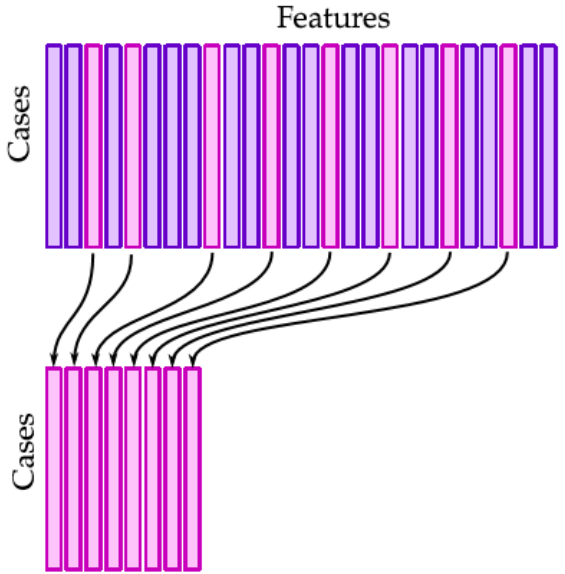
- Assignment Schedule is up on Moodle.
- TAs



Feature Selection

- when we have a regular data set ($|T| > D$), but too flexible model, and/or
- when we have a high-dimensional data set with not enough data ($|T| < D$).

Data sets with thousands or millions of variables (features) are quite usual these days: we want to choose only those, which are needed to *construct simpler, faster, and more accurate models*.



Feature Selection - approaches

based on the number of variables considered together:

- **Univariate methods, variable ranking:**
  consider the input variables (features, attributes) one by one.
- **Multivariate methods, variable subset selection:**
  consider whole groups of variables together.

Feature Selection - approaches

based on the number of variables considered together:

- **Univariate methods, variable ranking:**
  consider the input variables (features, attributes) one by one.
- **Multivariate methods, variable subset selection:**
  consider whole groups of variables together.

Feature Selection – Univariate methods

- The main or auxiliary technique in many more complex methods.
- Simple and scalable, often works well in practice.

Feature Selection – Univariate methods

- The main or auxiliary technique in many more complex methods.
- Simple and scalable, often works well in practice.

Incomplete list of methods usable for various combinations of input and output variable.

Input variable X	Output variable Y: Nominal	Output variable Y: Continuous
Nominal	Confusion matrix analysis $p(Y)$ vs. $p(Y\|X)$ χ^2-test of independence Inf. gain (see decision trees)	T-test, ANOVA ROC (AUC) discretize Y (see the left column)
Continuous	T-test, ANOVA ROC (AUC) logistic regression discretize X (see the top row)	correlation regression discretize Y (see the left column) discretize X (see the top row)

Feature Selection – Univariate methods

- The main or auxiliary technique in many more complex methods.
- Simple and scalable, often works well in practice.

Incomplete list of methods usable for various combinations of input and output variable.

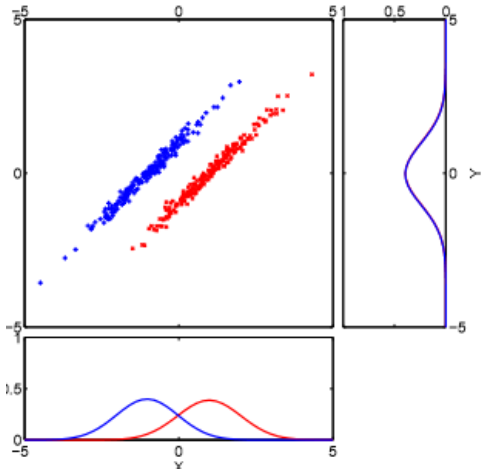
Input variable X	Output variable Y: Nominal	Output variable Y: Continuous
Nominal	Confusion matrix analysis $p(Y)$ vs. $p(Y\|X)$ χ^2-test of independence Inf. gain (see decision trees)	T-test, ANOVA ROC (AUC) discretize Y (see the left column)
Continuous	T-test, ANOVA ROC (AUC) logistic regression discretize X (see the top row)	correlation regression discretize Y (see the left column) discretize X (see the top row)

- All the methods provide a score which can be used to rank the input variables according to the "size of relationship" with the output variable.
- Statistical tests provide the so-called p-values (attained level of significance); these may serve to judge the absolute "importance" of an attribute.

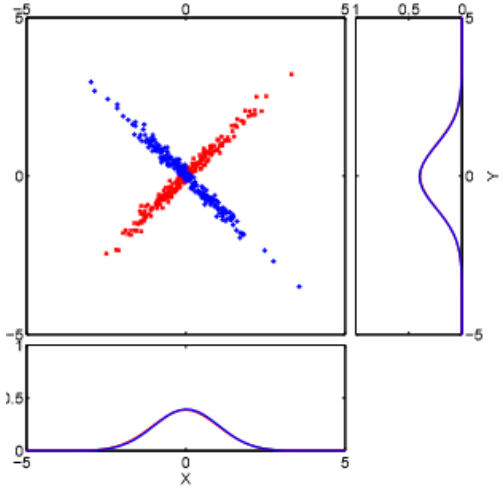
Feature Selection – Univariate methods

Univariate methods may fail:

- They needn't recognize that a feature is important (in combination with other variable).



Seemingly useless variable Y is useful in combination with X



Both X,Y seem useless but combination is useful

Feature Selection – Univariate methods

Univariate methods may fail:

- They needn't recognize that a feature is important (in combination with other variable).
- They can select a group of variables which are dependent and carry similar (or the same) information about the output, i.e. it is sufficient to use only one (or a few) of these variables.

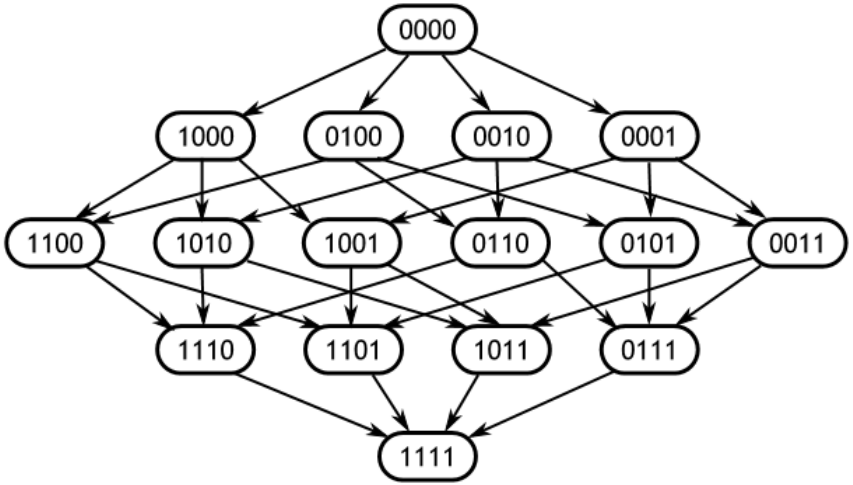
Feature Selection - approaches

based on the number of variables considered together:

- **Univariate methods, variable ranking:**
  consider the input variables (features, attributes) one by one.
- **Multivariate methods, variable subset selection:**
  consider whole groups of variables together.

Multivariate feature selection is complex!

- D variables, 2^D variable subsets!



Feature Selection - approaches

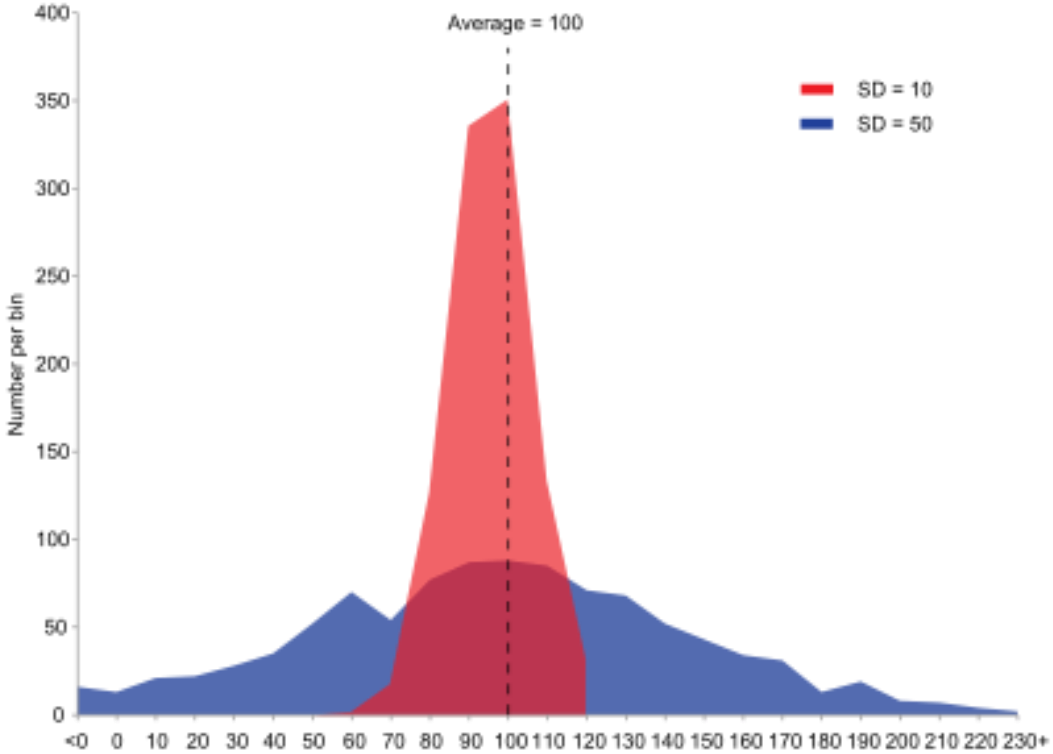
based on the use of the ML model in the feature selection process:

- **Filter:** selects a subset of variables independently of the model that shall subsequently use them.
- **Wrapper:** selects a subset of variables taking into accoiunt the model that shall use them.
- **Embedded method:** the feature selection method is built in the ML model (or rather its training algorithm) itself (e.g. decision trees).

Feature Selection – Filter Method

- Features are scored independently and the top *S* are used by the classifier.
- Score: correlation, mutual information, t-statistic, F-statistic, p-value, tree importance statistic, etc.

Variance, Covariance and Correlation



Population

Variance:

$$s^2 = \frac{\sum(\overline{X} - X_i)^2}{N}$$

Standard Deviation:

$$s = \sqrt{\frac{\sum(\overline{X} - X_i)^2}{N}}$$

Sample Variance

$$s^2 = \frac{\sum(x - \bar{x})^2}{n - 1}$$

Sample Standard Deviation

$$s = \sqrt{\frac{\sum(x - \bar{x})^2}{n - 1}}$$

Covariance

Vectors 1 and 3

Cell (3, 1) or (1, 3)

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 5 & 4 & 1 \\ 3 & 8 & 6 \end{bmatrix}$$

Covariance

$$\begin{bmatrix} 2.67 & 0.67 & -2.67 \\ 0.67 & 4.67 & 2.33 \\ -2.67 & 2.33 & 4.67 \end{bmatrix}$$

Covariance Matrix

Variance:

$$s^2 = \frac{\sum(\overline{X} - X_i)^2}{N}$$

$$cov(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

$$\begin{bmatrix} 0.39701 & 0.51117 \\ 0.55582 & 0.93003 \\ 0.59403 & 0.96645 \\ 0.51544 & 0.29759 \\ 0.85313 & 0.18118 \\ 0.88564 & 0.69114 \end{bmatrix}$$

$$\begin{pmatrix} & M1 & M2 & M3 & \dots & Mn \\ S1 & q_{1,1} & q_{1,2} & q_{1,3} & \dots & q_{1,n} \\ S2 & q_{2,1} & q_{2,2} & q_{2,3} & \dots & q_{2,n} \\ S3 & q_{3,1} & q_{3,2} & q_{3,3} & \dots & q_{3,n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ Sm & q_{m,1} & q_{m,2} & q_{m,3} & & q_{m,n} \end{pmatrix}$$

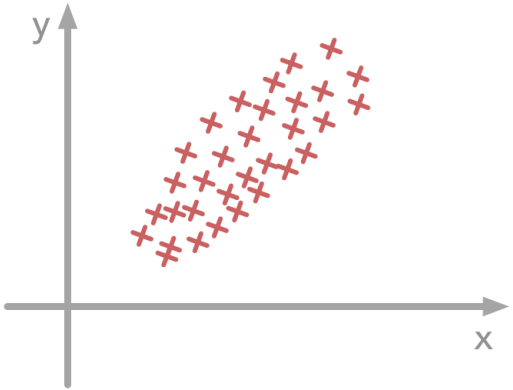
$$Cov(M_a, M_b) = \frac{1}{m}\sum_{i=1}^{m}(q_{i,a} - \overline{q_a})(q_{i,b} - \overline{q_b})$$

$$C = \begin{pmatrix} \operatorname{cov}(M_1, M_1) & \operatorname{cov}(M_1, M_2) & \cdots & \operatorname{cov}(M_1, M_n) \\ \operatorname{cov}(M_2, M_1) & \operatorname{cov}(M_2, M_2) & \cdots & \operatorname{cov}(M_2, M_n) \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{cov}(M_n, M_1) & \operatorname{cov}(M_n, M_2) & \cdots & \operatorname{cov}(M_n, M_n) \end{pmatrix}_{n \times n}$$

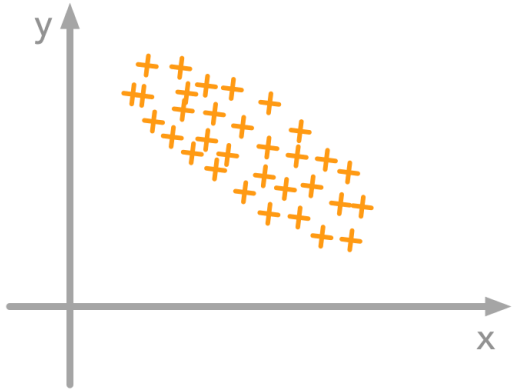
n-dimensional Covariance Matrix

Covariance

Positive covariance



Negative covariance

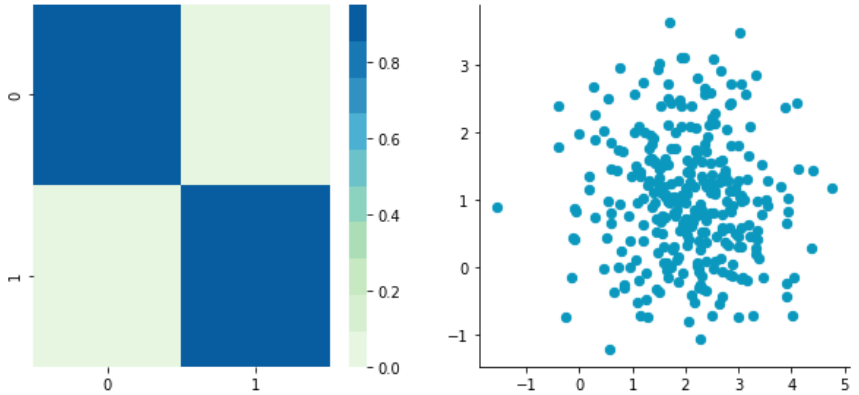


Variance:

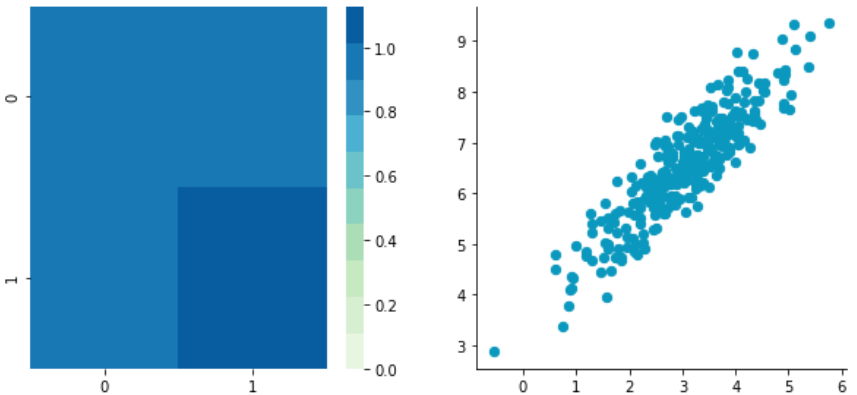
$$s^2 = \frac{\sum (\overline{X} - X_i)^2}{N}$$

$$cov(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$$

Covariance



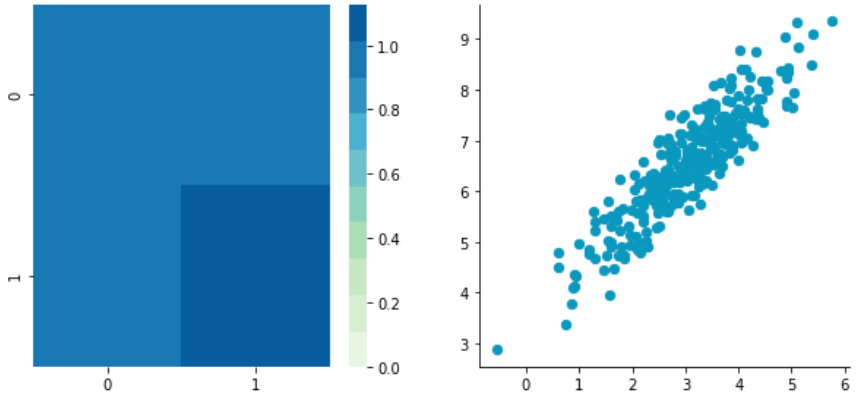
$$C = \begin{bmatrix} +0.95 & -0.04 \\ -0.04 & +0.87 \end{bmatrix}$$

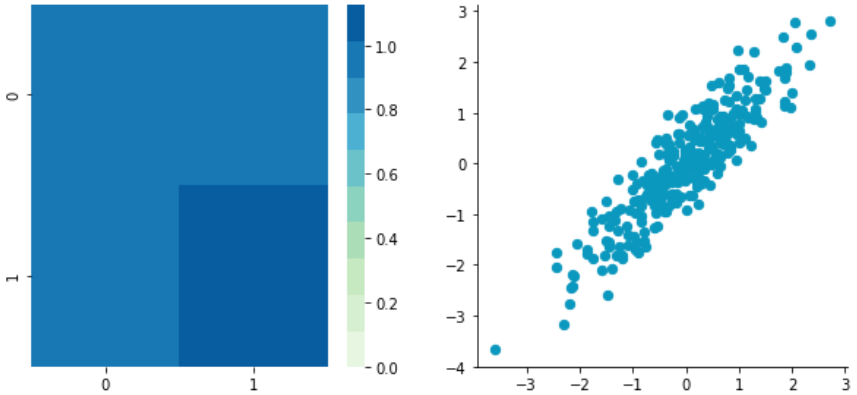


$$C = \begin{bmatrix} +0.95 & +0.92 \\ +0.92 & +1.12 \end{bmatrix}$$

Mean Normalization

$$\boldsymbol{X}' = \boldsymbol{X} - \bar{x}$$



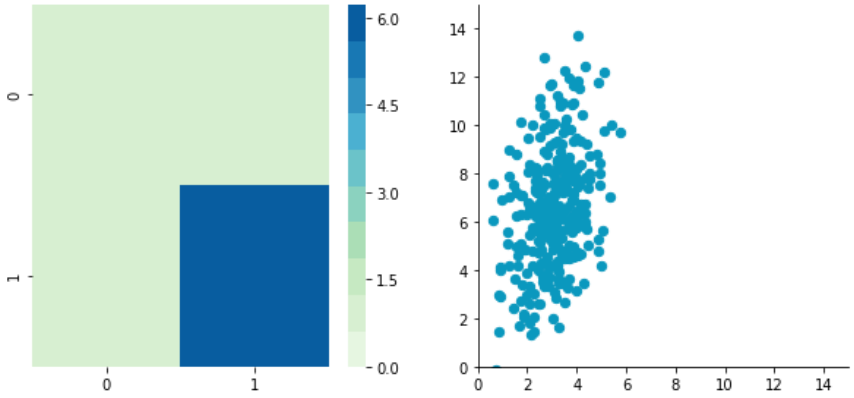


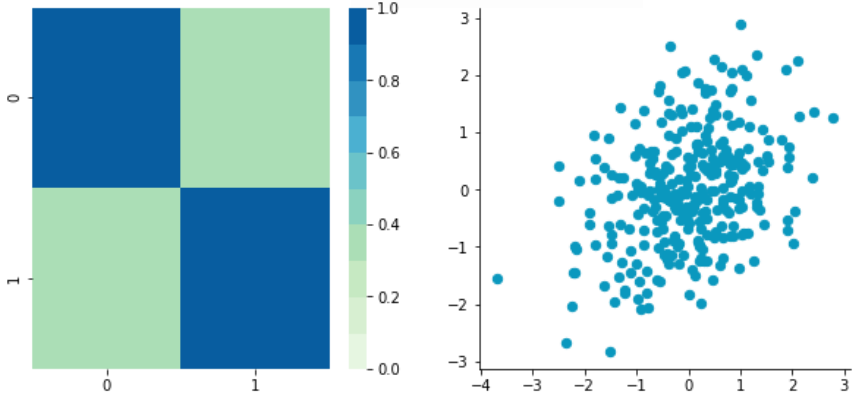
$$C = \begin{bmatrix} +0.95 & +0.92 \\ +0.92 & +1.12 \end{bmatrix}$$

$$C = \begin{bmatrix} +0.95 & +0.92 \\ +0.92 & +1.12 \end{bmatrix}$$

Standardization

$$X' = \frac{X - \bar{x}}{\sigma_X}$$





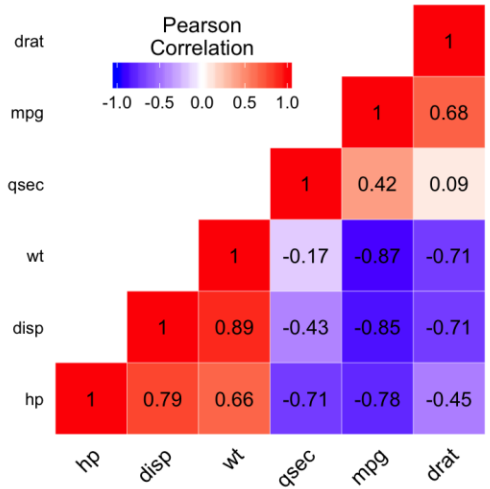
$$C = \begin{bmatrix} +0.95 & +0.83 \\ +0.83 & +6.22 \end{bmatrix}$$

$$C = \begin{bmatrix} +1.00 & +0.34 \\ +0.34 & +1.00 \end{bmatrix}$$

Correlation

$$cov(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

$\frac{1}{n}$ $x^T y$

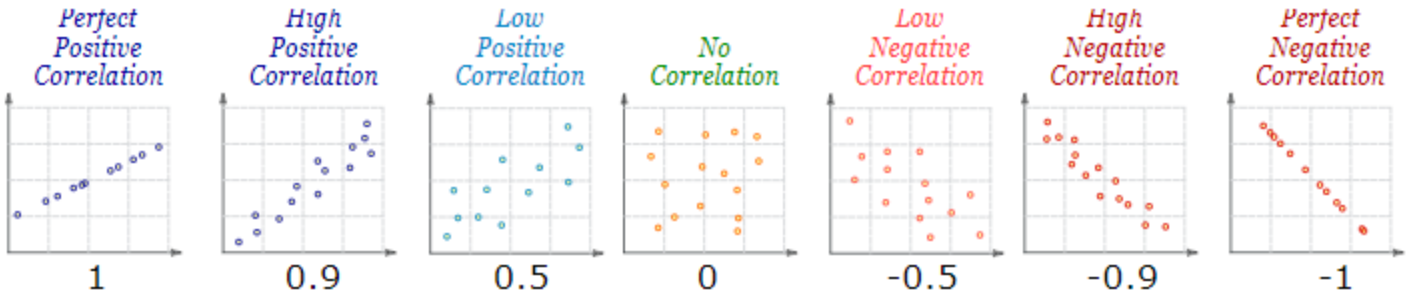


$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

$$\rho_{AB} = \frac{\mathrm{cov}_{AB}}{\sigma_A \sigma_B}$$

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s_x}\right)\left(\frac{y_i - \bar{y}}{s_y}\right)$$

$$X' = \frac{X - \bar{x}}{\sigma_X}$$



PCA Toy Example

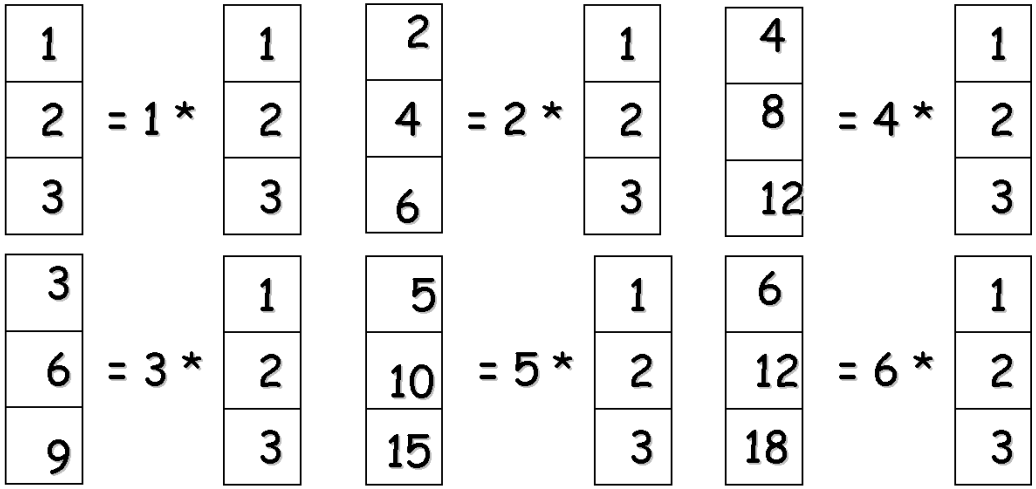
Consider the following 3D points

1	2	4	3	5	6
2	4	8	6	10	12
3	6	12	9	15	18

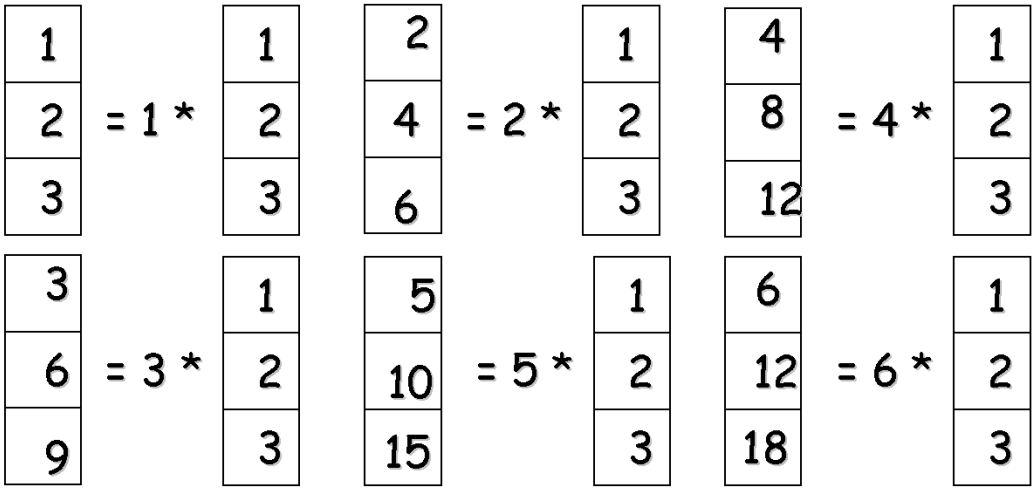
If each component is stored in a byte,
we need 18 = 3 x 6 bytes

PCA Toy Example

Looking closer, we can see that all the points are related geometrically: they are all the same point, scaled by a factor:



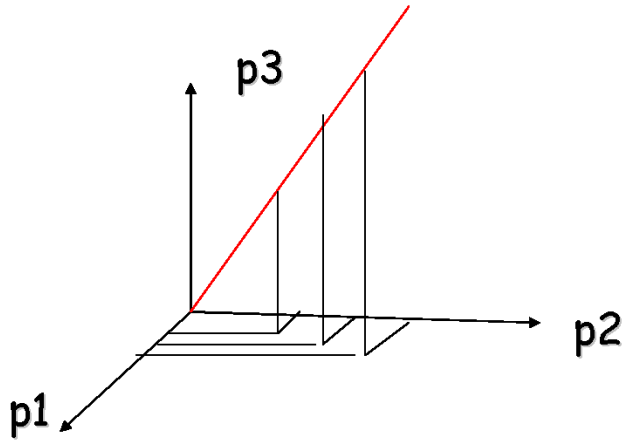
PCA Toy Example



They can be stored using only 9 bytes (50% savings!):
Store one point (3 bytes) + the multiplying constants (6 bytes)

Geometrical Interpretation:

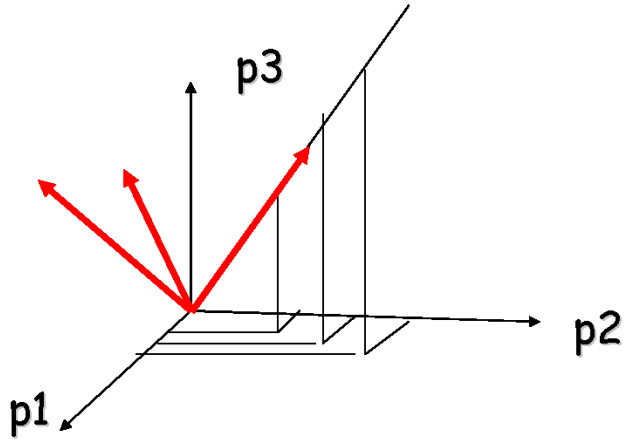
View each point in 3D space.



But in this example, all the points happen to belong to a line: a 1D subspace of the original 3D space.

Geometrical Interpretation:

Consider a new coordinate system where one of the axes is along the direction of the line:

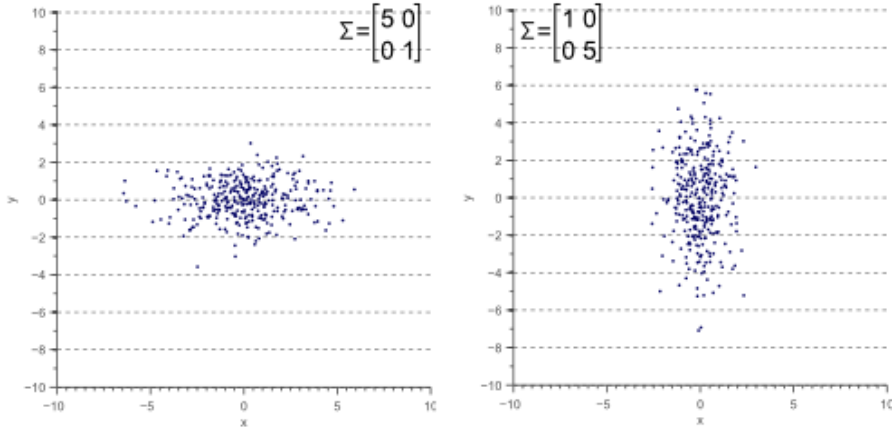


In this coordinate system, every point has only one non-zero coordinate: we *only* need to store the direction of the line (a 3 bytes image) and the non-zero coordinate for each of the points (6 bytes).

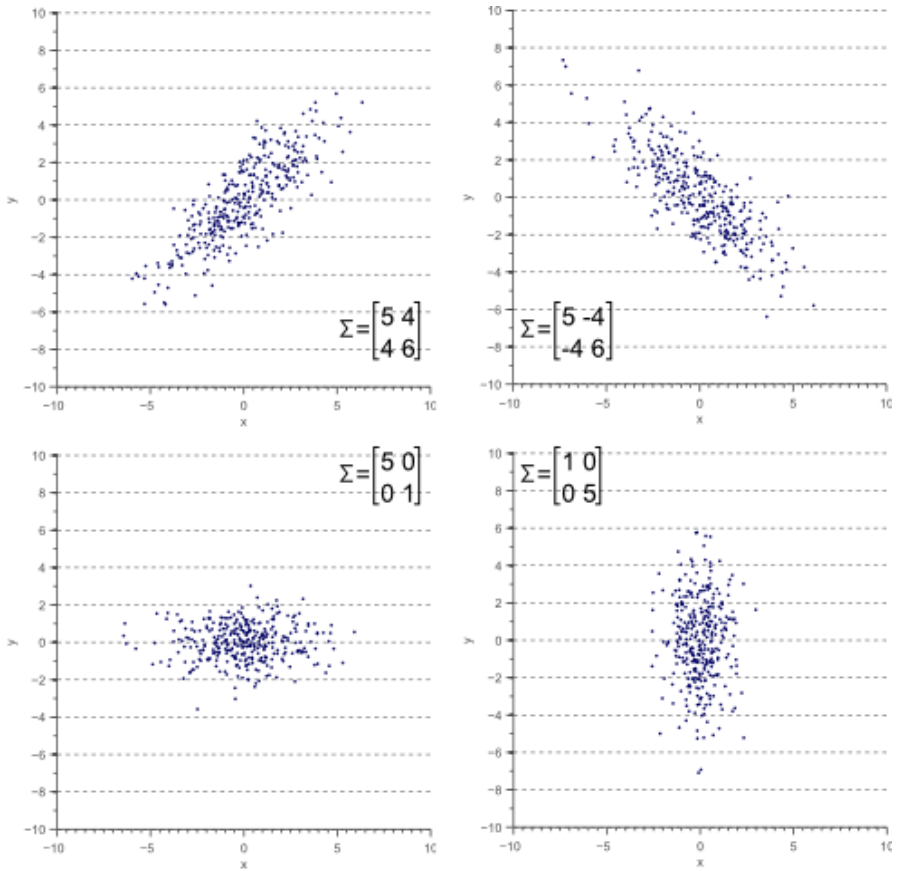
Principal Component Analysis (PCA)

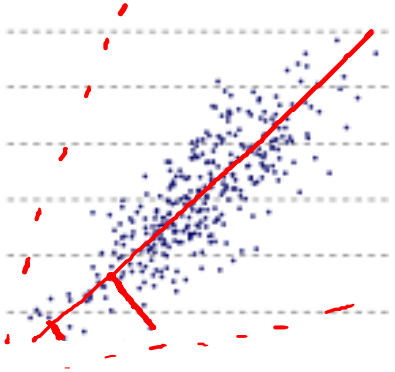
- Given a set of points, how do we know if they can be compressed like in the previous example?
  - The answer is to look into the correlation between the points
  - The tool for doing this is called PCA

Covariance Matrix encodes spread and orientation of data



Covariance Matrix encodes spread and orientation of data





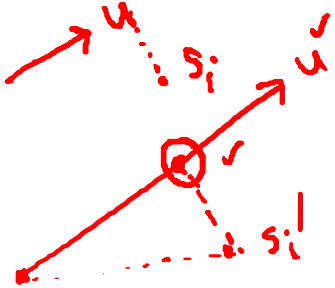
$D = \begin{bmatrix} | & | & & | \\ s_1 & s_2 & \cdots & s_n \\ | & | & & | \end{bmatrix}_{d \times n}$ ← n samples → , d ↕

mean centered

$= \begin{bmatrix} — & f_1^T & — \\ — & f_2^T & — \\ — & f_d^T & — \end{bmatrix}_{1 \times n}$

$f(x) = x^e$

$\Sigma = \begin{bmatrix} f_1^T f_1 & f_1^T f_2 & \\ & & \\ & & f_d^T f_d \end{bmatrix}$



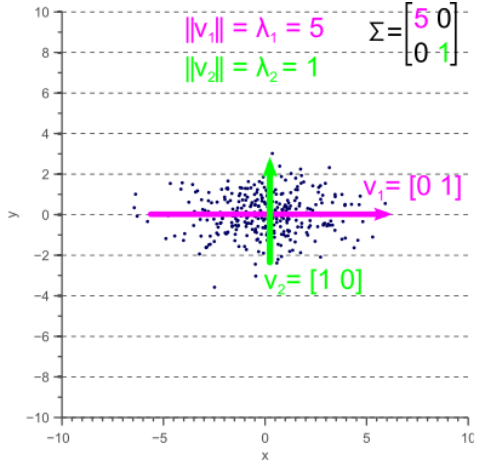
$\bar{p} = 0 \qquad p_i = u^T s_i$

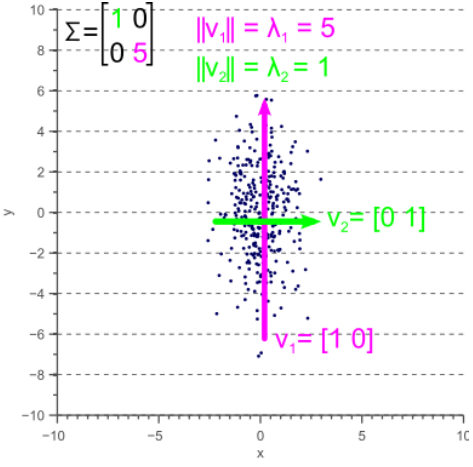
$p_i = \frac{u^T s_i}{\|u\|=1} \qquad u^T s_i = \frac{\sum_i p_i}{n} = \frac{\sum u^T s_i}{n} = u^T \bar{s} = 0$

$f(u) = u^T \Sigma u + \lambda(u^T u - 1) \rightarrow \sigma_p = u^T \Sigma u \qquad \|u\| = 1$

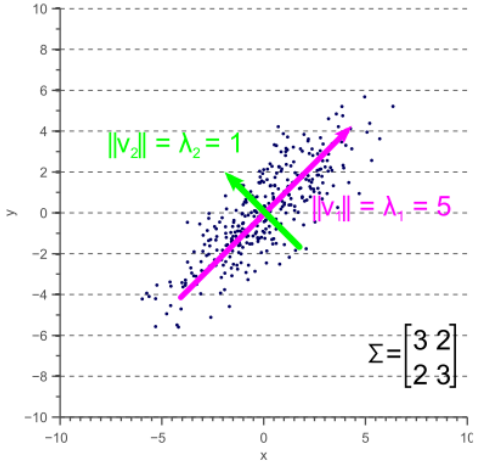
$f(u,\lambda) \qquad 2\Sigma u + 2\lambda u = 0 \rightarrow \Sigma u = \underbrace{-\lambda}_{p} u \qquad u^T u - 1 = 0 \qquad u^T u = 1$

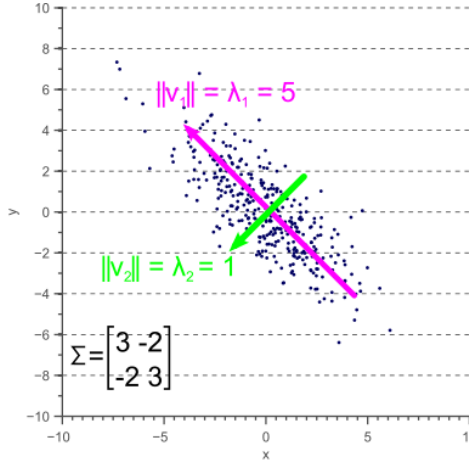
$$\Sigma\vec{v} = \lambda\vec{v}$$





$$\begin{bmatrix} \lambda_1 & \lambda_2 & \cdots & \lambda_n \end{bmatrix} \quad \Sigma$$





PCA

- By finding the eigenvalues and eigenvectors of the covariance matrix, we find that the eigenvectors with the largest eigenvalues correspond to the dimensions that have the strongest correlation in the dataset.
- This is the principal component.
- PCA is a useful statistical technique that has found application in:
  - fields such as face recognition and image compression
  - finding patterns in data of high dimension.

PCA Theorem

Let x_1 x_2 … x_n be a set of n N x 1 vectors and let $\bar{x}$ be their average:

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iN} \end{bmatrix} \qquad \bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{i=n} \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iN} \end{bmatrix}$$

PCA Theorem

Let X be the N x n matrix with columns
$x_1 - \bar{x}, x_2 - \bar{x}, \ldots x_n - \bar{x}$:

$$X = \left[\begin{array}{cccc} \mathrm{x}_1 - \bar{\mathrm{x}} & \mathrm{x}_2 - \bar{\mathrm{x}} & \cdots & \mathrm{x}_n - \bar{\mathrm{x}} \end{array} \right]$$

Note: subtracting the mean is equivalent to translating the coordinate system to the location of the mean.

PCA Theorem

Let Q = X X^T be the N x N matrix:

$$Q = XX^T = \left[\begin{array}{cccc} \mathbf{x}_1 - \bar{\mathbf{x}} & \mathbf{x}_2 - \bar{\mathbf{x}} & \cdots & \mathbf{x}_n - \bar{\mathbf{x}} \end{array} \right] \left[\begin{array}{c} (\mathbf{x}_1 - \bar{\mathbf{x}})^T \\ (\mathbf{x}_2 - \bar{\mathbf{x}})^T \\ \vdots \\ (\mathbf{x}_n - \bar{\mathbf{x}})^T \end{array} \right]$$

Notes:

1. Q is square
2. Q is symmetric
3. Q is the *covariance* matrix [aka scatter matrix]
4. Q can be very large (in vision, N is often the number of pixels in an image!)

PCA Theorem

Theorem:

Each x_j can be written as: $$\mathbf{x}_j = \bar{\mathbf{x}} + \sum_{i=1}^{i=n} g_{ji}\mathbf{e}_i$$

where e_i are the n eigenvectors of Q with non-zero eigenvalues.

Notes:

1. The eigenvectors e_1 e_2 … e_n span an ***eigenspace***
2. e_1 e_2 … e_n are N x 1 orthonormal vectors (directions in N-Dimensional space)
3. The scalars g_{ji} are the coordinates of x_j in the space.

$$g_{ji} = (\mathbf{x}_j - \bar{\mathbf{x}}).\mathbf{e}_i$$

Using PCA to Compress Data

- Expressing x in terms of $e_1 \ldots e_n$ has not changed the size of the data
- However, if the points are highly correlated many of the coordinates of x will be zero or closed to zero.

note: this means they lie in a lower-dimensional linear subspace

Using PCA to Compress Data

- Sort the eigenvectors e_i according to their eigenvalue:

$$\lambda_1 \geq \lambda_2 \geq \ldots \lambda_n$$

- Assuming that $\lambda_i \approx 0$ if $i > k$

- Then

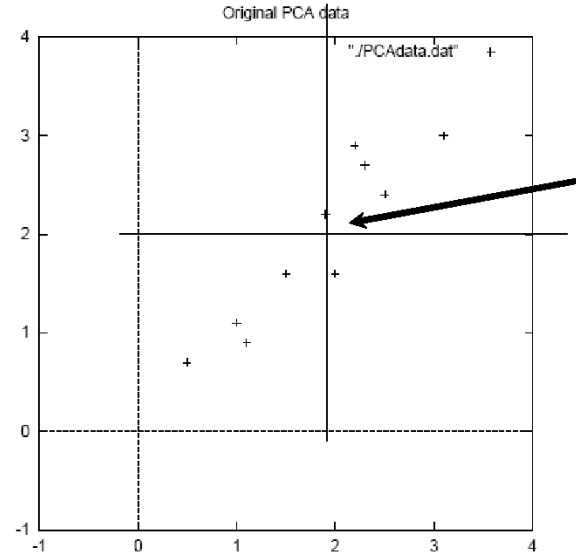
$$\mathbf{x}_j \approx \bar{\mathbf{x}} + \sum_{i=1}^{i=k} g_{ji}\mathbf{e}_i$$

PCA Example –STEP 1

http://kybele.psych.cornell.edu/~edelman/Psych-465-Spring-2003/PCA-tutorial.pdf

- DATA:

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9



mean

this becomes the new origin of the data from now on

Figure 3.1: PCA example data, original data on the left, data with the means subtracted on the right, and a plot of the data

PCA Example –STEP 2

- Calculate the covariance matrix

$$cov = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

- since the non-diagonal elements in this covariance matrix are positive, we should expect that both the x and y variable increase together.

PCA Example –STEP 3

- Calculate the eigenvectors and eigenvalues of the covariance matrix

$$\text{eigenvalues} = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$\text{eigenvectors} = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -735178656 \end{pmatrix}$$

PCA Example –STEP 3

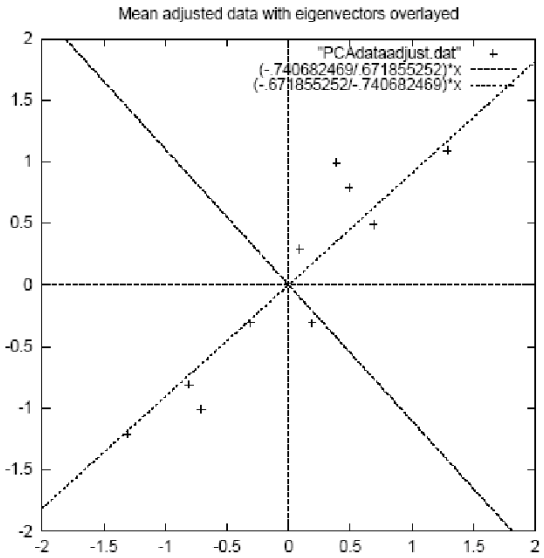


Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

- eigenvectors are plotted as diagonal dotted lines on the plot.
- Note they are perpendicular to each other.
- Note one of the eigenvectors goes through the middle of the points, like drawing a line of best fit.
- The second eigenvector gives us the other, less important, pattern in the data, that all the points follow the main line, but are off to the side of the main line by some amount.

PCA Example –STEP 4

- Feature Vector

  FeatureVector = (eig_1 eig_2 eig_3 … eig_n)

  We can either form a feature vector with both of the eigenvectors:

$$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

  or, we can choose to leave out the smaller, less significant component and only have a single column:

$$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

PCA Example –STEP 5

- Deriving new data coordinates

  FinalData = RowFeatureVector x RowZeroMeanData

  RowFeatureVector is the matrix with the eigenvectors in the columns *transposed* so that the eigenvectors are now in the rows, with the most significant eigenvector at the top

  RowZeroMeanData is the mean-adjusted data *transposed*, ie. the data items are in each column, with each row holding a separate dimension.

  Note: his is essential Rotating the coordinate axes so higher-variance axes come first.

PCA Example –STEP 5

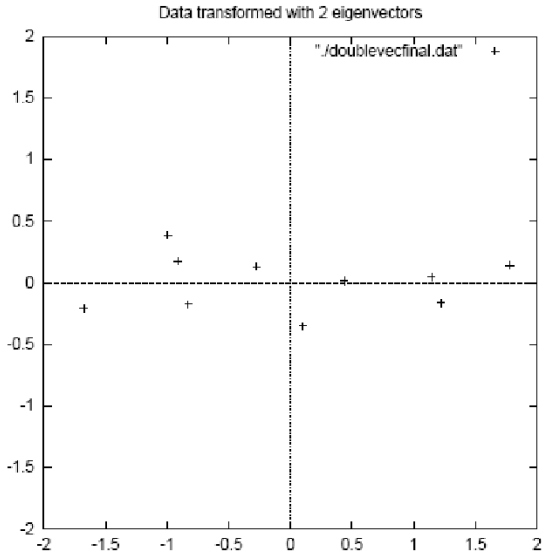
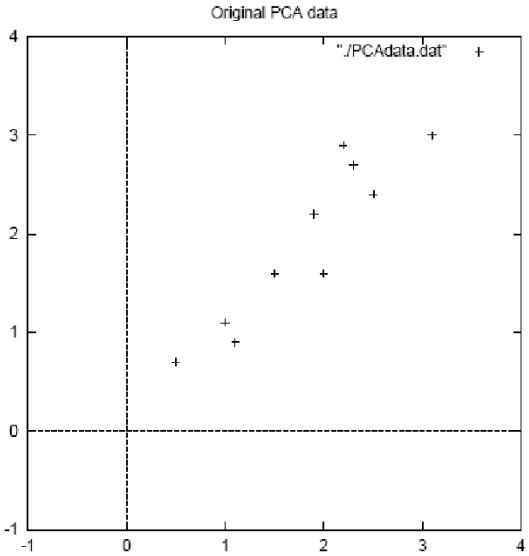


Figure 3.3: The table of data by applying the PCA analysis using both eigenvectors, and a plot of the new data points.

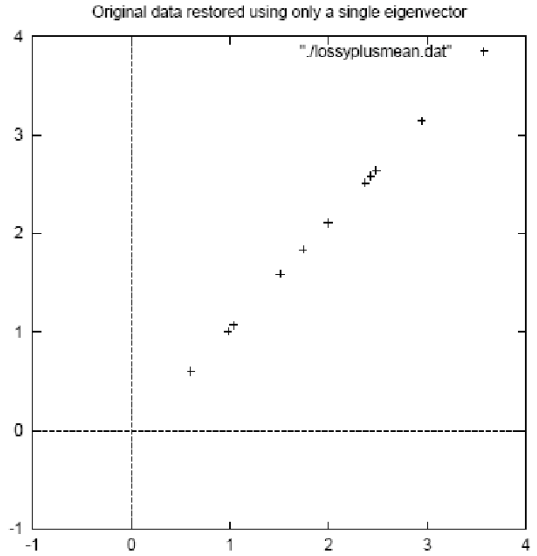
PCA Example : Approximation

- If we reduced the dimensionality, obviously, when reconstructing the data we would lose those dimensions we chose to discard. In our example let us assume that we considered only the x dimension…

PCA Example : Final Approximation



2D point cloud

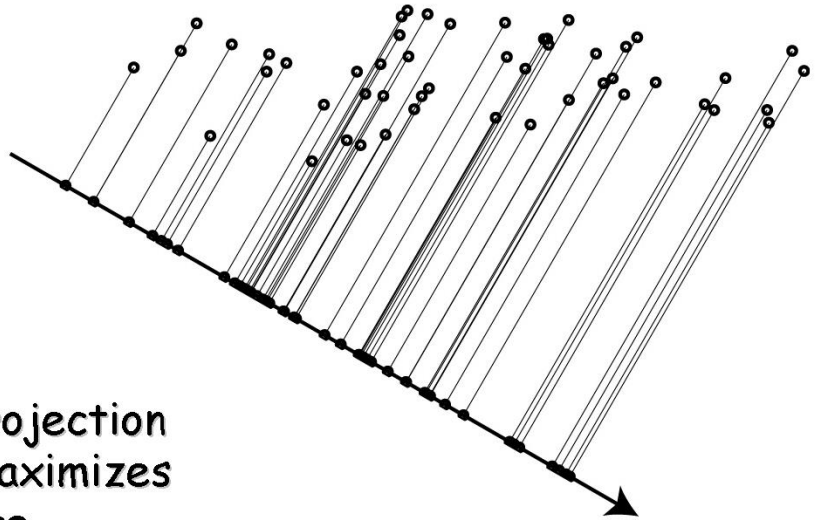


Approximation using one eigenvector basis

Another way of thinking about Principal component

- direction of maximum variance in the input space
- happens to be same as the principal eigenvector of the covariance matrix

One-dimensional projection



find projection
that maximizes
variance

Covariance to variance

- From the covariance, the variance of any projection can be calculated.
- Let *w* be a unit vector

$$\left\langle \left(w^T x\right)^2 \right\rangle - \left\langle w^T x \right\rangle^2 = w^T C w$$

$$= \sum_{ij} w_i C_{ij} w_j$$

Maximizing variance

- Principal eigenvector of C
  - the one with the largest eigenvalue.

$$w^* = \arg\max_{w:|w|=1} w^T C w$$

$$\lambda_{\max}(C) = \max_{w:|w|=1} w^T C w$$

$$= w^{*T} C w^*$$

1. Compute the mean feature vector

$$\mu = \frac{1}{p}\sum_{k=1}^{p} x_k$$, where, x_k is a pattern ($k = 1$ to p), p = number of patterns, x is the feature matrix

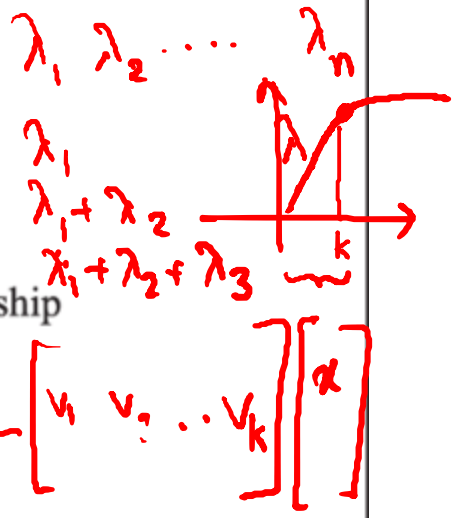
2. Find the covariance matrix

$$C = \frac{1}{p}\sum_{k=1}^{p}\{x_k - \mu\}\{x_k - \mu\}^T$$ where, T represents matrix transposition

3. Compute Eigen values λ_i and Eigen vectors v_i of covariance matrix
   $Cv_i = \lambda_i v_i$ ($i = 1, 2, 3, ...q$), q = number of features
4. Estimating high-valued Eigen vectors
   (i) Arrange all the Eigen values (λ_i) in descending order
   (ii) Choose a threshold value, θ
   (iii) Number of high-valued λ_i can be chosen so as to satisfy the relationship

$$\left(\sum_{i=1}^{s}\lambda_i\right)\left(\sum_{i=1}^{q}\lambda_i\right)^{-1} \geq \theta$$, where, s = number of high valued λ_i chosen

   (iv) Select Eigen vectors corresponding to selected high valued λ_i
5. Extract low dimensional feature vectors (principal components) from raw feature matrix.
   $P = V^T x$, where, V is the matrix of principal components and x is the feature matrix



Implementing PCA

- Need to find "first" k eigenvectors of Q:

$$Q = XX^T = \begin{bmatrix} \mathbf{x}_1 - \bar{\mathbf{x}} & \mathbf{x}_2 - \bar{\mathbf{x}} & \cdots & \mathbf{x}_n - \bar{\mathbf{x}} \end{bmatrix} \begin{bmatrix} (\mathbf{x}_1 - \bar{\mathbf{x}})^T \\ (\mathbf{x}_2 - \bar{\mathbf{x}})^T \\ \vdots \\ (\mathbf{x}_n - \bar{\mathbf{x}})^T \end{bmatrix}$$

Q is N x N (Again, N could be the number of pixels in an image. For a 256 x 256 image, N = 65536 !!)
Don't want to explicitly compute Q!!!!

Singular Value Decomposition (SVD)

Any m x n matrix X can be written as the product of 3 matrices:

$$X = UDV^T$$

Where:

- U is m x m and its columns are orthonormal vectors
- V is n x n and its columns are orthonormal vectors
- D is m x n diagonal and its diagonal elements are called the singular values of X, and are such that:

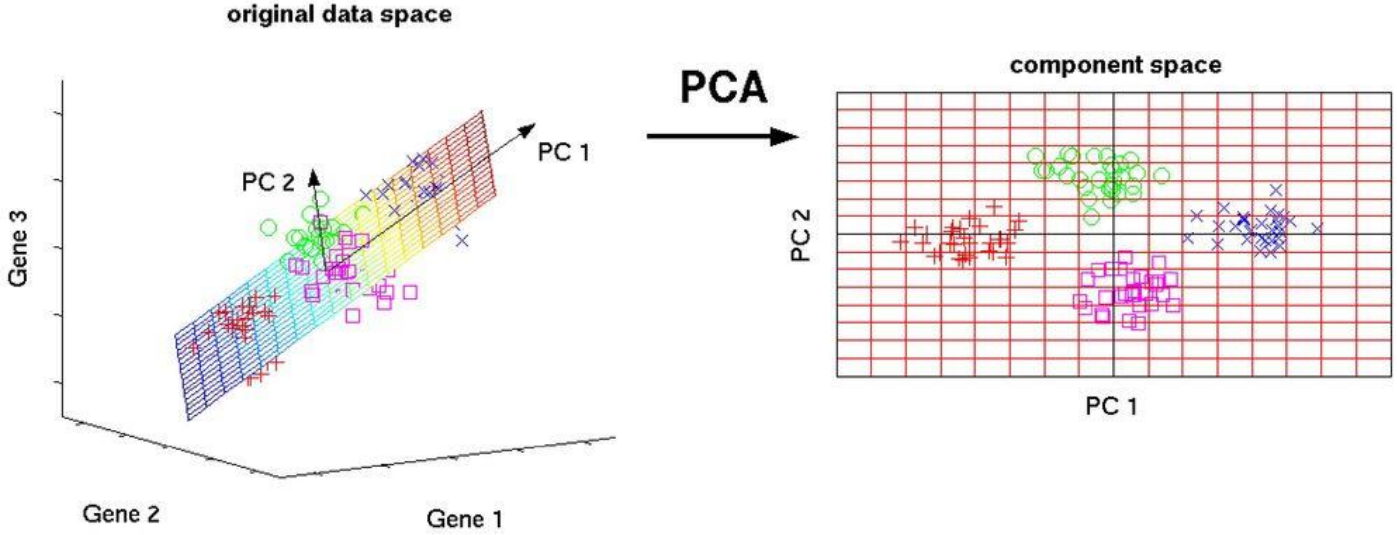
$$\sigma_1 \geq \sigma_2 \geq \ldots \sigma_n \geq 0$$

SVD Properties

$$X = U D V^T$$

- The columns of U are the eigenvectors of XX^T
- The columns of V are the eigenvectors of X^TX
- The squares of the diagonal elements of D are the eigenvalues of XX^T and X^TX

PCA finds new variables which are linear combinations of the original variables such that in the new space, the data has fewer dimensions.

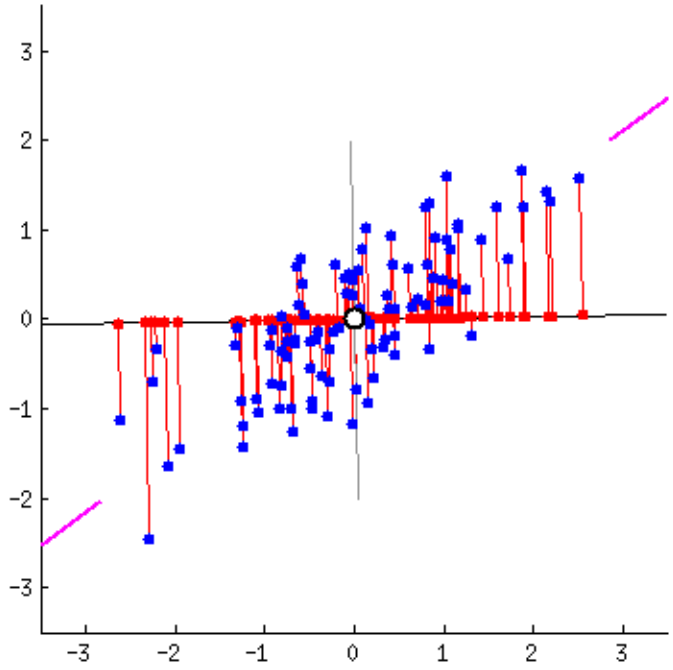


Dimensionality Reduction

- Linear transformations are simple to compute and tractable.

$$Y = U\ X \qquad (b_i = u_i^t a_i)$$

k x l k x d d x l (k<<d)


3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3

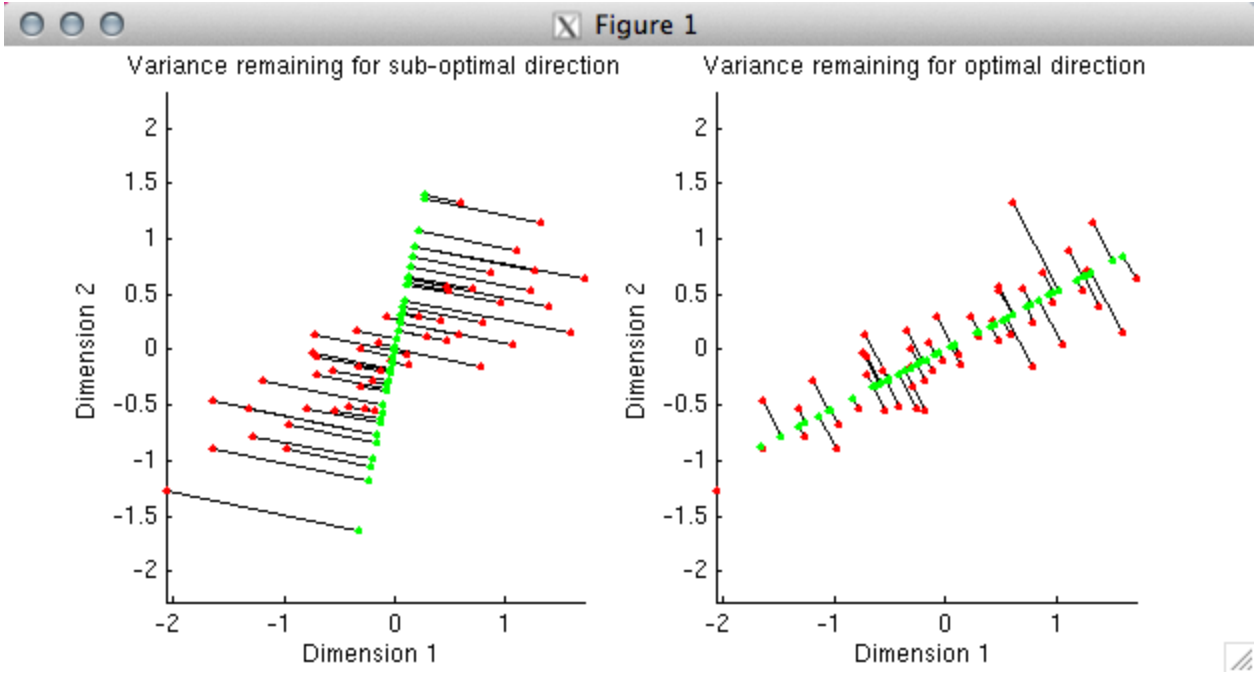
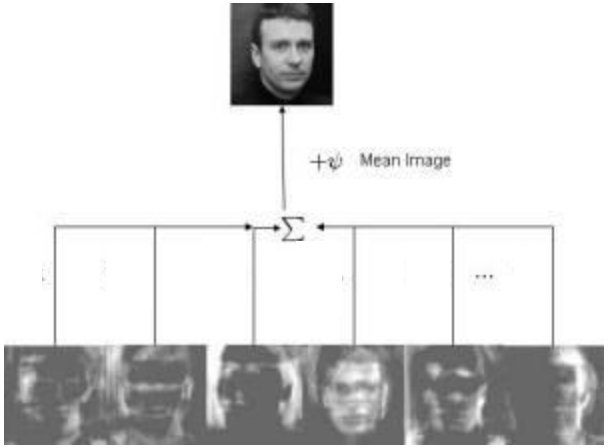


Figure 1
Variance remaining for sub-optimal direction
Variance remaining for optimal direction
Dimension 1
Dimension 2
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2

Each x_j can be written as: $$\mathbf{x}_j = \bar{\mathbf{x}} + \sum_{i=1}^{i=n} g_{ji}\mathbf{e}_i$$

where e_i are the n eigenvectors of Q with non-zero eigenvalues.



Feature Selection – Filter Method

- Features are scored independently and the top *S* are used by the classifier.
- Score: correlation, mutual information, t-statistic, F-statistic, p-value, tree importance statistic, etc.

Feature Selection - approaches

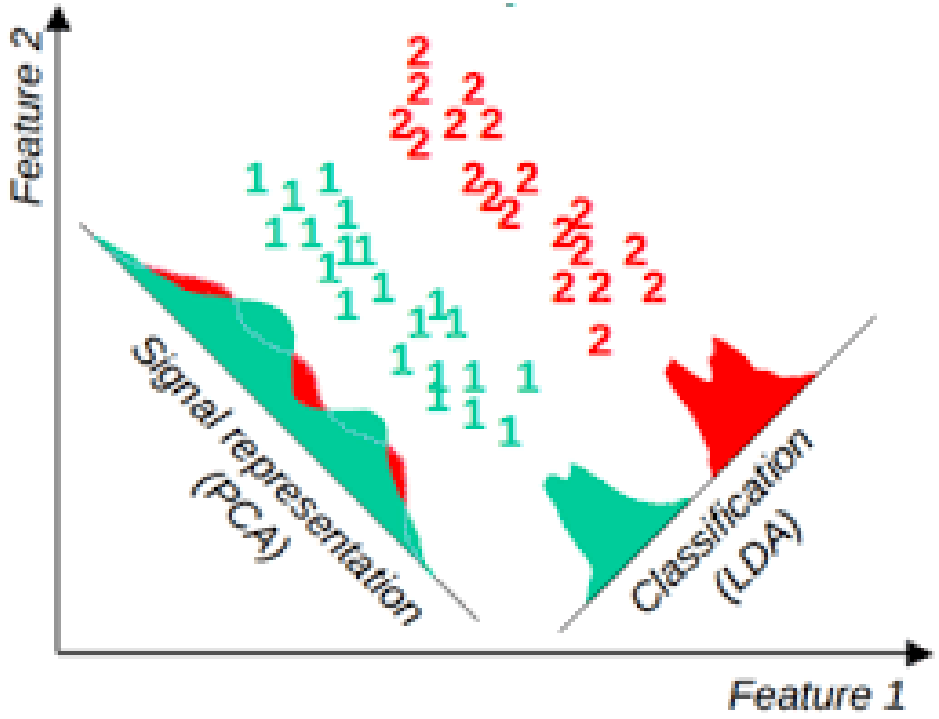
based on the number of variables considered together:

- **Univariate methods, variable ranking:**
  consider the input variables (features, attributes) one by one.
- **Multivariate methods, variable subset selection:**
  consider whole groups of variables together.

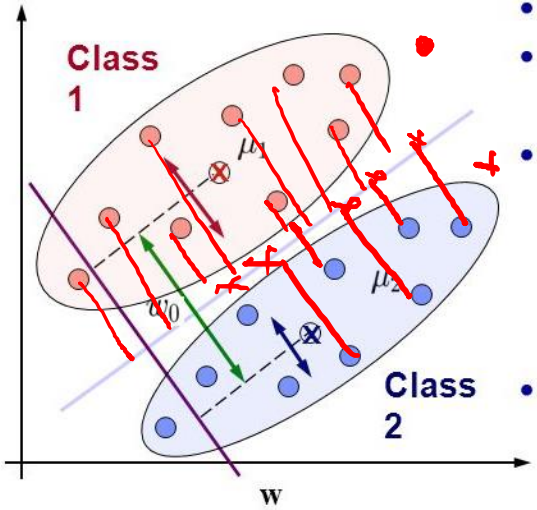
based on the use of the ML model in the feature selection process:

- **Filter:** selects a subset of variables independently of the model that shall subsequently use them.
- **Wrapper:** selects a subset of variables taking into accoiunt the model that shall use them.
- **Embedded method:** the feature selection method is built in the ML model (or rather its training algorithm) itself (e.g. decision trees).

Linear Discriminant Analysis (LDA)



Linear Discriminant Analysis (LDA)



- Maximize distance between classes
- Minimize distance within a class
- Criterion: $J(\mathbf{w}) = \frac{\mathbf{w}^T S_b \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}$

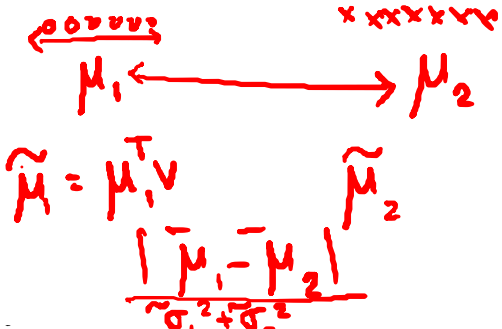
S_b ... between-class scatter matrix
S_w ... within-class scatter matrix

- Vector w is a solution of a generalized eigenvalue problem:

$$S_b \mathbf{w} = \lambda S_w \mathbf{w}$$

- Classification function:

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 \underset{\text{Class 2}}{\overset{\text{Class 1}}{\gtrless}} 0$$



mean vector for each class: $\mathbf{m}_c = \frac{\sum_{i=1}^{N_c} \mathbf{x}_i}{N_c} \in R^{d \times 1}$

total mean vector: $\mathbf{m} = \frac{\sum_{i=1}^{N} \mathbf{x}_i}{N} \in R^{d \times 1}$

within-class scatter: $S_w = \sum_{c=1}^{C} \sum_{j=1}^{N_c} (\mathbf{x}_j - \mathbf{m}_c)(\mathbf{x}_j - \mathbf{m}_c)^T \in R^{d \times d}$

between-class scatter: $S_b = \sum_{c=1}^{C} N_c (\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^T \in R^{d \times d}$

Slide credit: Ales Leonardis

LDA Algorithm

Require: data matrix $X \in R^{d \times N}$, $\mathbf{x}_i \in R^{d \times 1}$ is the i-th column of X. label vector $y_i \in \{1, 2, \cdots, C\}, i = 1, \cdots, N$, $N_c, c = 1, \cdots, C$ is the number of samples of each class.

Ensure: The projection matrix $P \in R^{p \times d}$

1: compute mean vector for each class: $\mathbf{m}_c = \frac{\sum_{i=1}^{N_c} \mathbf{x}_i}{N_c} \in R^{d \times 1}$

2: compute total mean vector: $\mathbf{m} = \frac{\sum_{i=1}^{N} \mathbf{x}_i}{N} \in R^{d \times 1}$

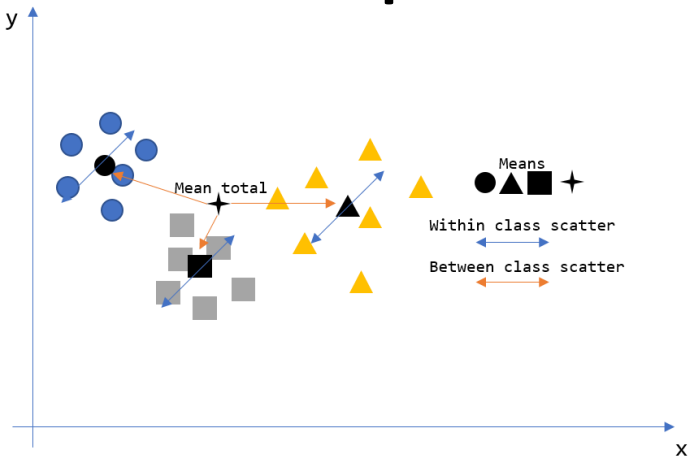
3: compute within-class scatter: $S_w = \sum_{c=1}^{C} \sum_{j=1}^{N_c} (\mathbf{x}_j - \mathbf{m}_c)(\mathbf{x}_j - \mathbf{m}_c)^T \in R^{d \times d}$

4: compute between-class scatter: $S_b = \sum_{c=1}^{C} N_c (\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^T \in R^{d \times d}$

5: eigen-decomposition: $[V, D] = eig({S_w}^{-1} S_b)$

6: get the projection matrix: P is composed of the top-p eigenvectors corresponding to the largest eigenvalues.

Multiple Discriminant Analysis (MDA)



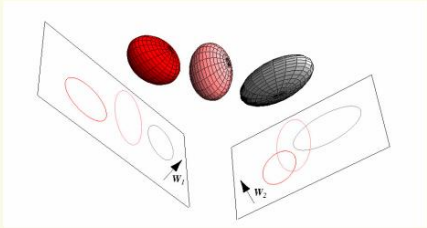
mean vector for each class: $\mathbf{m}_c = \frac{\sum_{i=1}^{N_c} \mathbf{x}_i}{N_c} \in R^{d\times 1}$

total mean vector: $\mathbf{m} = \frac{\sum_{i=1}^{N} \mathbf{x}_i}{N} \in R^{d\times 1}$

within-class scatter: $S_w = \sum_{c=1}^{C} \sum_{j=1}^{N_c} (\mathbf{x}_j - \mathbf{m}_c)(\mathbf{x}_j - \mathbf{m}_c)^T \in R^{d\times d}$

between-class scatter: $S_b = \sum_{c=1}^{C} N_c(\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^T \in R^{d\times d}$

- Can generalize FLD to multiple classes
- In case of ***c*** classes, can reduce dimensionality to 1, 2, 3,…, ***c***-1 dimensions
- Project sample $\boldsymbol{x_i}$ to a linear subspace $\boldsymbol{y_i = V^t x_i}$
  - ***V*** is called projection matrix



Finding random projections

- Vectors $x_i \epsilon R^d$, are projected onto a **k**-dimensional space (**k<<d**)
- Random projections can be represented by linear transformation matrix **R**
- $y_i = R\, x_i$

- What is the matrix **R**?

Finding matrix R

- Elements **R(i,j)** can be Gaussian distributed
- Achlioptas* has shown that the Gaussian distribution can be replaced by

$$R(i,j) = \begin{cases} +1 \text{ with prob } \frac{1}{6} \\ 0 \text{ with prob } \frac{2}{3} \\ -1 \text{ with prob } \frac{1}{6} \end{cases}$$

- All zero mean, unit variance distributions for **R(i,j)** would give a mapping that satisfies the ***JL*** lemma

Feature Selection - approaches

based on the number of variables considered together:

- **Univariate methods, variable ranking:**
  consider the input variables (features, attributes) one by one.
- **Multivariate methods, variable subset selection:**
  consider whole groups of variables together.

based on the use of the ML model in the feature selection process:

- **Filter:** selects a subset of variables independently of the model that shall subsequently use them.
- **Wrapper:** selects a subset of variables taking into accoiunt the model that shall use them.
- **Embedded method:** the feature selection method is built in the ML model (or rather its training algorithm) itself (e.g. decision trees).

References

- Duda and Hart, Section 3.8
- PRML (Bishop) : 12.1
- A visualization-based understanding of eigenvectors, eigenvalues and PCA
  - http://setosa.io/ev/eigenvectors-and-eigenvalues/
  - http://setosa.io/ev/principal-component-analysis/
- PCA : https://hadrienj.github.io/posts/Deep-Learning-Book-Series-2.12-Example-Principal-Components-Analysis/
- Required math background
  - Vector Derivatives:
    - http://15462.courses.cs.cmu.edu/fall2018content/lectures/03_vectorcalc/images/slide_032.jpg
  - Lagrange Multipliers:
    - Basic Idea (with example) : https://www.quora.com/What-is-happening-intuitively-when-using-Lagrange-multipliers-in-constrained-optimization/answer/Balaji-Pitchai-Kannu