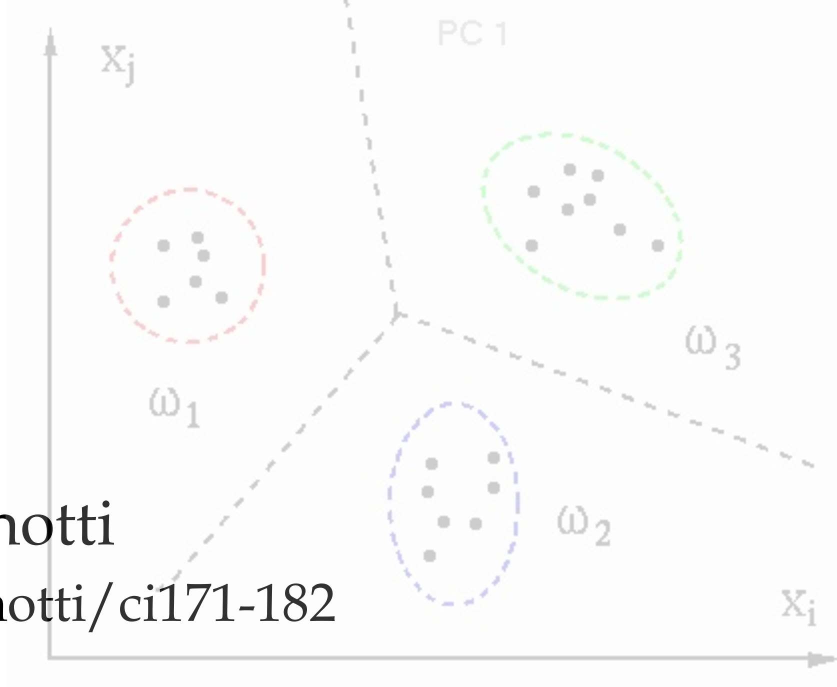
Universidade Federal do Paraná (UFPR)
Bacharelado em Informática Biomédica

Classificação



David Menotti
www.inf.ufpr.br/menotti/ci171-182

Hoje

- Redução de Dimensionalidade
  - *Principal Component Analysis* (PCA)
  - *Linear Discriminant Analysis* (LDA)

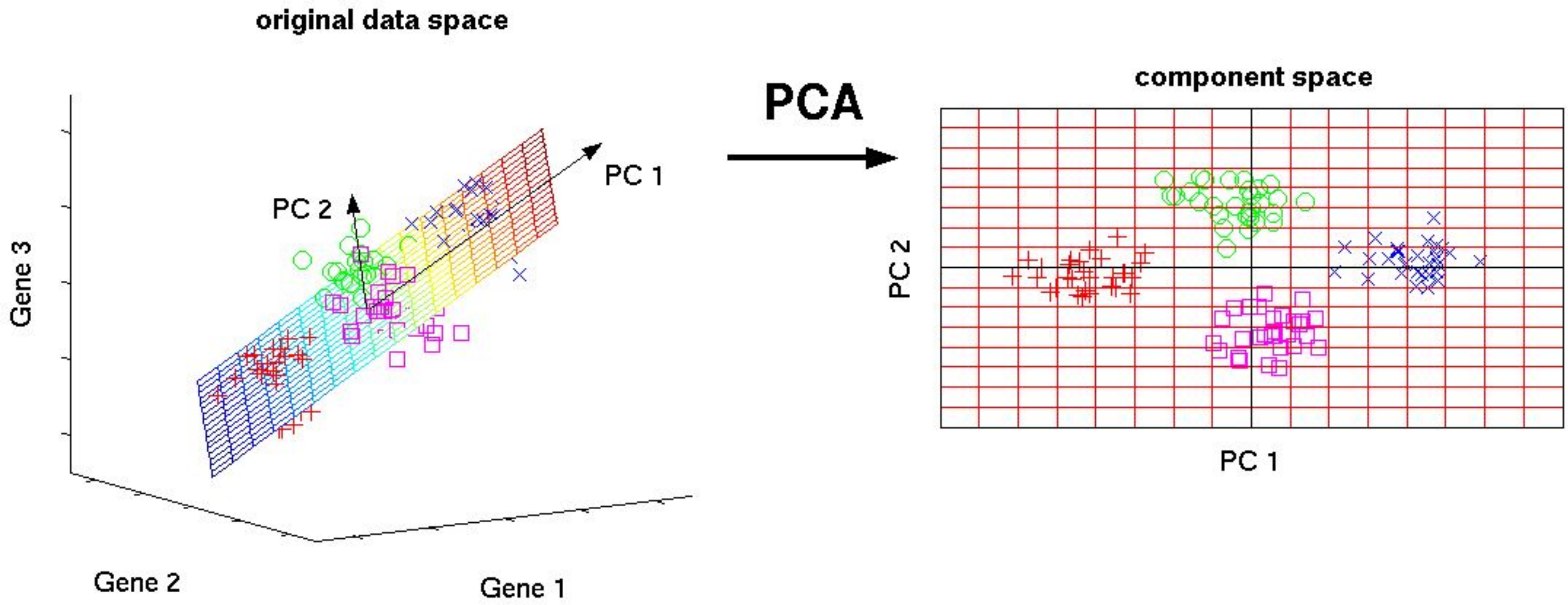
Redução de Dimensionalidade

Objetivos

- Introduzir os conceitos de PCA e suas aplicações para extração/redução de características
  - Revisão dos conceitos básicos de estatística e álgebra linear.
- Apresentar a *Linear Discriminant Analysis*
  - Técnica Supervisionada para redução de dimensionalidade

PCA

- Reduzir dimensionalidade **explicando**
  - Não supervisionado



Estatística

- Variância
  - Variância de uma variável aleatória é uma medida de dispersão estatística, indicando quão longe em geral os seus valores se encontram do valor esperado.

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2 \,,$$

  - Desvio padrão é a raiz da variância
    - O resultado do desvio se dá na mesma medida dos dados da população ou amostra.

Estatística

- Covariância
  - Variância é uma medida unidimensional.
  - É calculada de maneira independente pois não leva em consideração as outras dimensões.
  - Covariância por sua vez, é uma medida bi-dimensional. Verifica a dispersão, mas levando em consideração duas variáveis aleatórias.

$$cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

Estatística

- Matriz de covariância
  – Para 3 variáveis aleatórias, x, y e z, o cálculo de todas as covariâncias (x-y, x-z e y-z) pode ser acomodada em uma matriz, a qual denomina-se matriz de covariância.

$$C = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix}$$

Cov(x,y) = cov(y,x)

Cov(z,z) = var(z)

Álgebra

- Autovetores
  - Como sabe-se duas matrizes podem ser multiplicadas se elas possuem tamanhos compatíveis. Autovetores são casos especiais neste contexto.

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} 11 \\ 5 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 12 \\ 8 \end{pmatrix} = 4 \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} \longrightarrow \text{Múltiplo do vetor resultante}$$

Autovetores

- Nesse caso (3,2) representa um vetor que aponta da origem (0,0) para o ponto (3,2).
- A matriz quadrada, pode ser vista como uma matriz de transformação.
- Se esta matriz for multiplicada por outro vetor, a resposta será outro vetor transformado da sua posição original.
- É da natureza desta transformação que surgem os autovetores.

Autovetores

- Propriedades
  - Podem ser achados somente em matrizes quadradas.
  - Nem todas as matrizes possuem autovetores.
  - Para uma dada *n* x *n* matriz, existem *n* autovetores.
  - Se o vetor for multiplicado por uma constante, ainda obteremos o mesmo resultado

$$2 \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 6 \\ 4 \end{pmatrix} = \begin{pmatrix} 24 \\ 16 \end{pmatrix} = 4 \times \begin{pmatrix} 6 \\ 4 \end{pmatrix}$$

Apenas fazemos o vetor mais longo, mas não mudamos a direção.

Autovetores/Autovalores

- Todos os **autovetores** são ortogonais (perpendiculares), ou seja os dados podem ser expressos em termos destes vetores.

- O valor pelo qual o vetor é multiplicado é conhecido como **autovalor**
  - Um autovetor sempre possui um autovalor associado.

Definições

- Seja *A* uma matriz de ordem *n* x *n*
- O número **λ** é o **autovalor** (*eigenvalue*) de ***A*** se existe um vetor não-zero ***v*** tal que

$$\boldsymbol{A\,v = \lambda\,v}$$

- Neste caso, o vetor **v** é chamado de **autovetor** (*eigenvector*) de ***A*** correspondente à **λ**.

Calculando Autovalores e Autovetores

- Pode-se reescrever a condição:

$$\boldsymbol{A}\,\boldsymbol{v} = \boldsymbol{\lambda}\,\boldsymbol{v}$$

como

$$(\,A - \boldsymbol{\lambda}\, I\,)\,\boldsymbol{v} = \mathbf{0}$$

onde *I* é a matriz identidade de ordem *n* x *n.*

- Para que um vetor não-zero ***v*** satisfaça a equação, $(\,A - \boldsymbol{\lambda}\, I\,)$ deve ser **não** inversível.

Calculando Autovalores e Autovetores

- Caso contrário, se $(\boldsymbol{A} - \boldsymbol{\lambda}\, \boldsymbol{I})$ tiver uma inversa, então

$$(\boldsymbol{A} - \boldsymbol{\lambda}\, \boldsymbol{I})^{-1} (\boldsymbol{A} - \boldsymbol{\lambda}\, \boldsymbol{I})\, \boldsymbol{v} = (\boldsymbol{A} - \boldsymbol{\lambda}\, \boldsymbol{I})^{-1}\, \mathbf{0}$$

$$\boldsymbol{v} = \mathbf{0}$$

- Mas, procura-se por um vetor **v** não-zero.

Calculando Autovalores e Autovetores

- Voltando, isto é, o determinante de $(\boldsymbol{A} - \boldsymbol{\lambda}\, \boldsymbol{I})$ deve ser igual à 0.
- Chama-se

$$p(\boldsymbol{\lambda}) = \det(\boldsymbol{A} - \boldsymbol{\lambda}\, \boldsymbol{I})$$

de **polinômio característico** de ***A***.

- Os autovalores de ***A*** são as raízes do polinômio característico de ***A***.

Calculando Autovalores e Autovetores

- Para se calcular o i-ésimo autovetor $\boldsymbol{v}_{\mathbf{i}} = [\ \boldsymbol{v}_{\mathbf{1}}\ ;\ \boldsymbol{v}_{\mathbf{2}}\ ;\ \ldots\ ;\ \boldsymbol{v}_{\mathbf{n}}\ ]$

correspondente à um autovalor $\boldsymbol{\lambda}_{\mathbf{i}}$, basta resolver o sistema linear de equações dado por

$$(\boldsymbol{A} - \boldsymbol{\lambda}\, \boldsymbol{I})\, \boldsymbol{v} = 0$$

Análise dos Componentes Principais (PCA)

- Uma maneira de identificar padrões em dados, colocando em evidência suas similaridades e diferenças.
- Ferramenta importante para altas dimensões, onde não podemos fazer uma análise visual.
- Uma vez encontrados esses padrões, podemos comprimir os dados sem grande perda de qualidade.
- Extrator de características (representação)

PCA Tutorial

- 1) Escolha um conjunto de dados.
- 2) Normalize esses dados,
  - subtraindo-os da média.

Dados

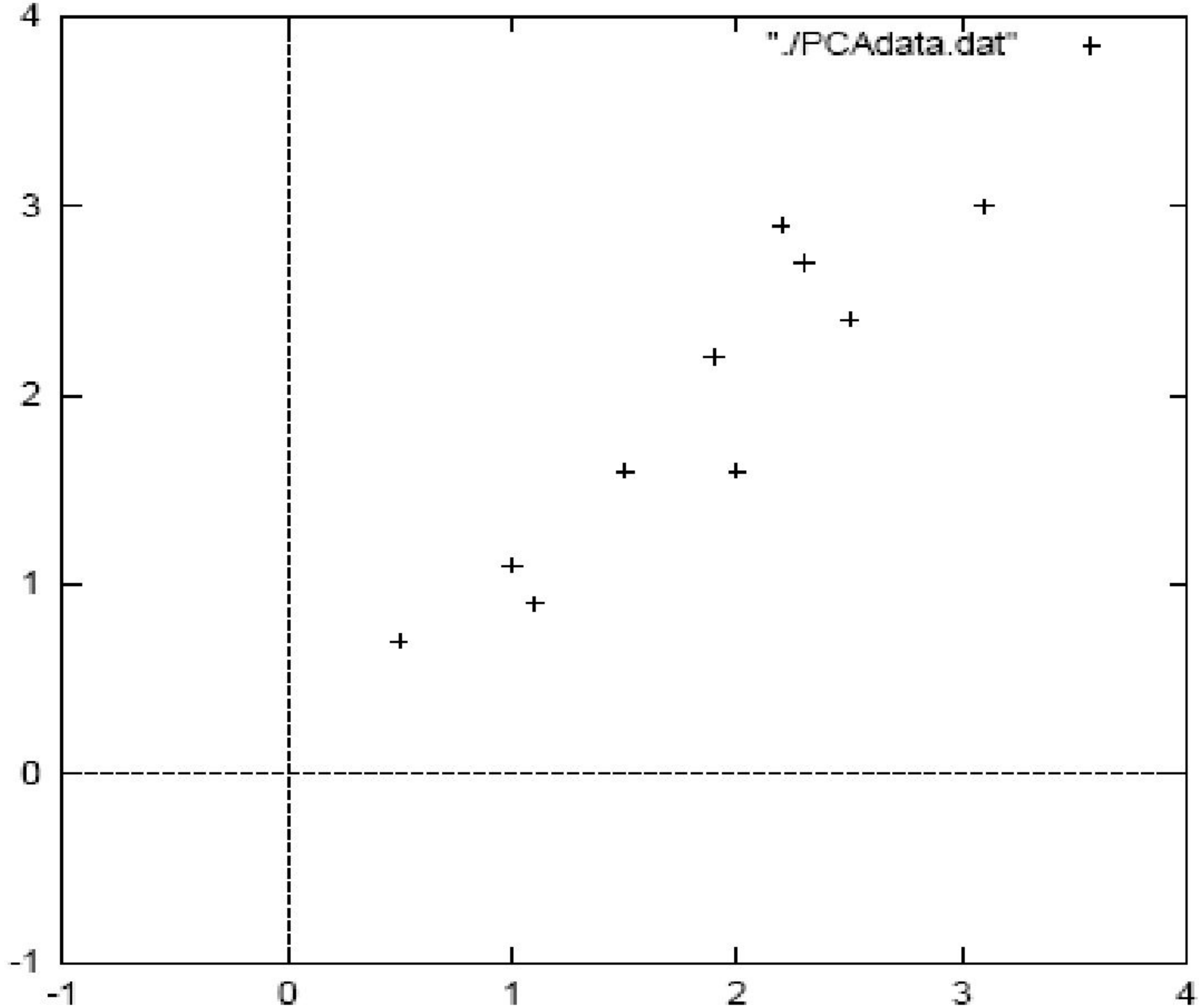
x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

Dados Normalizados

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

PCA Tutorial

x = 1,81
y = 1,91



PCA Tutorial

- 3) Calcule a matriz de correlação para os dados normalizados.
  - Uma vez que os dados possuem duas dimensões, teremos uma matriz 2x2

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

PCA Tutorial

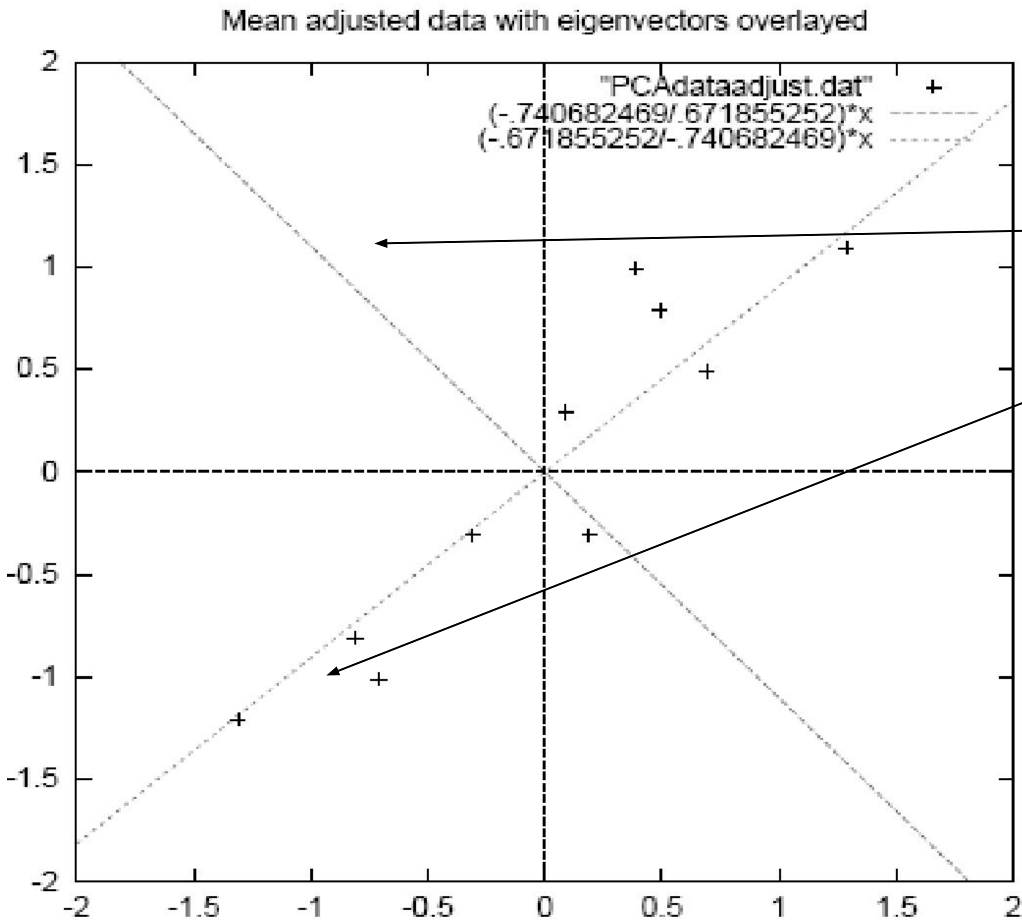
- 4) Encontre os autovetores e autovalores para a matriz de covariância.
  – Uma vez que a matriz de covariância é quadrada podemos encontrar os autovetores e autovalores.

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

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

O que esses valores significam ???

PCA Tutorial



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

PCA Tutorial

- 5) Escolhendo os componentes que vão formar o vetor
  - Como vimos, os autovalores são bastante diferentes.
  - Isso permite ordenar os autovetores por ordem de importância.
  - Se quisermos eliminar um componente, devemos então eliminar os que têm menos importância.

$$FeatureVector = (eig_1\ eig_2\ eig_3\ ....\ eig_n)$$

PCA Tutorial

- No nosso exemplo temos duas escolhas
  - Manter os dois.
  - Eliminar um autovetor, diminuindo assim a dimensionalidade dos dados
    - Maldição da dimensionalidade
    - Quanto maior a dimensionalidade do seu vetor, mais dados serão necessários para a aprendizagem do modelo.

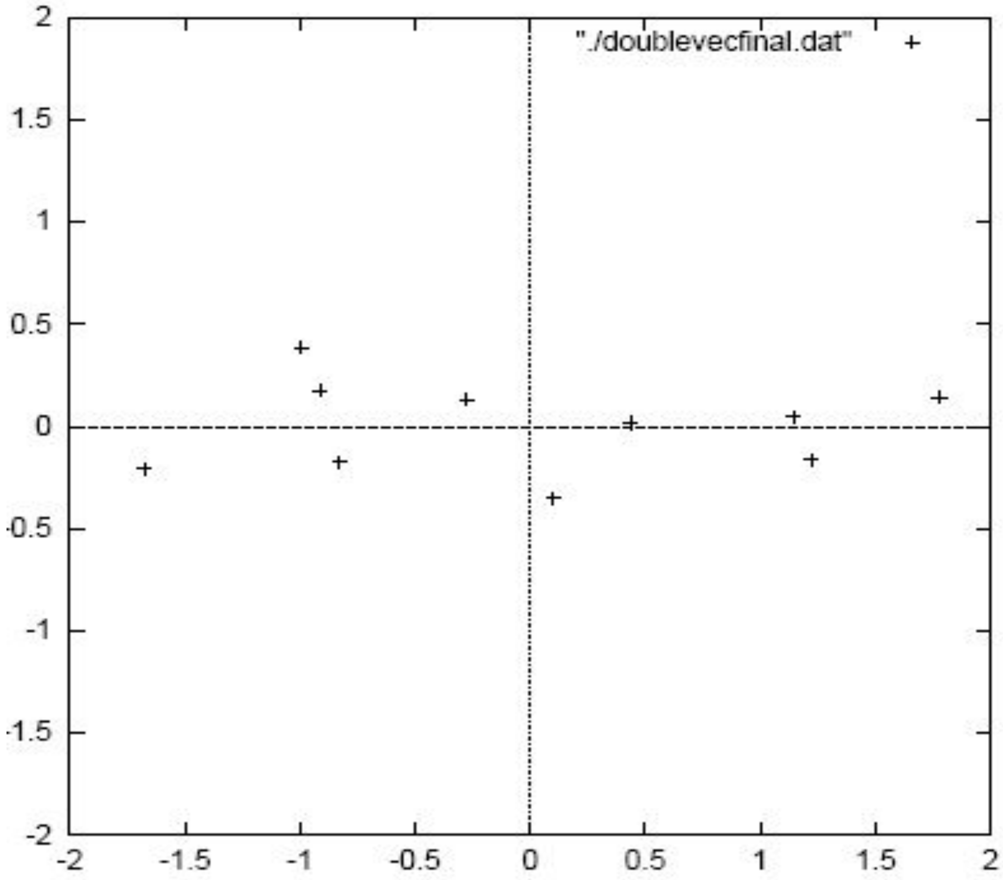
PCA Tutorial

- 6) Construindo novos dados.
  - Uma vez escolhidos os componentes (autovetores), nós simplesmente multiplicamos os dados pelo autovetor(es) escolhidos.
  - O que temos?
    - Dados transformados de maneira que expressam os padrões entre eles.
    - Os PCs (*Principal Components*) são combinações lineares de todas as características, produzindo assim novas características não correlacionadas.

PCA Tutorial

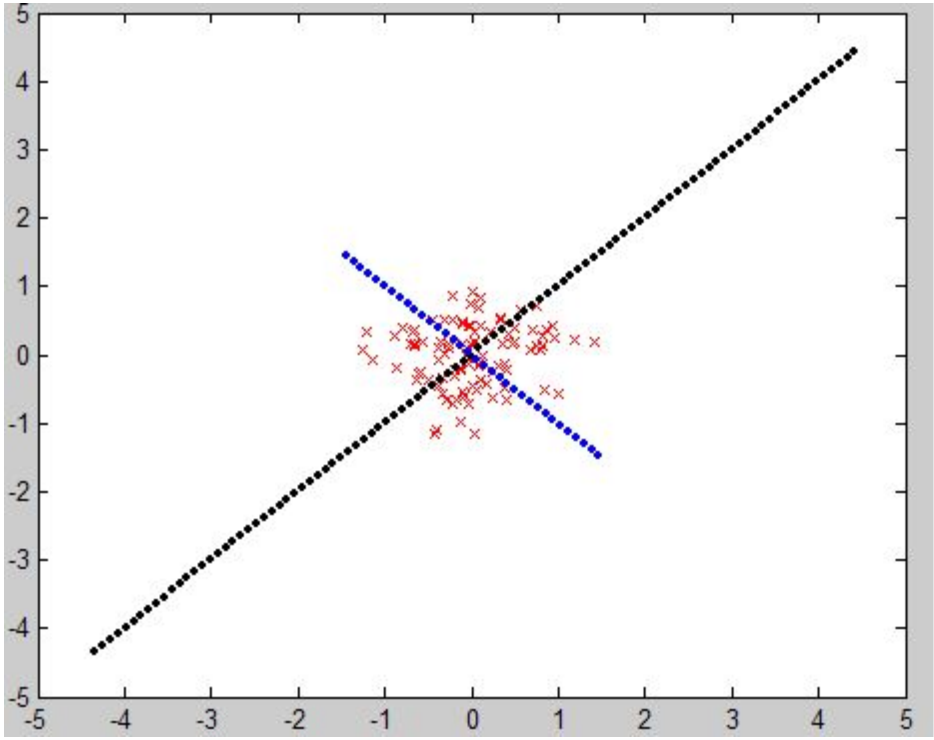
Dados transformados
usando 2 autovetores

x	y
-.827970186	-.175115307
1.77758033	.142857227
-.992197494	.384374989
-.274210416	.130417207
-1.67580142	-.209498461
-.912949103	.175282444
.0991094375	-.349824698
1.14457216	.0464172582
.438046137	.0177646297
1.22382056	-.162675287



PCA Tutorial

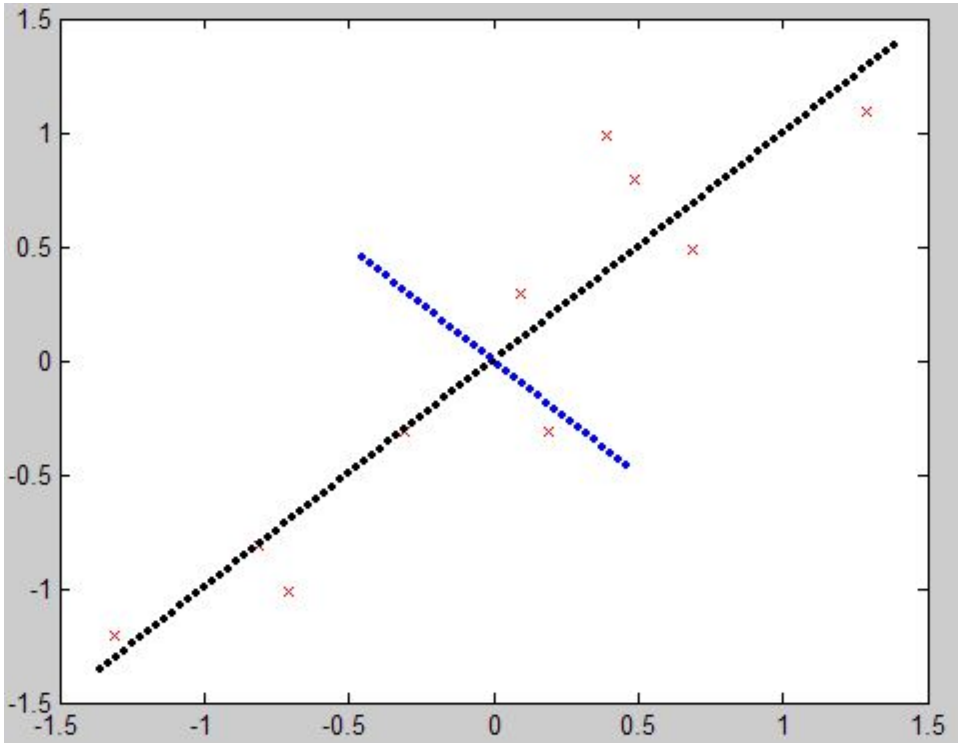
- Exemplo



Usando a função **pcacov** do Matlab, três parâmetros são retornados.
-autovetores
-autovalores
-percentagem da variância total explicada para cada modelo

PCA Tutorial

- Exemplo 2



AUTOVETORES

-0.6779	-0.7352
-0.7352	0.6779

AUTOVALORES

1.2840
0.0491

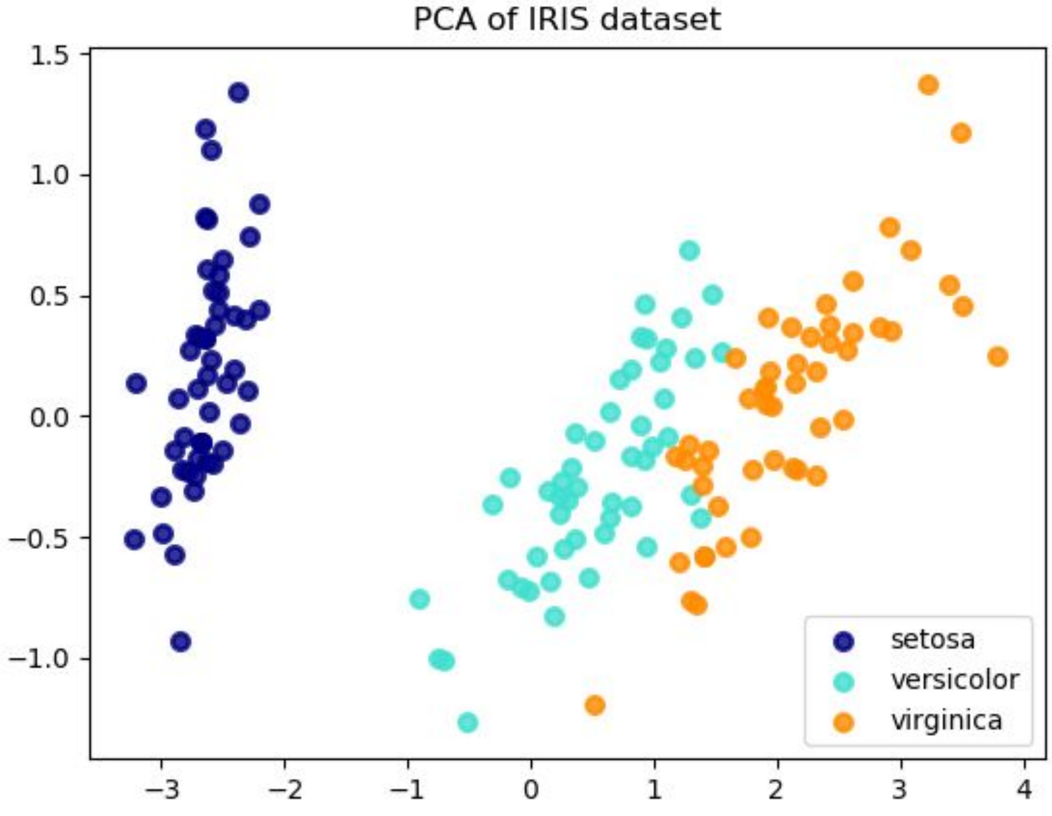
VARIÂNCIA EXPLICADA

96.3181
3.6819

PCA

Exemplo - Iris Database

- Dois componentes ([0.92461621 0.05301557]



PCA Tutorial

- Exercício
  - Gere diferentes conjuntos de dados em duas dimensões e aplique PCA. Verifique e discuta a variância explicada em função da dispersão dos dados.

PCA Tutorial 2 (1/2)

- `t = 2*pi*rand(1,1000);`
- `rho = rand(1,1000);`
- `xc=3;yc=3;`
- `a=3;b=2;phi=atan(2/3);`

- `x = xc + (rho*a) .* cos(t) * cos(phi) - (rho*b) .* sin(t) * sin(phi);`
- `y = yc + (rho*a) .* cos(t) * sin(phi) + (rho*b) .* sin(t) * cos(phi);`

- `%data = [x'-repmat(mean(x),1000,1) y'-repmat(mean(y),1000,1)];`
- `data = [x' y'];`
- `covdata = cov(data);`

- `[autovetor,score,autovalor] = princomp(data);`

PCA Tutorial 2 (2/2)

- `%% cov(data) * autovetor(:,1) % A * v1`
- ` % lambda * v1`
- `fprintf(1,'v1 = (%7.4f,%7.4)\n',autovalor(1) * autovetor(1,1),autovalor(1) * autovetor(1,2));`

- `%%cov(data) * autovetor(:,2) % A * v2`
- `% lambda * v2`
- `fprintf(1,'v2 = (%7.4f,%7.4)\n',autovalor(1) * autovetor(2,1),autovalor(1) * autovetor(2,2));`

- `hold;`
- `axis([0 6 0 6]);`
- `plot(x,y,'o');`
- `plot([xc xc+autovalor(1) * autovetor(1,1)],[yc yc+autovalor(1) * autovetor(2,1)],'r-d');`
- `plot([xc xc+autovalor(2) * autovetor(1,2)],[yc yc+autovalor(2) * autovetor(2,2)],'r-d');`

Eigen Faces

Utilizar PCA para o reconhecimento de faces

PCA Faces

$$= \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{N^2} \end{pmatrix}$$

$$= \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{N^2} \end{pmatrix}$$

$$= \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{N^2} \end{pmatrix}$$

$$= \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_{N^2} \end{pmatrix}$$

$$= \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N^2} \end{pmatrix}$$

$$= \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_{N^2} \end{pmatrix}$$

$$= \begin{pmatrix} i_1 \\ i_2 \\ \vdots \\ i_{N^2} \end{pmatrix}$$

PCA Faces

- Calcula-se a face média

$$\vec{m} = \frac{1}{M}\begin{pmatrix} a_1 & +b_2 & +\square & +h_1 \\ a_2 & +b_2 & +\square & +b_2 \\ \square & \square & \square & \square \\ a_{N^2} & +b_{N^2} & +\square & +h_{N^2} \end{pmatrix}, \text{onde } M = 8$$

PCA Faces

- Subtrair as faces media de cada imagem.

$$\vec{a}_m = \begin{pmatrix} a_1 - m_1 \\ a_2 - m_2 \\ \vdots \\ a_{N^2} - m_{N^2} \end{pmatrix} \quad \vec{b}_m = \begin{pmatrix} b_1 - m_1 \\ b_2 - m_2 \\ \vdots \\ b_{N^2} - m_{N^2} \end{pmatrix} \quad \vec{c}_m = \begin{pmatrix} c_1 - m_1 \\ c_2 - m_2 \\ \vdots \\ c_{N^2} - m_{N^2} \end{pmatrix} \quad \vec{d}_m = \begin{pmatrix} d_1 - m_1 \\ d_2 - m_2 \\ \vdots \\ d_{N^2} - m_{N^2} \end{pmatrix}$$

$$\vec{e}_m = \begin{pmatrix} e_1 - m_1 \\ e_2 - m_2 \\ \vdots \\ e_{N^2} - m_{N^2} \end{pmatrix} \quad \vec{f}_m = \begin{pmatrix} f_1 - m_1 \\ f_2 - m_2 \\ \vdots \\ f_{N^2} - m_{N^2} \end{pmatrix} \quad \vec{g}_m = \begin{pmatrix} g_1 - m_1 \\ g_2 - m_2 \\ \vdots \\ g_{N^2} - m_{N^2} \end{pmatrix} \quad \vec{h}_m = \begin{pmatrix} h_1 - m_1 \\ h_2 - m_2 \\ \vdots \\ h_{N^2} - m_{N^2} \end{pmatrix}$$

PCA Faces

- Agora, construímos a matriz N^2 X M, na qual cada coluna representa uma imagem.

$$A = \left[\vec{a_m}, \vec{b_m}, \vec{c_m}, \vec{d_m}, \vec{e_m}, \vec{f_m}, \vec{g_m}, \vec{h_m} \right]$$

- A matriz de covariância é

$$Cov = AA^T$$

PCA Faces

- Encontrar autovalores e autovetores para a matriz Cov
  - Matriz muito grande.
  - Custo computacional elevado.
- A dimensão da matriz pode ser reduzida calculando-se a matriz L, que tem o tamanho M X M.

$$L = A^T A$$

PCA Faces

- Encontrar os autovetores e autovalores de L (matriz *V*)
  - Os autovetores de L e Cov são equivalentes.
- A matriz de transformação é dada por

$$U = AV$$

PCA Faces

- Para cada face, calcular a sua projeção no espaço de faces

$$\Omega_1 = U^T\left(\vec{a}_m\right), \ \Omega_2 = U^T\left(\vec{b}_m\right), \ \cdots \qquad \Omega_8 = U^T\left(\vec{h}_m\right)$$

PCA Faces

- Reconhecendo uma face

$$= \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_{N^2} \end{pmatrix}$$

- Subtrair a face média da face de teste.

$$\vec{r}_m = \begin{pmatrix} r_1 - m_1 \\ r_2 - m_2 \\ \vdots \\ r_{N^2} - m_{N^2} \end{pmatrix}$$

PCA Face

- Calcular sua projeção no espaço de faces

$$\Omega = U^T \left(\vec{r}_m \right)$$

- Calcular a distância para todas as faces conhecidas

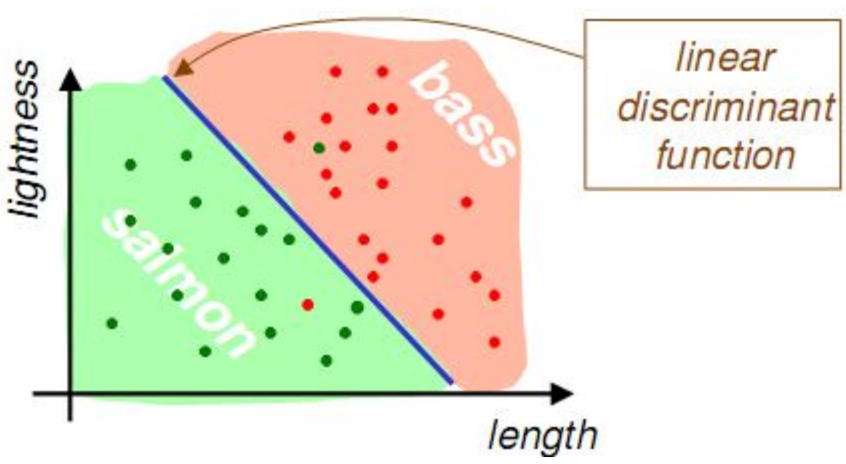
$$\varepsilon_i^2 = \left\| \Omega - \Omega_i \right\|^2, i = 1,2,\ldots,M.$$

- Atribui-se a imagem r a imagem com a menor distância,

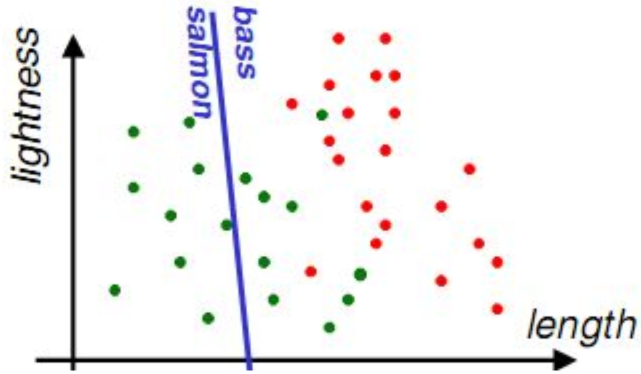
Linear Discriminant Analysis

Introdução

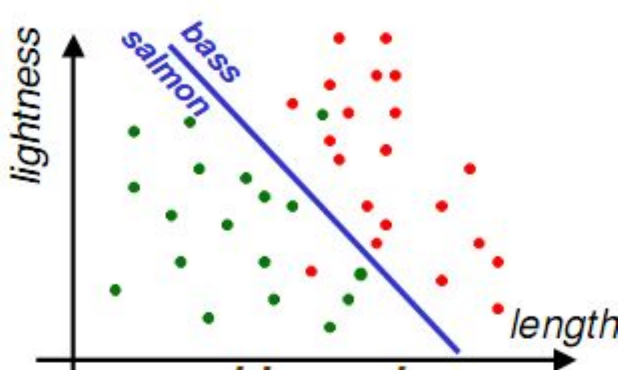
- Para utilizar uma **função discriminante linear** (*Linear Discriminant Function*) precisamos ter:
  - Dados rotulados (supervisão)
  - Conhecer o *shape* da fronteira
  - Estimar os parâmetros desta fronteira a partir dos dados de treinamento.
    - Nesse caso uma reta.



Introdução: Ideia Básica



Ruim



Boa

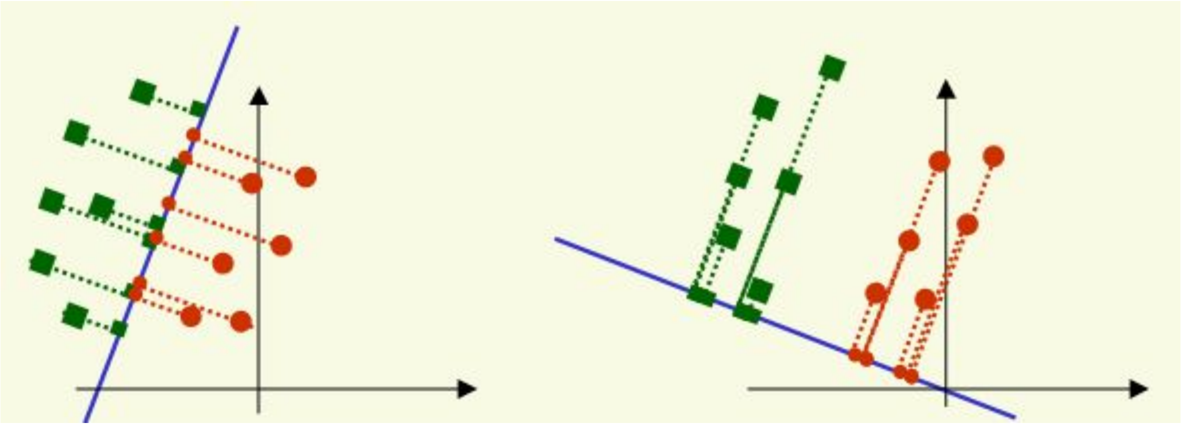
- Suponha duas classes
- Assuma que elas são linearmente separáveis por uma fronteira l(θ)
- Otimizar o parâmetro θ (não a distribuição) para encontrar a melhor fronteira.
- Como encontrar o parâmetro
  - Minimizar o erro no treinamento
  - O ideal é utilizar uma base de validação.

Introdução

- Funções discriminantes podem ser mais gerais do que lineares
  - Quadrática, Cúbicas, transcedentais, etc
- Vamos focar em problemas lineares
  - Mais fácil de compreender
  - Entender a base da classificação linear
- Diferentemente de métodos paramétricos, não precisamos conhecer a distribuição dos dados
  - Dessa forma, podemos dizer que temos uma abordagem não paramétrica.

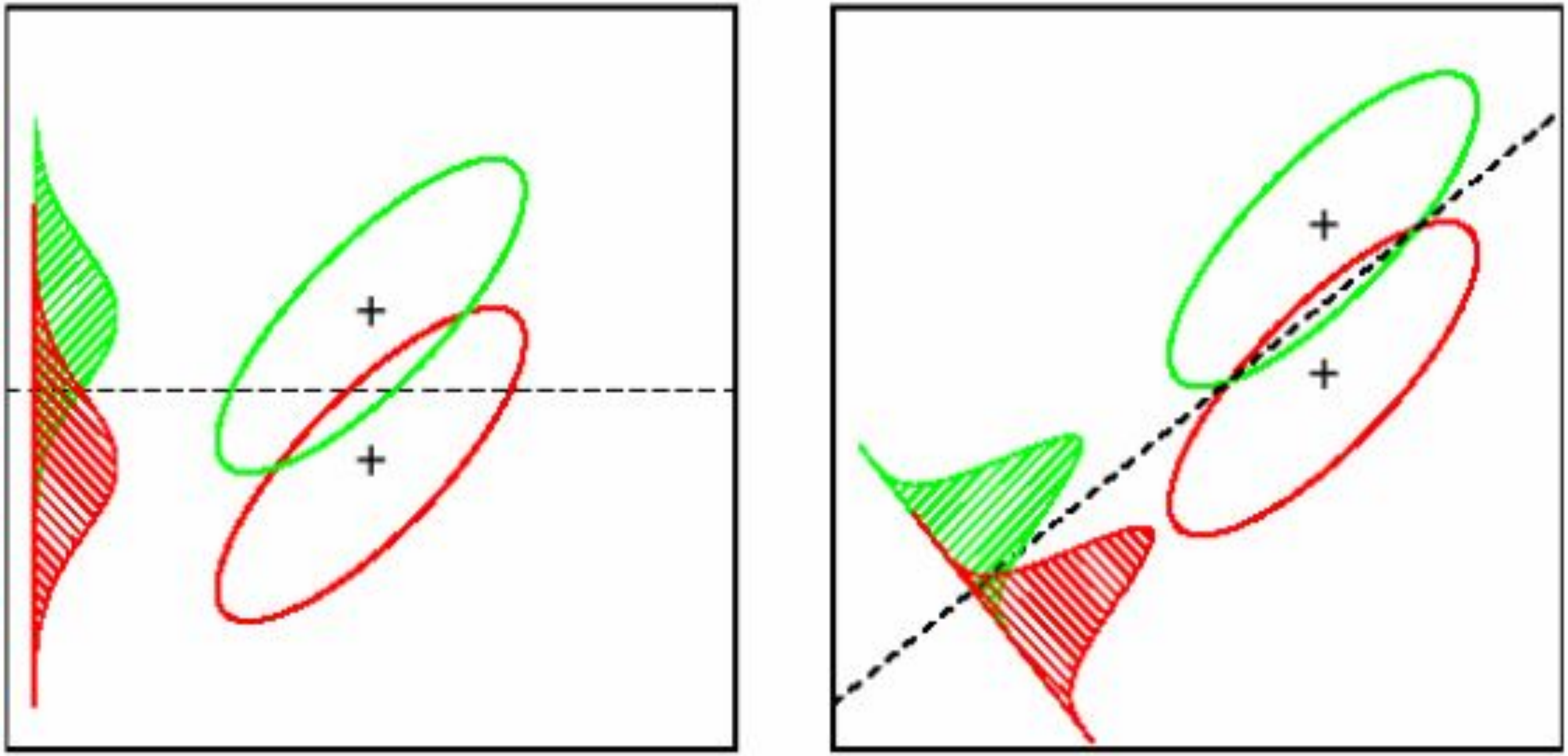
Análise Discriminante Linear

- LDA tenta encontrar uma transformação linear através da maximização da distância entre-classes e minimização da distância intra-classe.
- O método tenta encontrar a melhor direção de maneira que quando os dados são projetados em um plano, as classes possam ser separadas.



Reta ruim | Reta boa

LDA



LDA

- Projetar amostras

Maximizar inter-classe

$$y = \boldsymbol{w}\, x$$

Minimizar intra-classe

- Maximizar $J(w)$, qual $\boldsymbol{w}$?

$$J(w) = |\, m_1 - m_2 \,| \,/\, (\, s_1^2 + s_2^2 \,)$$

$$J(w) = (\, w^t S_B\, w \,) \,/\, (\, w^t S_w\, w \,)$$

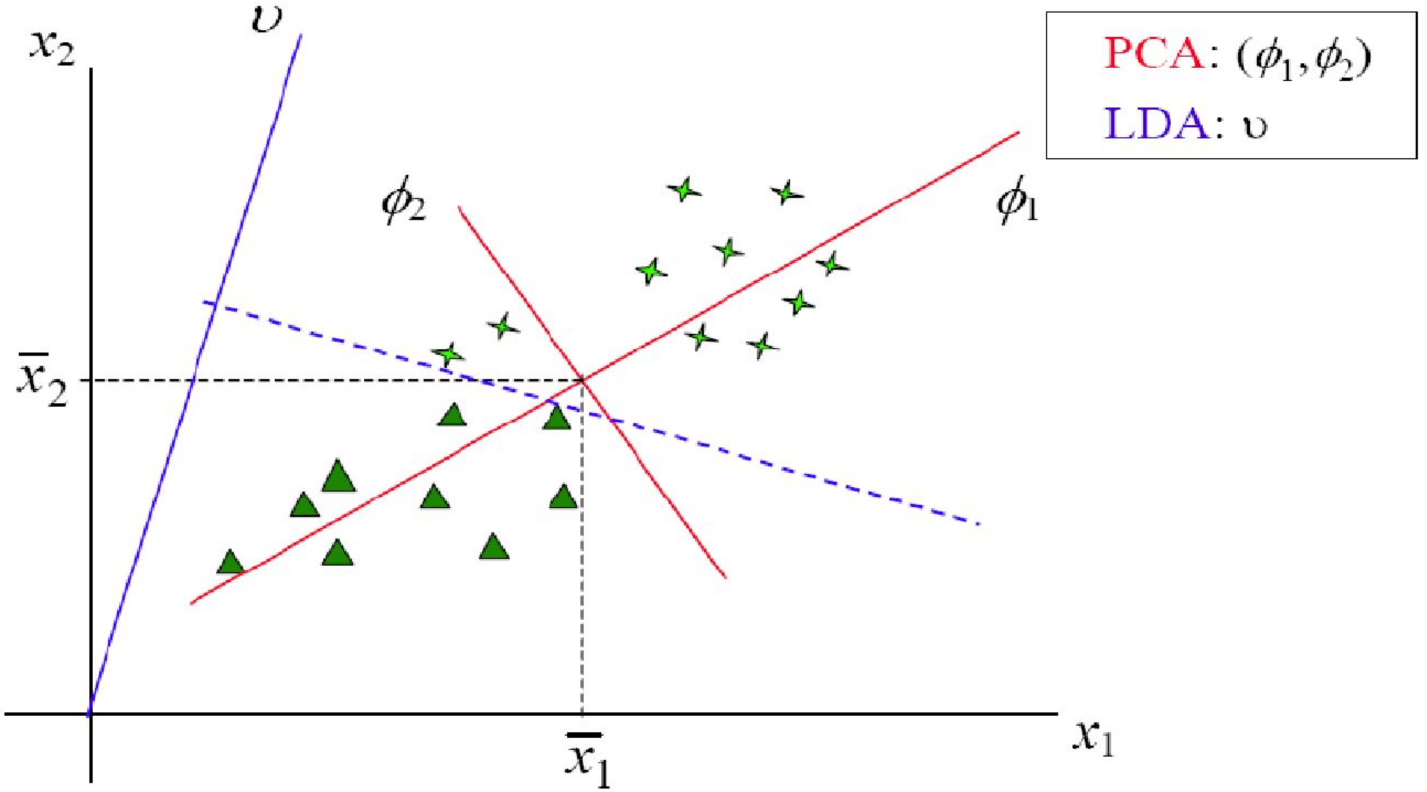
$$d\, J(w) \,/\, d\, w = 0 \Rightarrow S_B\, w \;-\; S_w\, \lambda w \;=\; 0$$

$$\Rightarrow S_B\, S_w^{-1}\, w \;-\; \lambda w \;=\; 0$$

$$\Rightarrow w = S_W^{-1} \;(\, m_1 - m_2 \;)$$

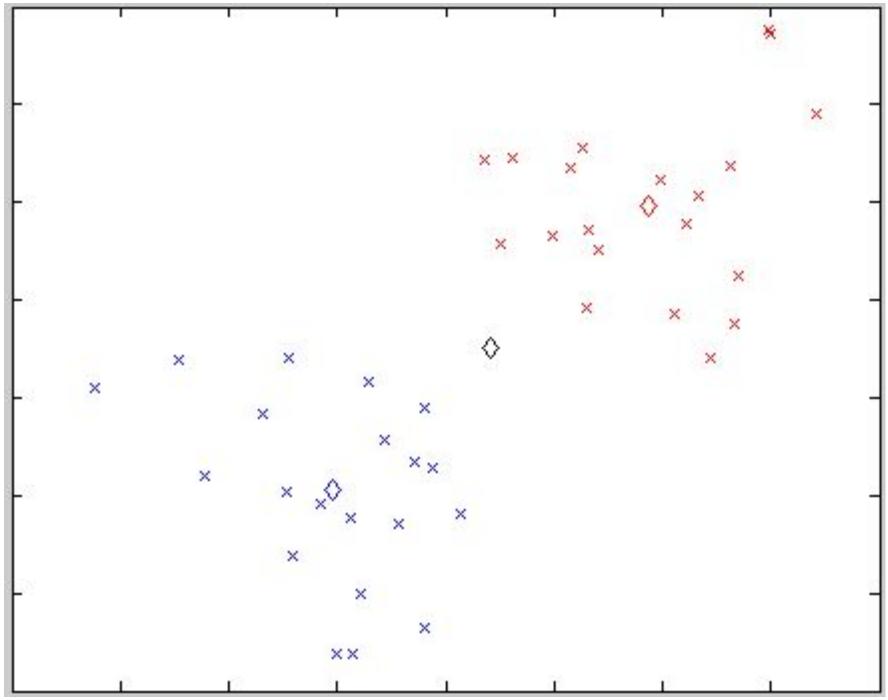
LDA

- Diferença entre PCA e LDA quando aplicados sobre os mesmos dados



LDA

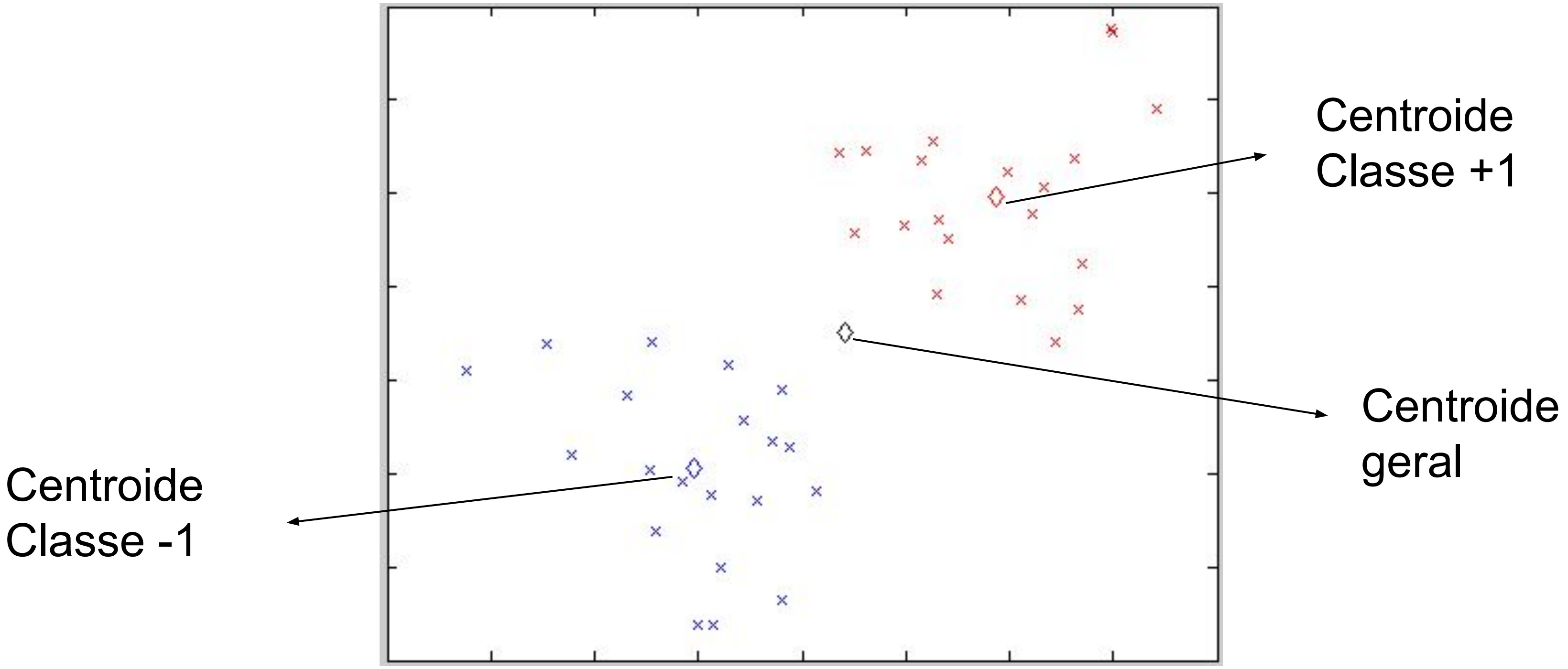
- Para um problema linearmente separável, o problema consiste em rotacionar os dados de maneira a maximizar a distância entre as classes e minimizar a distância intra-classe.



LDA

Exemplo

- 1) Para um dado conjunto de dados, calcule os vetores médios de cada classe μ_1 e μ_2 (centróides) e o vetor médio geral, μ.



LDA Tutorial

Exemplo - 7 pontos

- Calcular as médias de cada classe e a total.

$$\mathbf{x}_1 = \begin{bmatrix} 2.95 & 6.63 \\ 2.53 & 7.79 \\ 3.57 & 5.65 \\ 3.16 & 5.47 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 2.58 & 4.46 \\ 2.16 & 6.22 \\ 3.27 & 3.52 \end{bmatrix}$$

$$\boldsymbol{\mu}_1 = \begin{bmatrix} 3.05 & 6.38 \end{bmatrix}, \quad \boldsymbol{\mu}_2 = \begin{bmatrix} 2.67 & 4.73 \end{bmatrix}$$

$$\boldsymbol{\mu} = \begin{bmatrix} 2.88 & 5.676 \end{bmatrix}$$

LDA Tutorial

Exemplo

- Calcular o espalhamento de cada classe

$$S_i = \sum (m - x_i)(m - x_i)^t$$

- Calcular o espalhamento entre classes (*within class*)

$$S_W = S_1 + S_2$$

LDA

- Calcular a inversa de S_W
  - Custo???

- Finalmente, o vetor projeção

$$w = S_W^{-1} (m_1 - m_2)$$

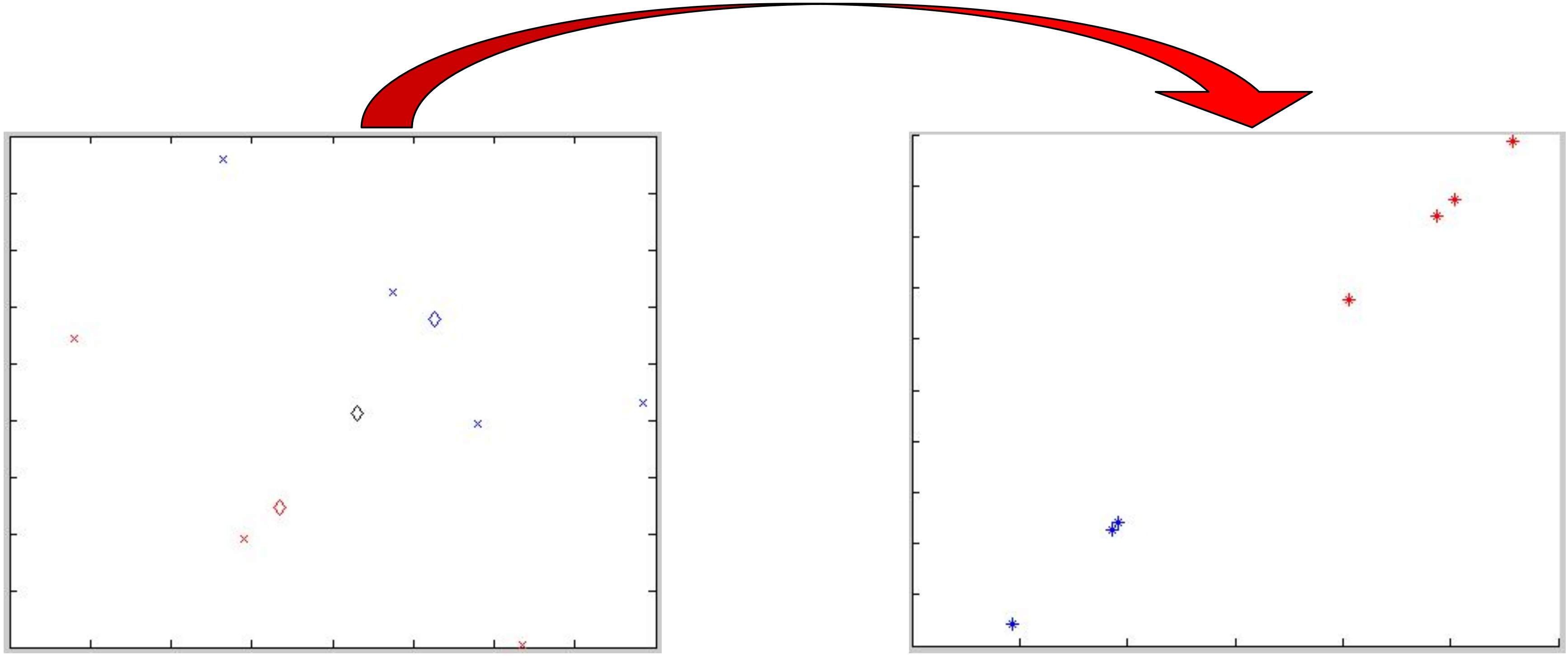
- Reprojetando os vetores sobre w

$$x_1' = (x_1 w) w^t$$
$$x_2' = (x_2 w) w^t$$

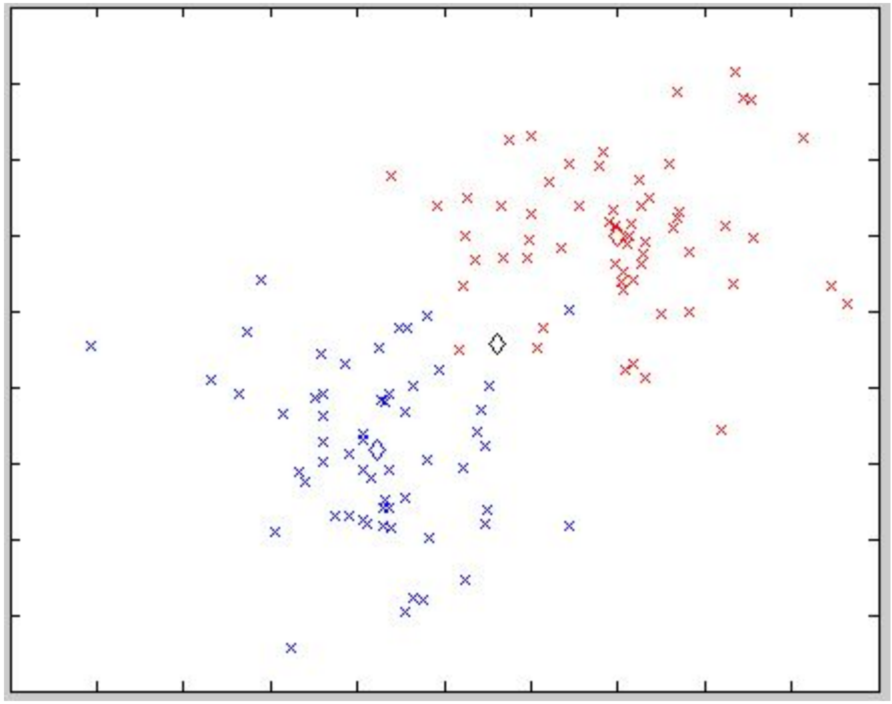
LDA Tutorial

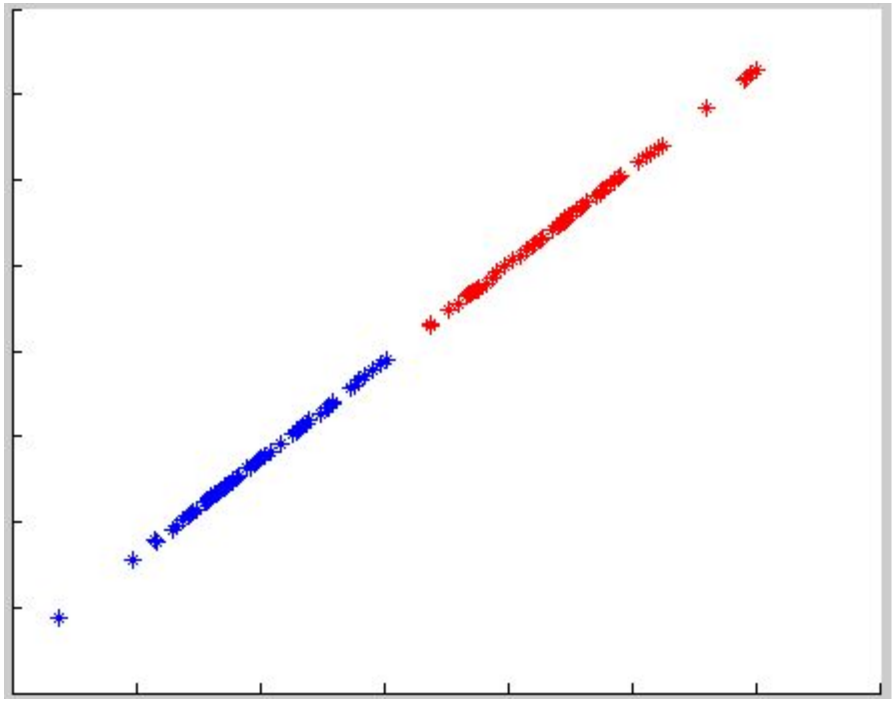
Exemplo

- Para visualizar a transformação, basta aplicar a função discriminante a todos os dados



LDA Tutorial





Taxa de Reconhecimento = 99%

Exercício

- Gere duas distribuições
- Classifique os dados usado LDA
- Verifique o impacto da sobreposição das distribuições.

Tutorial 1/2 - Matlab

```
• x1 = [ 2.95 6.63; 2.53 7.79; 3.57 5.65; 3.16 5.47];
• x2 = [2.58 4.46; 2.16 6.22; 3.27 3.52];
•
• m1 = mean(x1);m2 = mean(x2);m = mean([x1;x2]);
• S1 = (x1-repmat(m1,size(x1,1),1))'* ...
•       (x1-repmat(m1,size(x1,1),1));
• S2 = (x2-repmat(m2,size(x2,1),1))'* ...
•       (x2-repmat(m2,size(x2,1),1));

• S = S1 + S2;
• w=inv(S)*(m1-m2)';
```

Tutorial 2/2 - Matlab

- `figure,hold on`
- `axis([0 8 0 8]);`
- `plot(x1(:,1),x1(:,2),'bx');`
- `plot(m1(1),m1(2),'bd');`
- `plot(x2(:,1),x2(:,2),'rx');`
- `plot(m2(1),m2(2),'rd');`
- `plot(m(1),m(2),'kd');`
- `plot([w(1) 0],[w(2) 0],'g');`
- `w = w/norm(w);`
-
- `x1l=(x1*w)*w‘; x2l=(x2*w)*w';`
-
- `plot(x1l(:,1),x1l(:,2),'bo');`
- `plot(x2l(:,1),x2l(:,2),'ro');`

Tutorial 2 1/3

- ```
  a = 5*[randn(500,1)+5, randn(500,1)+5];
  b = 5*[randn(500,1)+5, randn(500,1)-5];
  c = 5*[randn(500,1)-5, randn(500,1)+5];
  d = 5*[randn(500,1)-5, randn(500,1)-5];
  e = 5*[randn(500,1), randn(500,1)];

  Group_X = [a;b;c];
  Group_Y = [d;e];

  All_data = [Group_X; Group_Y];
  All_data_label = [];

  for k = 1:length(All_data)
    if k<=length(Group_X)
      All_data_label = [All_data_label; 'X'];
    else
      All_data_label = [All_data_label; 'Y'];
    end
  end

  testing_ind = [];
  for i = 1:length(All_data)
    if rand>0.8
      testing_ind = [testing_ind, i];
    end
  end
  training_ind = setxor(1:length(All_data), testing_ind);
  ```
  ```

Tutorial 2 2/3

- ```
  [ldaClass,err,P,logp,coeff] = classify(All_data(testing_ind,:),...
  All_data((training_ind),:),All_data_label(training_ind,:),'linear');
  [ldaResubCM,grpOrder] = confusionmat(All_data_label(testing_ind,:),ldaClass)

  K = coeff(1,2).const;
  L = coeff(1,2).linear;
  f = @(x,y) K + [x y]*L;
  h2 = ezplot(f,[min(All_data(:,1)) max(All_data(:,1)) min(All_data(:,2))
  max(All_data(:,2))]);
  hold on


  [ldaClass,err,P,logp,coeff] = classify(All_data(testing_ind,:),...
  All_data((training_ind),:),All_data_label(training_ind,:),'diagQuadratic');
  [ldaResubCM,grpOrder] = confusionmat(All_data_label(testing_ind,:),ldaClass)

  K = coeff(1,2).const;
  L = coeff(1,2).linear;
  Q = coeff(1,2).quadratic;
  f = @(x,y) K + [x y]*L + sum(([x y]*Q) .* [x y], 2);
  h2 = ezplot(f,[min(All_data(:,1)) max(All_data(:,1)) min(All_data(:,2))
  max(All_data(:,2))]);
  hold on
  ```
  ```

Tutorial 2 3/3

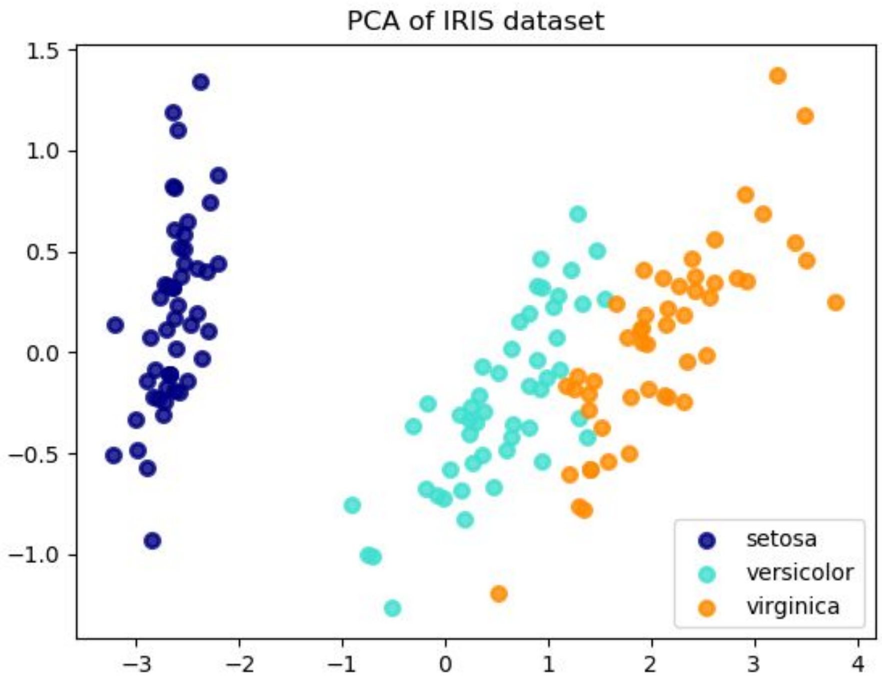
- ```
  Group_X_testing = [];
  Group_Y_testing = [];

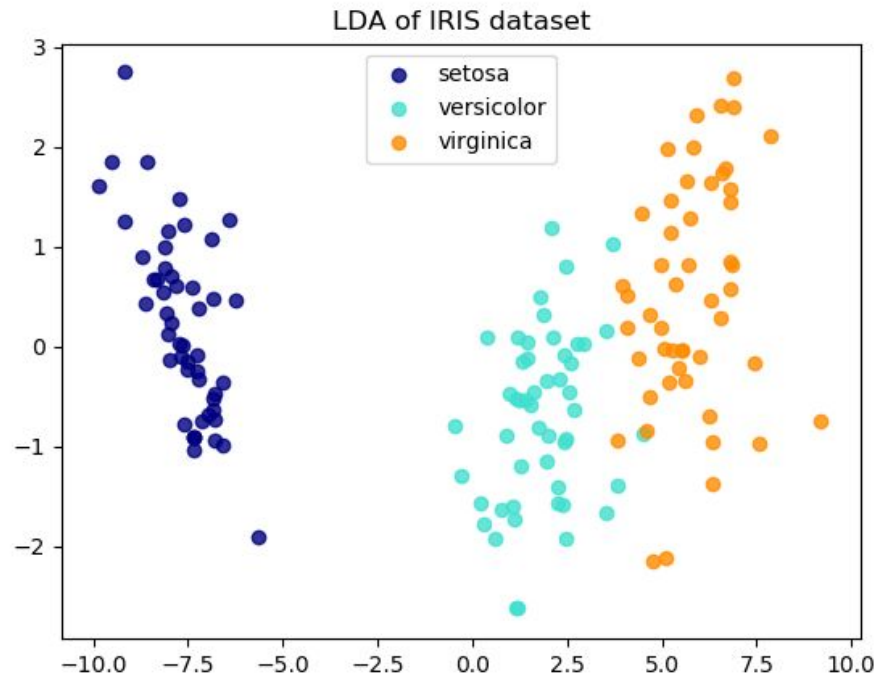
  for k = 1:length(All_data)
    if ~isempty(find(testing_ind==k))
      if strcmp(All_data_label(k,:),'X')==1
        Group_X_testing = [Group_X_testing,k];
      else
        Group_Y_testing = [Group_Y_testing,k];
      end
    end
  end
  plot(All_data(Group_X_testing,1),All_data(Group_X_testing,2),'g.');
  hold on
  plot(All_data(Group_Y_testing,1),All_data(Group_Y_testing,2),'r.');
  ```
  ```

LDA

Iris database (UCI Machine Learning Repository)

- PCA vs LDA





Referências