Redução de Dimensionalidade e Variáveis Latentes

1. Motivação

Neste tópico, vamos estudar algumas técnicas gerais e não-supervisionadas para redução de dimensionalidade ou extração de atributos (*feature extraction*). Em termos simples, o que se deseja obter é uma representação compacta para os dados observados que preserve a maior parte da informação ou da estrutura que os dados possuem no espaço original.

Vantagens:

- Reduzir o custo computacional envolvido no processamento dos dados.
- Eliminar redundâncias nas informações disponíveis.
- Possibilitar a visualização dos dados.

Uma maneira alternativa de se considerar esta questão traz à baila o conceito de **variáveis latentes**.

Seja $\mathbf{x} \in \mathbb{R}^{K\times 1}$ um padrão pertencente ao conjunto de dados. Cada atributo $x_i, i = 1, \ldots K$ corresponde a uma variável observável. Quando aplicamos uma metodologia de redução de dimensionalidade, obtemos uma representação $\mathbf{y} \in \mathbb{R}^{M\times 1}$ cujos atributos podem ser interpretados como variáveis que estavam subjacentes (ou escondidas em meio) aos dados (daí o nome de variáveis latentes).

Com isto, torna-se possível analisar, explicar e processar os dados originais a partir das variáveis latentes descobertas.

É importante ressaltar a diferença entre **extração de atributos** e **seleção de variáveis**: no primeiro caso, os dados originais sofrem algum tipo de transformação (*e.g.*, projeção linear ou não-linear), dando origem a novos atributos em um espaço de dimensão possivelmente diferente; no segundo caso, do conjunto de atributos

originais do dado, somente um subconjunto deles é selecionado para fazer parte da análise.

1.1. Maldição da dimensionalidade

Esta expressão, cunhada por Richard E. Bellman, em seu trabalho sobre programação dinâmica (1957), refere-se a vários fenômenos que surgem quando analisamos dados em espaços de alta dimensão.

O cerne da questão é que quando a dimensionalidade aumenta, o volume do espaço cresce tão rapidamente que os dados disponíveis se tornam esparsos, isto é, a densidade de informação neste espaço é reduzida.

Figura extraída de (GOODFELLOW ET AL., 2016). À medida que a dimensionalidade aumenta, o número de regiões de interesse pode crescer exponencialmente. No caso unidimensional, desejamos distinguir 10 regiões de interesse; com amostras suficientes em cada uma dessas regiões, algoritmos de aprendizado podem generalizar corretamente. Com duas dimensões, é mais difícil distinguir 10 valores diferentes para cada variável: neste caso, devemos cobrir $10 \times 10 = 100$ regiões. Com três dimensões, este número cresce para $10 \times 10 \times 10 = 1000$. Para d dimensões e v valores por variável, precisamos de $O(v^d)$ regiões e exemplos.

Implicações:

- Crescimento exponencial no número de exemplos necessários para manter uma determinada densidade de amostras.

- Crescimento da complexidade da função objetivo envolvida no processo de aprendizado.
- Tratamento estatístico em espaços de dimensão mais elevada: a distribuição Gaussiana admite uma extensão para este caso com uma parametrização relativamente simples; contudo, não há tantas opções quanto no caso unidimensional.

2. Análise de Componentes Principais (PCA)

A análise de componentes principais (PCA, do inglês *principal component analysis*) é uma técnica amplamente utilizada para redução de dimensionalidade, compressão de dados (com perda), extração de atributos e visualização de dados (JOLLIFFE, 2002).

Para sua exposição formal e sem perda de generalidade, vamos considerar que os dados disponíveis $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1{}^T \\ \vdots \\ \mathbf{x}_N{}^T \end{bmatrix} \in \mathbb{R}^{N \times K}$ têm média nula, *i.e.*, $E\{\mathbf{x}_i\} = 0$.

2.1. Formulações de mínimo erro de reconstrução e máxima variância

Existem duas formulações possíveis e equivalentes para a PCA:

- Encontrar a projeção linear dos dados em um espaço de dimensão reduzida $M < K$ que faça com que o erro quadrático médio de reconstrução seja minimizado.
- Encontrar a projeção linear dos dados em um espaço de dimensão reduzida $M < K$ que faça com que as variáveis obtidas, denominadas de componentes principais (ou, simplesmente, *features*) possuam máxima variância (energia).

Em PCA, um novo vetor de atributos $\mathbf{y} \in \mathbb{R}^{M\times 1}$ é obtido por meio da projeção de $\mathbf{x}$ sobre as direções definidas pelas colunas de uma matriz $\mathbf{W} = [\mathbf{w}_1 \ \ldots \ \mathbf{w}_M] \in \mathbb{R}^{K\times M}$, onde $\mathbf{w}_i \in \mathbb{R}^{K\times 1}$:

$$\mathbf{y} = \mathbf{W}^T\mathbf{x}, \tag{1}$$

sendo que $\mathbf{W}^T\mathbf{W} = \mathbf{I}$, ou seja, as colunas de $\mathbf{W}$ formam uma base ortonormal do subespaço de dimensão M.

Reconstrução: $\hat{\mathbf{x}} = \mathbf{W}\mathbf{y} = \mathbf{W}\mathbf{W}^T\mathbf{x}$.

A PCA busca determinar a matriz $\mathbf{W}$ que leva ao menor erro quadrático médio de reconstrução, expresso como:

$$J_{\text{PCA}} = E\{\|\mathbf{x} - \hat{\mathbf{x}}\|^2\} = E\left\{\text{tr}\left((\mathbf{x} - \hat{\mathbf{x}})(\mathbf{x} - \hat{\mathbf{x}})^T\right)\right\} \tag{2}$$

Note que:

$$\mathbf{e} = (\mathbf{x} - \hat{\mathbf{x}}) \;\Rightarrow\; \|\mathbf{e}\|^2 = \mathbf{e}^\mathbf{T}\mathbf{e} = \sum_{i=1}^{K} {e_i}^2 \tag{3}$$

Isto também pode ser calculado via produto externo:

$$\mathbf{e}\mathbf{e}^\mathbf{T} = \begin{bmatrix} e_1 e_1 & \cdots & e_1 e_K \\ \vdots & \ddots & \vdots \\ e_K e_1 & \cdots & e_K e_K \end{bmatrix} \tag{4}$$

A soma dos elementos da diagonal principal é dada por:

$$\text{tr}(\mathbf{e}\mathbf{e}^\mathbf{T}) = \sum_{i=1}^{K} {e_i}^2 \tag{5}$$

Ou seja, $\|\mathbf{e}\|^2 = \mathbf{e}^\mathbf{T}\mathbf{e} = \text{tr}(\mathbf{e}\mathbf{e}^\mathbf{T})$.

Retomando a expressão da função custo de PCA, podemos expandi-la como:

$$J_{\text{PCA}} = E\{\text{tr}(\mathbf{x}\mathbf{x}^T - \mathbf{x}\hat{\mathbf{x}}^T - \hat{\mathbf{x}}\mathbf{x}^T + \hat{\mathbf{x}}\hat{\mathbf{x}}^T)\} \tag{6}$$

Substituindo $\hat{\mathbf{x}} = \mathbf{W}\mathbf{W}^T\mathbf{x}$ em (6), obtemos:

$$J_{\text{PCA}} = E\{\text{tr}(\mathbf{x}\mathbf{x}^T)\} - E\{\text{tr}(\mathbf{x}(\mathbf{W}\mathbf{W}^T\mathbf{x})^T)\} - E\{\text{tr}(\mathbf{W}\mathbf{W}^T\mathbf{x}\mathbf{x}^T)\} + E\{\text{tr}(\mathbf{W}\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W}\mathbf{W}^T)\} \tag{7}$$

Propriedades: dadas duas matrizes $\mathbf{A} \in \mathbb{R}^{K\times M}$ e $\mathbf{B} \in \mathbb{R}^{M\times K}$,

- $\text{tr}(\mathbf{AB}) = \text{tr}(\mathbf{BA})$.
- $\text{tr}(\mathbf{A}) = \text{tr}(\mathbf{A}^T)$.

Vamos combinar estas duas propriedades para simplificar os termos de (7):

$$E\{\text{tr}(\underbrace{\mathbf{x}(\mathbf{W}\mathbf{W}^T\mathbf{x})^T}_{\mathbf{H}})\} = E\{\mathbf{H}\} = E\{\mathbf{H}^T\} = E\{\text{tr}(\mathbf{W}\mathbf{W}^T\mathbf{x}\mathbf{x}^T)\} \tag{8}$$

$$E\{\text{tr}(\underbrace{\mathbf{W}\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W}}_{\mathbf{A}\in\mathbb{R}^{K\times M}} \underbrace{\mathbf{W}^T}_{\mathbf{B}\in\mathbb{R}^{M\times K}})\} = E\{\text{tr}(\mathbf{AB})\} = E\{\text{tr}(\mathbf{BA})\} = E\{\text{tr}(\mathbf{W}^T\mathbf{W}\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W})\} \tag{9}$$

Como $\mathbf{W}^T\mathbf{W} = \mathbf{I}$, então:

$$E\{\text{tr}(\mathbf{W}\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W}\mathbf{W}^T)\} = E\{\text{tr}(\mathbf{W}^T\mathbf{W}\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W})\} = E\{\text{tr}(\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W})\} \tag{10}$$

Com estas alterações, o segundo e o terceiro termo de J_{PCA} são iguais. Prosseguindo, então, com as manipulações:

$$\begin{aligned} J_{\text{PCA}} &= \text{tr}(E\{\mathbf{x}\mathbf{x}^T\}) - 2E\{\text{tr}(\mathbf{x}(\mathbf{W}\mathbf{W}^T\mathbf{x})^T)\} + E\{\text{tr}(\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W})\} \\ &= \text{tr}(E\{\mathbf{x}\mathbf{x}^T\}) - 2E\{\text{tr}(\mathbf{x}\mathbf{x}^T\mathbf{W}\mathbf{W}^T)\} + E\{\text{tr}(\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W})\} \end{aligned} \tag{11}$$

Ora, $E\{\mathbf{x}\mathbf{x}^T\} = \mathbf{R_x}$ é a matriz de autocorrelação dos dados. Então:

$$\begin{aligned} J_{\text{PCA}} &= \text{tr}(\mathbf{R_x}) - 2\text{tr}(E\{\mathbf{x}\mathbf{x}^T\}\mathbf{W}\mathbf{W}^T) + \text{tr}(\mathbf{W}^T E\{\mathbf{x}\mathbf{x}^T\}\mathbf{W}) \\ &= \text{tr}(\mathbf{R_x}) - 2\text{tr}(\mathbf{R_x}\mathbf{W}\mathbf{W}^T) + \text{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W}) \\ &= \text{tr}(\mathbf{R_x}) - 2\text{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W}) + \text{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W}) \end{aligned} \tag{12}$$

Finalmente,

$$J_{\text{PCA}} = \text{tr}(\mathbf{R_x}) - \text{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W}). \tag{13}$$

O objetivo em PCA é minimizar a função custo J_{PCA}. Contudo, como o primeiro termo em (13) não depende da matriz $\mathbf{W}$, basta minimizarmos $-\text{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W})$, o que equivale a maximizar $\text{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W})$.

$$\min J_{\text{PCA}} = \max \text{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W}) \tag{14}$$

Ora, podemos reescrever o termo $\mathrm{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W})$ em (14) da seguinte forma:

$$\mathrm{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W}) = \mathrm{tr}(\mathbf{W}^T E\{\mathbf{x}\mathbf{x}^T\}\mathbf{W}) = E\{\mathrm{tr}(\mathbf{W}^T\mathbf{x}\mathbf{x}^T\mathbf{W})\} \quad (15)$$

Como $\mathbf{W}^T\mathbf{x} = \mathbf{y}$, o termo dentro do operador de esperança corresponde a $\mathbf{y}\mathbf{y}^T$**:**

$$E\{\mathrm{tr}(\mathbf{W}^T\mathbf{R_x}\mathbf{W})\} = E\{\mathrm{tr}(\mathbf{y}\mathbf{y}^T)\} = E\{\mathbf{y}^T\mathbf{y}\} = \sum_{i=1}^{M} E\{{y_i}^2\}. \quad (16)$$

Com isto, percebemos que minimizar o erro de reconstrução é equivalente a buscar as direções em **W** tais que as projeções de **x** sobre elas tenham máxima variância.

2.2. Solução ótima para PCA

Nesta seção, faremos o desenvolvimento da solução ótima de PCA através de um processo de indução matemática.

Começaremos com a primeira componente principal ($M = 1$). O problema em questão pode ser definido da seguinte maneira:

$$\max_{\mathbf{w}_1} E\{{y_1}^2\} = \max_{\mathbf{w}_1} \mathrm{tr}({\mathbf{w}_1}^T \mathbf{R_x} \mathbf{w}_1) \tag{17}$$

$$s.a. \quad \|\mathbf{w}_1\| = 1$$

Utilizando o método dos multiplicadores de Lagrange, temos que a solução para (17) pode ser obtida a partir do seguinte problema sem restrições:

$$\max_{\mathbf{w}_1} \mathcal{L}(\mathbf{w}_1, \lambda_1) = \max_{\mathbf{w}_1} \mathrm{tr}({\mathbf{w}_1}^T \mathbf{R_x} \mathbf{w}_1) - \lambda_1({\mathbf{w}_1}^T \mathbf{w}_1 - 1) \tag{18}$$

Aplicando as condições de otimalidade:

$$\frac{\partial \mathcal{L}(\mathbf{w}_1, \lambda_1)}{\partial \mathbf{w}_1} = \frac{\partial \mathrm{tr}({\mathbf{w}_1}^T \mathbf{R_x} \mathbf{w}_1)}{\partial \mathbf{w}_1} - \frac{\partial \lambda_1({\mathbf{w}_1}^T \mathbf{w}_1 - 1)}{\partial \mathbf{w}_1} = 0 \tag{19}$$

Ora, $\mathrm{tr}({\mathbf{w}_1}^T \mathbf{R_x} \mathbf{w}_1) = {\mathbf{w}_1}^T \mathbf{R_x} \mathbf{w}_1$, pois se trata de um escalar. Portanto,

$$\frac{\partial \mathcal{L}(\mathbf{w}_1, \lambda_1)}{\partial \mathbf{w}_1} = \frac{\partial {\mathbf{w}_1}^T \mathbf{R_x} \mathbf{w}_1}{\partial \mathbf{w}_1} - \frac{\partial \lambda_1({\mathbf{w}_1}^T \mathbf{w}_1 - 1)}{\partial \mathbf{w}_1} = 2\mathbf{R_x}\mathbf{w}_1 - 2\lambda_1 \mathbf{w}_1 = 0 \tag{20}$$

Finalmente, chegamos a:

$$\mathbf{R_x}\mathbf{w}_1 = \lambda_1 \mathbf{w}_1. \tag{21}$$

Reconhecemos que (21) é a expressão matemática que descreve os autovalores e autovetores da matriz de autocorrelação dos dados $\mathbf{R_x}$. Ou seja, a direção $\mathbf{w}_1$ que define a primeira componente principal está associada a um dos autovetores de $\mathbf{R_x}$. Mas qual dos autovetores deve ser escolhido?

Usando (21), podemos reescrever a função custo original em (17) da seguinte forma:

$$\max_{\mathbf{w}_1} \mathrm{tr}(\mathbf{w}_1{}^T \mathbf{R_x} \mathbf{w}_1) = \max_{\mathbf{w}_1} \mathbf{w}_1{}^T \mathbf{R_x} \mathbf{w}_1 = \max_{\mathbf{w}_1} \mathbf{w}_1{}^T \lambda_1 \mathbf{w}_1 = \max_{\mathbf{w}_1} \lambda_1 \underbrace{\mathbf{w}_1{}^T \mathbf{w}_1}_{=1} = \max_{\mathbf{w}_1} \lambda_1 \qquad (22)$$

Como queremos maximizar a variância da projeção, isto é equivalente a maximizar o autovalor de $\mathbf{R_x}$, λ_1.

Portanto, devemos escolher $\mathbf{w}_1$ como o autovetor de $\mathbf{R_x}$ associado ao maior autovalor, de modo a maximizar a variância da projeção.

Como podemos obter as demais componentes principais ($M \geq 2$)?

A segunda componente principal é obtida fazendo a projeção de $\mathbf{x}$ sobre a direção $\mathbf{w}_2$ tal que a variância $E\{y_2{}^2\} = \mathbf{w}_2{}^T \mathbf{R_x} \mathbf{w}_2$ seja maximizada.

Porém, não só devemos impor a restrição de norma unitária a $\mathbf{w}_2$, mas também exigir que $y_2 = \mathbf{w}_2{}^T\mathbf{x}$ seja descorrelacionada (ou ortogonal) à primeira componente principal $y_1 = \mathbf{w}_1{}^T\mathbf{x}$:

$$\begin{aligned} E\{y_2 y_1{}^T\} = 0 \;\Rightarrow\; E\{\mathbf{w}_2{}^T\mathbf{x}\mathbf{x}^T\mathbf{w}_1\} &= \mathbf{w}_2{}^T E\{\mathbf{x}\mathbf{x}^T\}\mathbf{w}_1 = \mathbf{w}_2{}^T\mathbf{R}_\mathbf{x}\mathbf{w}_1 \\ &= \mathbf{w}_2{}^T\lambda_1\mathbf{w}_1 = 0 \\ &\Rightarrow \mathbf{w}_2{}^T\mathbf{w}_1 = 0. \end{aligned} \tag{23}$$

Usaremos, novamente, o método dos multiplicadores de Lagrange:

$$\max_{\mathbf{w}_2} \mathcal{L}(\mathbf{w}_2, \lambda_2, \varphi_2) = \max_{\mathbf{w}_2}(\mathbf{w}_2{}^T\mathbf{R}_\mathbf{x}\mathbf{w}_2) - \lambda_2(\mathbf{w}_2{}^T\mathbf{w}_2 - 1) - \varphi_2\mathbf{w}_2{}^T\mathbf{w}_1 \tag{24}$$

Calculando a derivada parcial com respeito a $\mathbf{w}_2$:

$$\begin{aligned} \frac{\partial\mathcal{L}(\mathbf{w}_2, \lambda_2, \varphi_2)}{\partial\mathbf{w}_2} &= \frac{\partial\mathbf{w}_2{}^T\mathbf{R}_\mathbf{x}\mathbf{w}_2}{\partial\mathbf{w}_2} - \frac{\partial\lambda_2(\mathbf{w}_2{}^T\mathbf{w}_2 - 1)}{\partial\mathbf{w}_2} - \frac{\partial\varphi_2\mathbf{w}_2{}^T\mathbf{w}_1}{\partial\mathbf{w}_2} \\ &= 2\mathbf{R}_\mathbf{x}\mathbf{w}_2 - 2\lambda_2\mathbf{w}_2 - \varphi_2\mathbf{w}_1 = 0 \end{aligned} \tag{25}$$

Se pré-multiplicarmos à esquerda por $\mathbf{w}_1{}^T$, obtemos:

$$2\mathbf{w}_1{}^T\mathbf{R}_\mathbf{x}\mathbf{w}_2 - 2\lambda_2\mathbf{w}_1{}^T\mathbf{w}_2 - \varphi_2\mathbf{w}_1{}^T\mathbf{w}_1 = 0 \tag{26}$$

Pela ortogonalidade imposta, os dois primeiros termos são nulos, de modo que:

$$\varphi_2 \underbrace{\mathbf{w}_1{}^T \mathbf{w}_1}_{=1} = 0 \tag{27}$$

Logo, $\varphi_2 = 0$. Neste caso, ficamos apenas com os dois primeiros termos em (25):

$$2\mathbf{R_x}\mathbf{w}_2 - 2\lambda_2 \mathbf{w}_2 = 0, \tag{28}$$

o que nos leva a:

$$\mathbf{R_x}\mathbf{w}_2 = \lambda_2 \mathbf{w}_2. \tag{29}$$

Ou seja, a direção $\mathbf{w}_2$ que define a segunda componente principal também está associada a um dos autovetores de $\mathbf{R_x}$. Seguindo o mesmo raciocínio do caso anterior, é possível concluir que $\mathbf{w}_2$ corresponde ao autovetor de $\mathbf{R_x}$ associado ao segundo maior autovalor.

Caso geral: a k-ésima componente principal de $\mathbf{x}$ é determinada por $y_k = \mathbf{w}_k{}^T \mathbf{x}$, em que $\mathbf{w}_k$ é o autovetor da matriz de autocorrelação dos dados $\mathbf{R_x}$ associado ao k-ésimo maior autovalor.

A matriz de aucorrelação $\mathbf{R_x}$ pode ser estimada pela média amostral:

$$\mathbf{R_x} \approx \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n{}^T$$

2.3. Número de componentes principais

A escolha do número de componentes principais (M) é feita tendo em vista um **compromisso** (*trade-off*) entre o erro de aproximação e o grau de compressão atingido: enquanto o erro de aproximação (ou reconstrução) é reduzido quanto mais componentes principais são consideradas, a compressão é cada vez maior quanto menos componentes principais são mantidas.

Uma estratégia simples para guiar esta escolha consiste em olhar para a porcentagem da variância total dos dados que conseguimos capturar com M componentes principais. Como

$$E\{y_k{}^2\} = \mathbf{w}_k{}^T \mathbf{R_x} \mathbf{w}_k = \mathbf{w}_k{}^T \lambda_k \mathbf{w}_k = \lambda_k, \qquad (30)$$

então a soma de todos os autovalores de $\mathbf{R_x}$ equivale a $\sum_{k=1}^{K} E\{{y_k}^2\}$.

Sendo assim, a parcela da variância total associada às M componentes principais é dada por:

$$V_M = \frac{\sum_{k=M+1}^{K} E\{{y_k}^2\}}{\sum_{k=1}^{K} E\{{y_k}^2\}} = \frac{\sum_{k=M+1}^{K} \lambda_k}{\sum_{k=1}^{K} \lambda_k}, \tag{31}$$

supondo uma ordenação crescente dos autovalores de $\mathbf{R_x}$.

Exemplo:

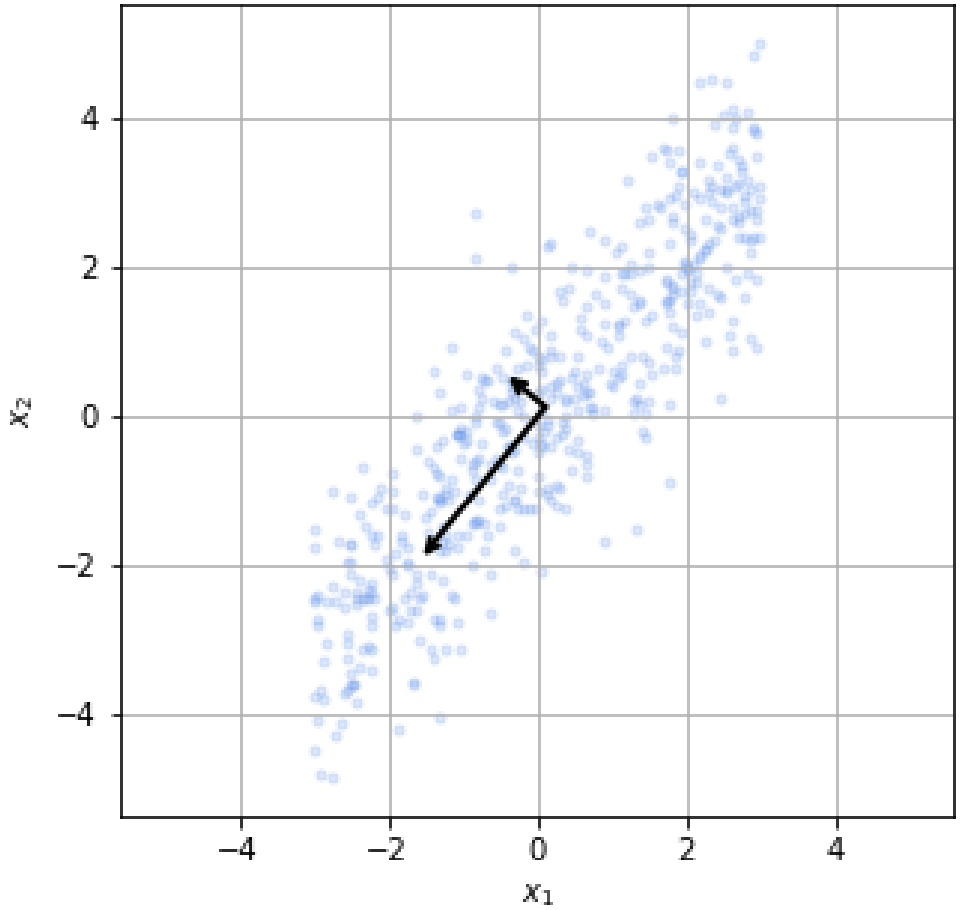
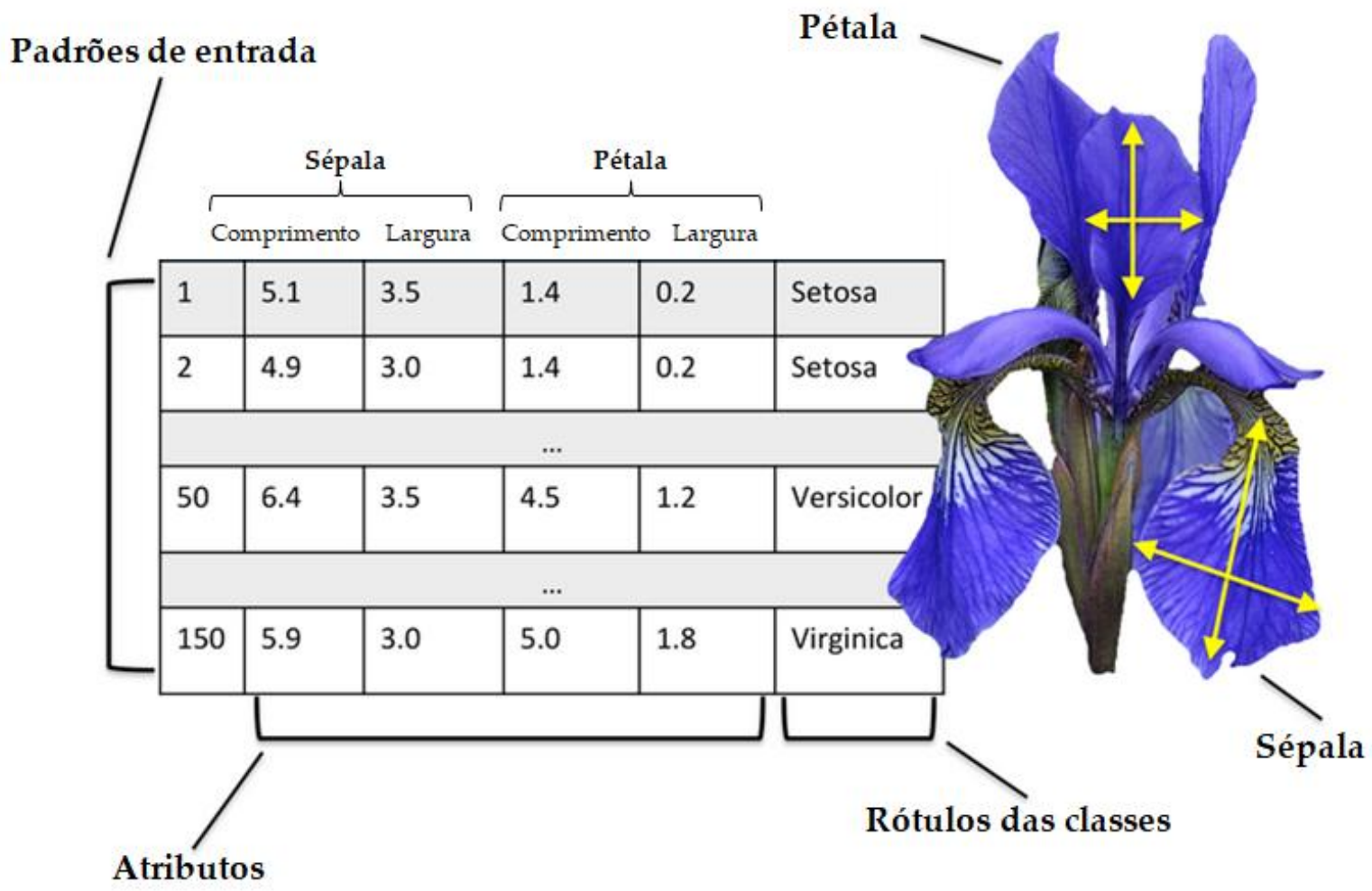


Figura. Distribuição dos dados no espaço original e as duas direções principais identificadas pela PCA. O comprimento dos vetores reflete a diferença das variâncias capturadas por cada direção.

Exemplo: conjunto de dados Iris



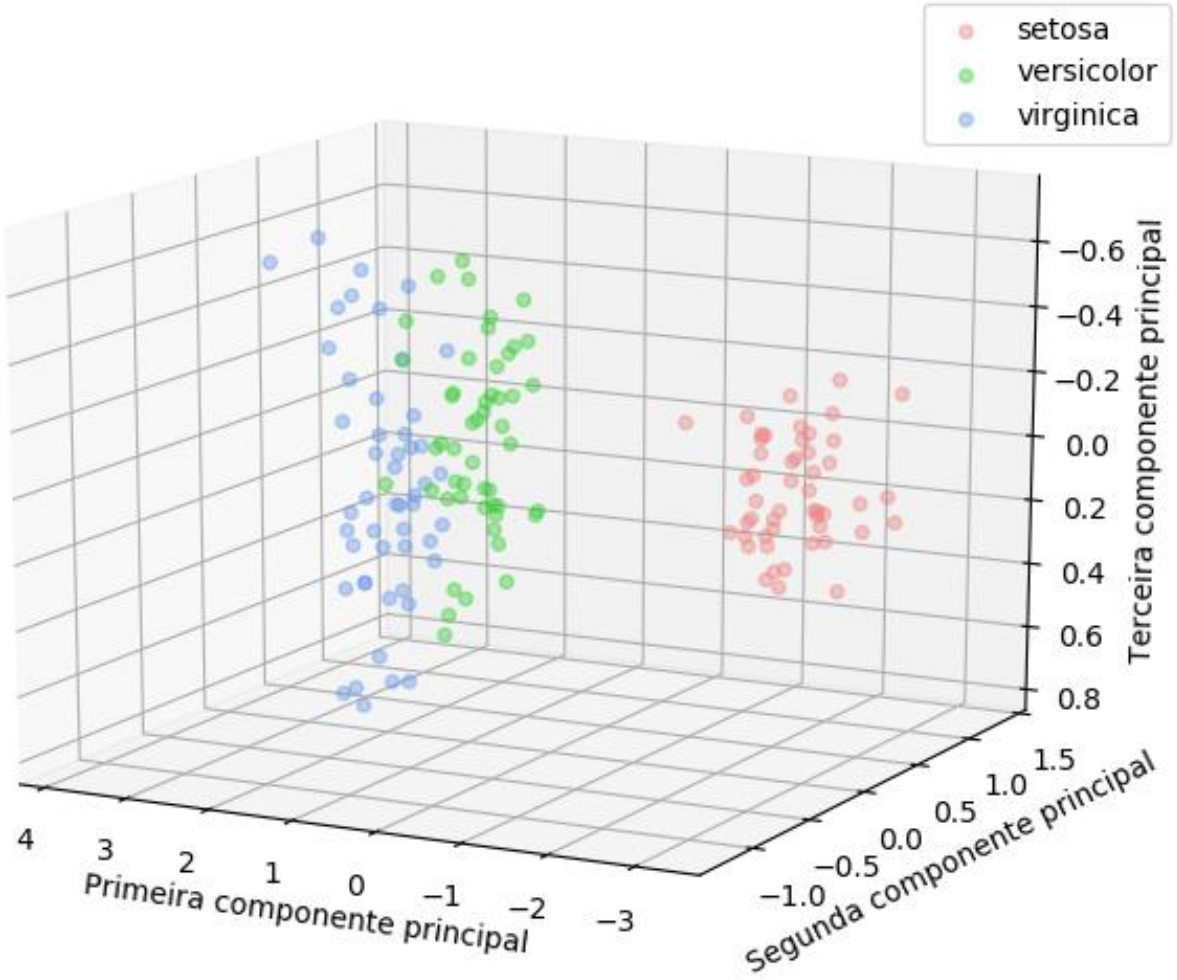


Figura. Visualização das três componentes principais obtidas por PCA para o conjunto Iris. Observe que uma das classes (setosa) ocupa uma região bem definida e isolada do espaço, enquanto as outras (versicolor e virginica) ficam bem próximas.

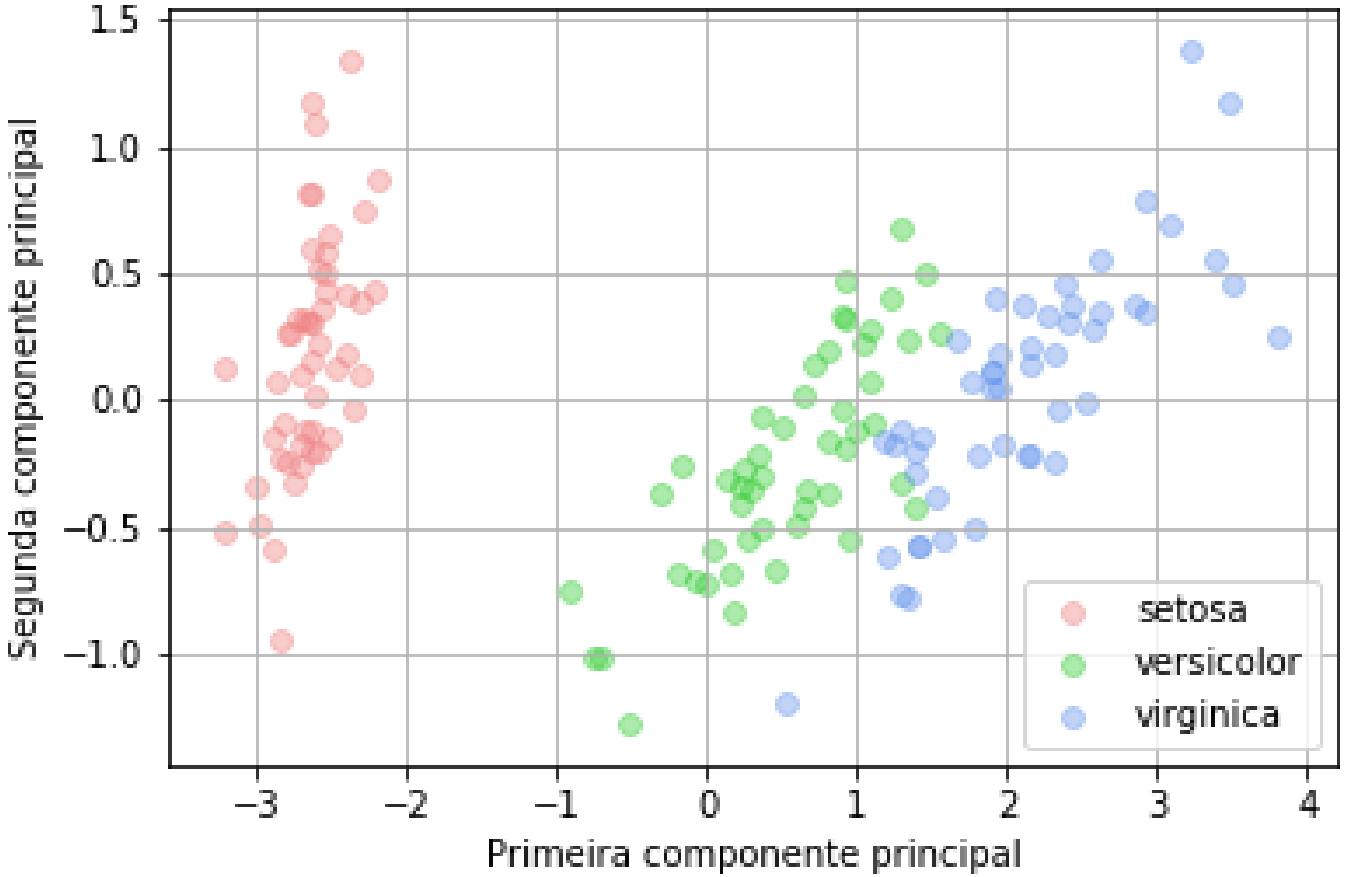


Figura. Visualização das duas componentes principais obtidas por PCA para o conjunto Iris.

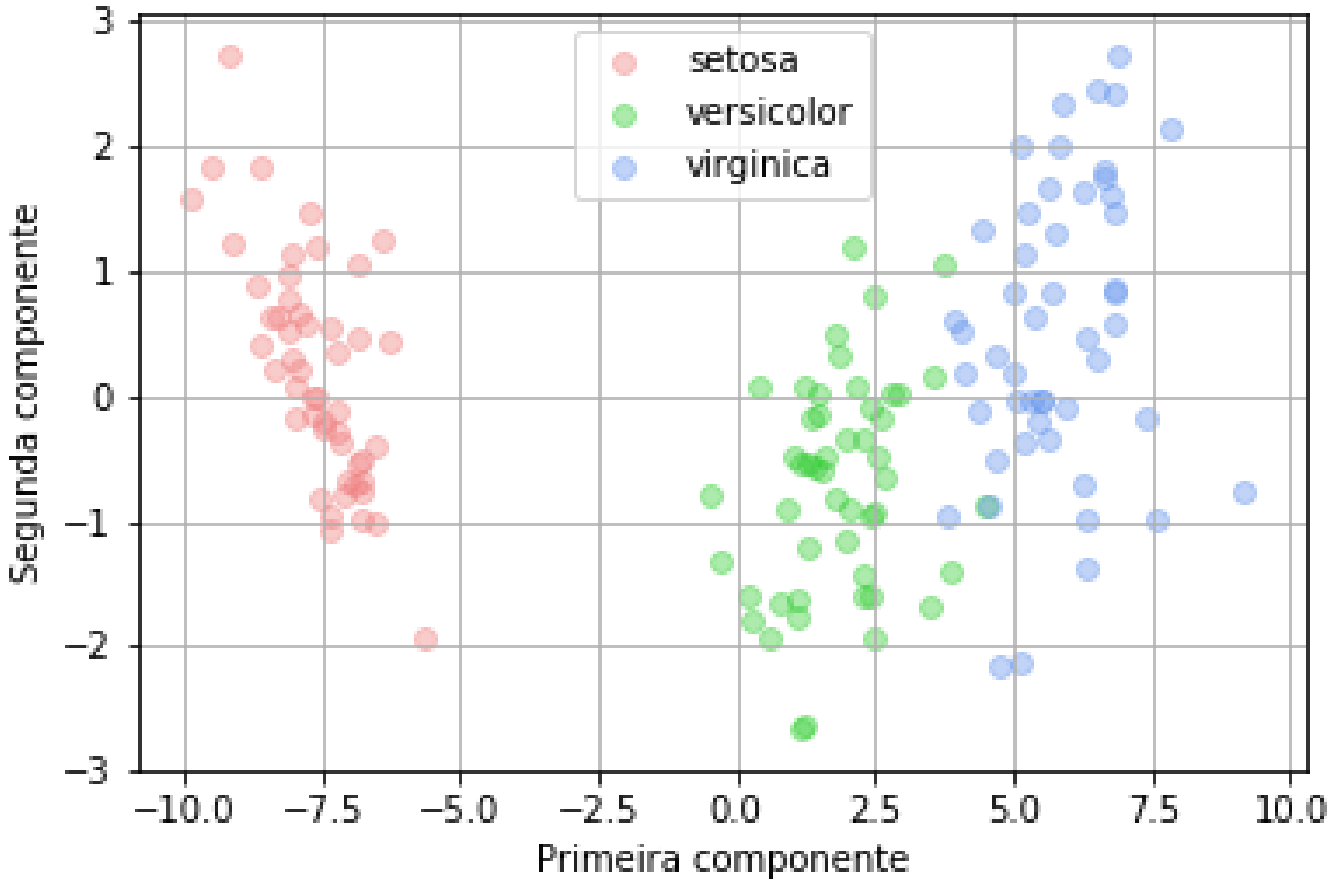


Figura. Visualização dos dados Iris após a projeção sobre as duas primeiras direções definidas pelo discriminante de Fisher. Embora o resultado seja, até certo ponto, semelhante àquele verificado para a PCA, é importante recordar que a LDA é uma técnica que usa a informação dos rótulos de cada padrão, enquanto a PCA é inteiramente não-supervisionada.

3. *Kernel* PCA

Semelhantemente ao que é feito no contexto das máquinas de vetores-suporte (SVMs, do inglês *support vector machines*), a ideia da *kernel* PCA (KPCA) consiste em aplicar uma transformação não-linear sobre os dados originais $\mathbf{x}_i \in \mathbb{R}^K, i = 1, \ldots, N$, levando-os a um espaço de dimensão L, no qual se obtém as direções principais, sobre as quais projetamos os dados transformados. Com isto, no espaço original, o efeito é semelhante a uma projeção dos dados sobre curvas não-lineares (SCHÖLKOPF ET AL., 1998).

Porém, em vez de explicitamente escolher um mapeamento que faça esta transformação $\phi(\mathbf{x}): \mathbb{R}^K \rightarrow \mathbb{R}^L$, a KPCA também explora o truque do *kernel*.

Para isto, é necessário formular o problema exclusivamente em termos de produtos internos entre os dados transformados $\phi(\mathbf{x}_i)$, os quais serão, então, substituídos pelos valores da função *kernel*, conforme mostra a expressão abaixo:

$$\phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m) = \kappa(\mathbf{x}_n, \mathbf{x}_m) \tag{32}$$

Na PCA clássica, as componentes principais são definidas pelos autovetores $\mathbf{u}_i$ da matriz de autocorrelação dos dados $\mathbf{R_x} \in \mathbb{R}^{K \times K}$, conforme a expressão:

$$\mathbf{R_x}\mathbf{u}_i = \lambda_i \mathbf{u}_i, \tag{33}$$

na qual $i = 1, \ldots, K$, $\mathbf{R_x} = E\{\mathbf{x}\mathbf{x}^T\} \approx \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n{}^T$ e os autovetores são ortonormais, *i.e.*,

$$\mathbf{u}_i{}^T \mathbf{u}_j = \begin{cases} 1, \text{para } i = j \\ 0, \text{para } i \neq j \end{cases}. \tag{34}$$

Considere, então, uma transformação não-linear $\phi(\mathbf{x}): \mathbb{R}^K \rightarrow \mathbb{R}^L$ em um espaço de atributos (*feature space*) L-dimensional, de modo que cada ponto $\mathbf{x}_n \in \mathbb{R}^K$ é projetado em um ponto $\phi(\mathbf{x}_n) \in \mathbb{R}^L$. Podemos, então, aplicar a PCA convencional neste novo espaço ($\mathbb{R}^L$), o que equivale a implicitamente definir um modelo de componentes principais não-lineares no domínio original dos dados.

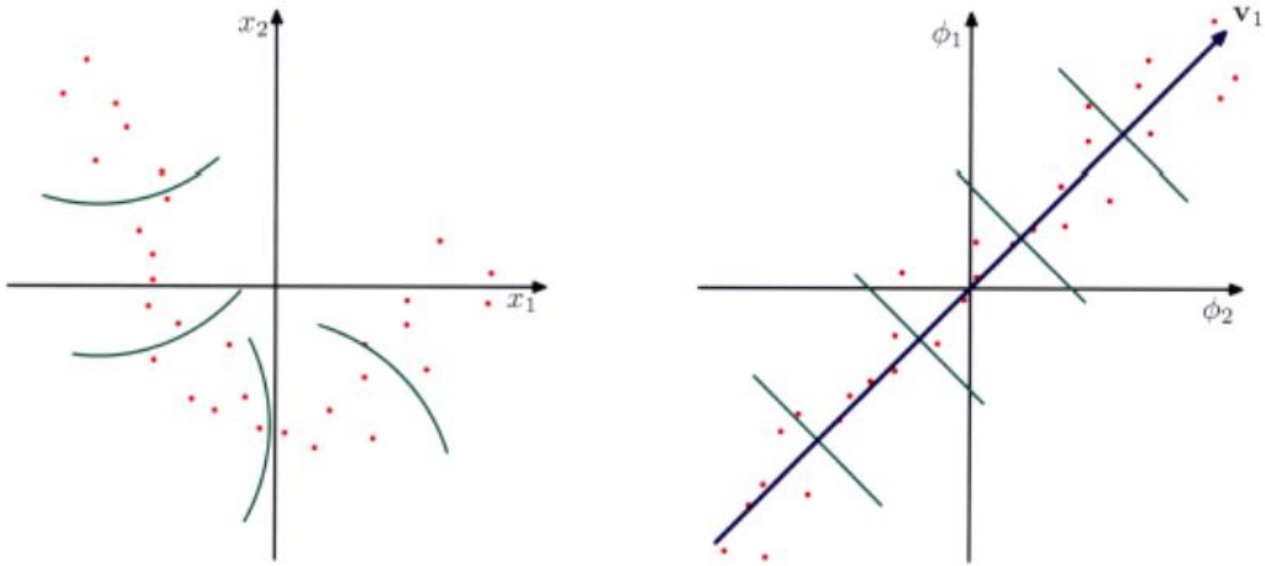


Figura extraída de (BISHOP, 2006). Ilustração da operação de KPCA. Por meio da transformação não-linear $\phi(\cdot)$, os dados são projetados no espaço de *features*, onde a PCA clássica fornece as direções principais. As linhas em verde indicam as projeções lineares sobre a primeira componente principal (vetor $\mathbf{v}_1$), as quais correspondem a projeções não-lineares no espaço original.

Por enquanto, vamos supor que os dados projetados também possuem média nula ($\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n) = 0$). No espaço de atributos, a matriz de autocorrelação dos dados é dada por:

$$\mathbf{C} = \frac{1}{N}\sum_{n=1}^{N} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T \tag{35}$$

e sua expansão em autovetores é definida por:

$$\mathbf{C}\mathbf{v}_i = \lambda_i \mathbf{v}_i, \tag{36}$$

$i = 1, \ldots, L$. Nosso objetivo é resolver este problema sem ter que explicitamente trabalhar no espaço de atributos, isto é, sem ter que definir um mapeamento $\phi(\cdot)$ e realizar, de fato, a projeção não-linear dos dados.

Substituindo a definição da matriz de autocorrelação em (36), as equações dos autovetores nos dizem que $\mathbf{v}_i$ deve satisfazer à seguinte condição:

$$\frac{1}{N}\sum_{n=1}^{N} \phi(\mathbf{x}_n) \underbrace{\{\phi(\mathbf{x}_n)^T \mathbf{v}_i\}}_{\text{escalar}} = \lambda_i \mathbf{v}_i. \tag{37}$$

Vemos, então, que o vetor $\mathbf{v}_i$ é dado por uma combinação linear dos vetores $\phi(\mathbf{x}_n)$ – desde que $\lambda_i > 0$ – de modo que podemos escrevê-lo na forma:

$$\mathbf{v}_i = \sum_{n=1}^{N} a_{in}\phi(\mathbf{x}_n), \tag{38}$$

em que $a_{in} = \phi(\mathbf{x}_n)^T\mathbf{v}_i$.

Substituindo (38) na equação dos autovetores, obtemos:

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_m) = \lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_n). \tag{39}$$

Multiplicando ambos os lados de (39) por $\phi(\mathbf{x}_l)^T$, chegamos a:

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_l)^T\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_m) = \lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_l)^T\phi(\mathbf{x}_n) \tag{40}$$

Agora, em toda a expressão aparecem termos envolvendo produtos internos entre pontos no espaço transformado. Usando o truque do *kernel*, descrito na Equação (32), podemos escrever que:

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_l)^T\phi(\mathbf{x}_n)\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)=\lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_l)^T\phi(\mathbf{x}_n)$$
$$\frac{1}{N}\sum_{n=1}^{N}\kappa(\mathbf{x}_l,\mathbf{x}_m)\sum_{m=1}^{N}a_{im}\kappa(\mathbf{x}_n,\mathbf{x}_m)=\lambda_i\sum_{n=1}^{N}a_{in}\kappa(\mathbf{x}_l,\mathbf{x}_n) \quad (41)$$

Seja $\mathbf{K}=\boldsymbol{\Phi}\boldsymbol{\Phi}^T$ a matriz de Gram, de dimensão $N\times N$, simétrica, cujos elementos são dados por:

$$[\mathbf{K}]_{nm}=\phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)=\kappa(\mathbf{x}_n,\mathbf{x}_m). \quad (42)$$

Então, podemos adotar uma notação matricial para (41), obtendo:

$$\mathbf{K}^2\mathbf{a}_i=\lambda_i N\mathbf{K}\mathbf{a}_i, \quad (43)$$

na qual $\mathbf{a}_i$ é um vetor coluna N-dimensional com elementos $a_{in}, n=1,\ldots,N$.

É possível mostrar que as soluções úteis de (43), as quais estão relacionadas aos autovetores referentes a autovalores não-nulos de $\mathbf{K}$, são as mesmas que aquelas obtidas na seguinte igualdade (BISHOP, 2006):

$$\mathbf{K}\mathbf{a}_i = \lambda_i N \mathbf{a}_i. \tag{44}$$

A condição de normalização para os vetores de coeficientes $\mathbf{a}_i$ é obtida a partir da exigência de que os autovetores no espaço de atributos sejam normalizados. Usando (38) e (44), temos que:

$$1 = \mathbf{v}_i{}^T \mathbf{v}_i = \sum_{n=1}^{N} \sum_{m=1}^{N} a_{in} a_{im}\, \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m) = \mathbf{a}_i{}^T \mathbf{K} \mathbf{a}_i = \lambda_i N \mathbf{a}_i{}^T \mathbf{a}_i \tag{45}$$

Uma vez resolvido o problema de cálculo dos autovetores, as projeções nas componentes principais também podem ser expressas em termos da função *kernel*. Com efeito, a projeção de um ponto $\mathbf{x}$ sobre o autovetor i é dada por:

$$y_i(x) = \phi(\mathbf{x})^T \mathbf{v}_i = \sum_{n=1}^{N} a_{in} \phi(\mathbf{x})^T \phi(\mathbf{x}_n) = \sum_{n=1}^{N} a_{in} \kappa(\mathbf{x}, \mathbf{x}_n). \tag{46}$$

Observação:

- No domínio original dos dados ($\mathbb{R}^K$), existem K autovetores ortogonais e podemos encontrar, no máximo, K componentes principais lineares.
- A dimensionalidade L do espaço de atributos pode ser muito maior que K, de forma que podemos encontrar um número de componentes principais não-lineares superior a K. Contudo, o número de autovalores não-nulos não pode exceder N (o número de dados disponíveis), uma vez que a matriz de autocorrelação dos dados transformados $\mathbf{C}$ possui, no máximo, posto igual a N. Isto é refletido no fato de que KPCA envolve a expansão em autovetores da matriz de Gram $\mathbf{K} \in \mathbb{R}^{N \times N}$.

No início da derivação, fizemos a suposição de que os dados projetados tinham média nula, o que, em geral, não é verdade. Por isso, precisamos ajustar o método para considerar dados projetados centralizados, denotados por $\tilde{\phi}(\mathbf{x}_n)$:

$$\tilde{\phi}(\mathbf{x}_n) = \phi(\mathbf{x}_n) - \frac{1}{N}\sum_{l=1}^{N} \phi(\mathbf{x}_l) \tag{47}$$

Neste caso, os elementos da matriz de Gram são dados por:

$$\left[\tilde{\mathbf{K}}\right]_{nm} = \tilde{\phi}(\mathbf{x}_n)^T \tilde{\phi}(\mathbf{x}_m) \tag{48}$$

Substituindo (47) em (48), obtemos:

$$\left[\tilde{\mathbf{K}}\right]_{nm} = \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m) - \frac{1}{N}\sum_{l=1}^{N} \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_l) - \frac{1}{N}\sum_{l=1}^{N} \phi(\mathbf{x}_l)^T \phi(\mathbf{x}_m) + \frac{1}{N^2}\sum_{j=1}^{N}\sum_{l=1}^{N} \phi(\mathbf{x}_j)^T \phi(\mathbf{x}_l) \tag{49}$$

Então:

$$\left[\tilde{\mathbf{K}}\right]_{nm} = \kappa(\mathbf{x}_n, \mathbf{x}_m) - \frac{1}{N}\sum_{l=1}^{N} \kappa(\mathbf{x}_l, \mathbf{x}_m) - \frac{1}{N}\sum_{l=1}^{N} \kappa(\mathbf{x}_n, \mathbf{x}_l) + \frac{1}{N^2}\sum_{j=1}^{N}\sum_{l=1}^{N} \kappa(\mathbf{x}_j, \mathbf{x}_l) \tag{50}$$

Explorando novamente uma notação matricial, podemos escrever que:

$$\tilde{\mathbf{K}} = \mathbf{K} - \mathbf{1}_N\mathbf{K} - \mathbf{K}\mathbf{1}_N + \mathbf{1}_N\mathbf{K}\mathbf{1}_N, \tag{51}$$

onde $\mathbf{1}_N$ denota uma matriz $N \times N$ cujos elementos são iguais a $1/N$.

Assim, podemos avaliar $\tilde{\mathbf{K}}$ usando apenas a função *kernel* e, então, utilizar $\tilde{\mathbf{K}}$ para determinar os autovalores e autovetores, conforme a expressão em (44).

Diferença: na PCA clássica, geralmente retemos um número reduzido $L < K$ de autovetores da matriz de autocorrelação e, então, aproximamos um vetor de dados por sua projeção no subespaço L-dimensional, definida como:

$$\hat{\mathbf{x}}_n = \sum_{i=1}^{L} (\mathbf{x}_n{}^T \mathbf{u}_i)\mathbf{u}_i.$$

Na KPCA, isto geralmente não é possível. O vetor $\mathbf{x}_n$ (original) é denominado a pré-imagem do ponto correspondente $\phi(\mathbf{x}_n)$ no espaço de atributos. Entretanto, a projeção dos pontos no espaço de atributos sobre o subespaço de PCA linear em $\mathbb{R}^L$ tipicamente não cairá em um *manifold* K-dimensional não-linear, de modo que não haverá uma pré-imagem correspondente no espaço dos dados. Algumas técnicas foram propostas na literatura para determinar pré-imagens aproximadas (BAKIR et al., 2004), de modo que se possa observar em $\mathbb{R}^K$ o vetor aproximado $\hat{\mathbf{x}}_n$.

Exemplo:

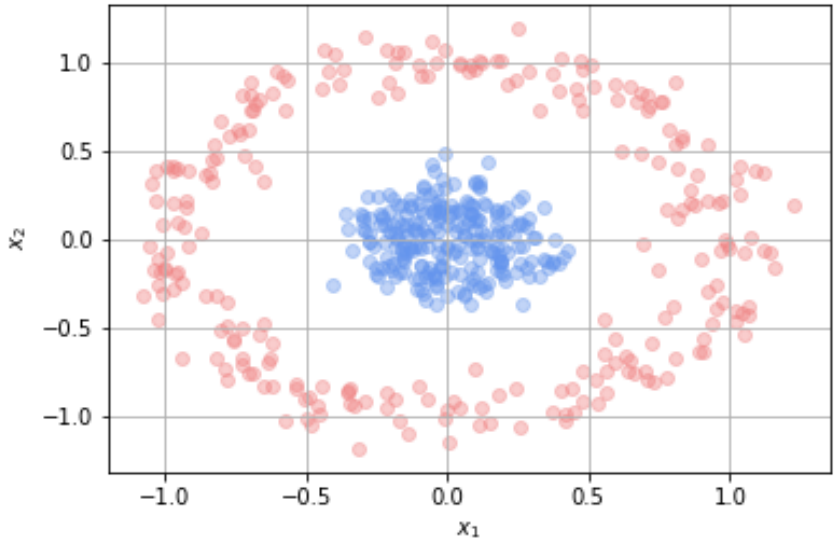
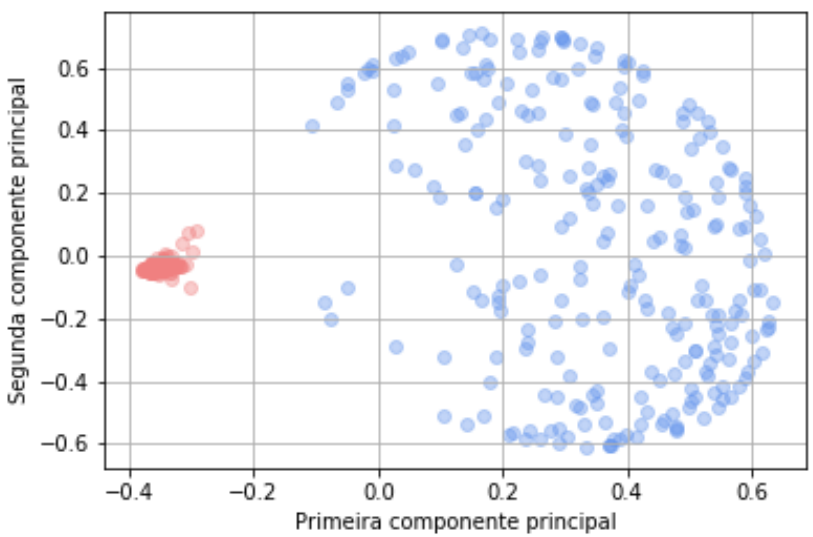
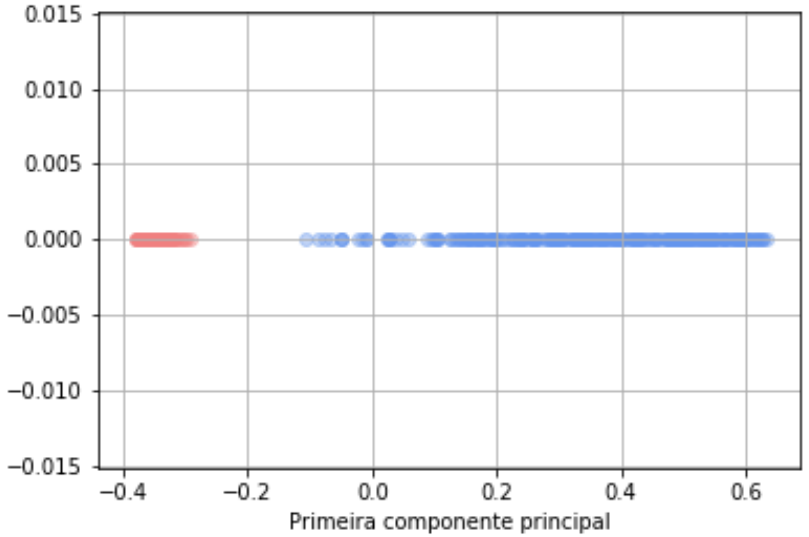


Figura. Distribuição dos dados no espaço original ($\mathbb{R}^2$). As cores discriminam as classes existentes.



(a)



(b)

Figura. Visualização das duas primeiras componentes principais obtidas pela KPCA para o conjunto de dados formado por dois círculos concêntricos. Em ambos os casos, as duas classes ficam bem separadas.

4. Análise de Componentes Independentes (ICA)

Assim como a PCA, a análise de componentes independentes (ICA, do inglês *independent component analysis*) também é uma técnica para identificação de variáveis latentes. Entretanto, a hipótese feita em ICA é a de que as variáveis latentes são estatisticamente independentes, uma condição mais forte do que a ortogonalidade (ou descorrelação) considerada em PCA.

Independência: sejam θ e ξ duas variáveis aleatórias. Então, θ e ξ são independentes se, e somente se,

$$p_{\theta,\xi}(\theta;\xi) = p_\theta(\theta)p_\xi(\xi),$$

em que $p_{\theta,\xi}(\theta;\xi)$ denota a função densidade de probabilidade (PDF, do inglês *probability density function*) conjunta das variáveis θ e ξ, $p_\theta(\theta)$ é a PDF marginal de θ e $p_\xi(\xi)$ é a PDF marginal de ξ. Ou seja, os eventos aleatórios relacionados a θ e ξ não

exercem influência mútua, de maneira que observar um resultado específico para uma das variáveis não interfere no que acontece com a outra variável.

Implicitamente, ICA assume a seguinte relação entre as variáveis observadas $x_i(n), i = 1, \dots, K$ e as variáveis latentes $z_i(n), i = 1, \dots, L$:

$$x_i(n) = a_{i,1}z_1(n) + a_{i,2}z_2(n) + \cdots + a_{i,L}z_L(n),$$

onde $a_{i,j}$ denota o peso que $z_j(n)$ tem na composição de $x_i(n)$. Ou seja, considera-se que as observações resultam de misturas lineares e instantâneas das variáveis latentes. Explorando uma notação matricial, podemos escrever que:

$$\mathbf{x}(n) = \mathbf{A}\mathbf{z}(n),$$

em que $\mathbf{x}(n) = [x_1(n) \ \dots \ x_K(n)]^T$, $\mathbf{z}(n) = [z_1(n) \ \dots \ z_L(n)]^T$ e $\mathbf{A} \in \mathbb{R}^{K \times L}$ é a matriz de mistura.

O desafio em ICA consiste em estimar o vetor de variáveis latentes $\mathbf{z}$ a partir de $\mathbf{x}$ sem conhecer os coeficientes das misturas (*i.e.*, os elementos da matriz $\mathbf{A}$), tendo um conjunto mínimo de informações sobre $\mathbf{z}$. Tal desafio está intimamente relacionado a

uma instância do problema de separação cega de fontes (BSS, do inglês *blind source separation*) (HYVÄRINEN ET AL., 2001; ROMANO ET AL., 2012).

4.1. Definição

A análise de componentes independentes (ICA) de um vetor $\mathbf{x}(n) = [x_1(n) \quad \cdots \quad x_K(n)]^T$ consiste em determinar uma matriz de separação $\mathbf{W}$ de forma que os elementos do vetor

$$\mathbf{y} = \mathbf{W}^T\mathbf{x}$$

sejam tão estatisticamente independentes quanto possível. Para isto, é necessário realizar a otimização de um funcional $J(\mathbf{W})$ que expresse, direta ou indiretamente, uma noção de independência entre os sinais em $\mathbf{y}$ (HYVÄRINEN ET AL., 2001; ROMANO ET AL., 2012).

4.2. Possíveis abordagens para ICA

A formalização de uma técnica de ICA passa pela escolha de um critério matemático, o qual dá origem a uma função custo, que, ao ser otimizada, deve levar à obtenção de sinais com máximo grau de independência estatística entre si. Na literatura, existem diversas opções de critérios e, consequentemente, de algoritmos para ICA.

4.2.1. Informação mútua

Uma possibilidade está baseada no conceito de informação mútua, denotada aqui por $I(\theta, \xi)$. Esta medida, derivada da teoria da informação, revela o grau de informação que uma variável aleatória θ fornece sobre a variável ξ, e vice-versa. Mais especificamente, a informação mútua indica a redução no nível de incerteza associado a uma variável graças à informação trazida pela outra variável. O menor valor possível para a informação mútua é justamente zero, que acontece somente quando as variáveis θ e ξ consideradas forem independentes. Desta maneira, um

possível critério para ICA consiste em minimizar a informação mútua entre os elementos do vetor **y** contendo as estimativas das variáveis latentes.

4.2.2. Estatísticas de ordem superior

Outra maneira de medir independência se dá através dos cumulantes conjuntos. Para sinais independentes (e de média nula), podemos afirmar que os cumulantes conjuntos de qualquer ordem serão iguais a zero. O algoritmo JADE (*Joint Approximate Diagonalization of Eigenmatrices*) (CARDOSO, 1999; HYVARINEN ET AL., 2001), o qual é baseado em tensores de cumulantes de quarta ordem, busca minimizar os cumulantes conjuntos das estimativas, formulando o problema como uma tarefa de diagonalização de matrizes.

4.2.3. Não-Gaussianidade

Uma abordagem um pouco diferente, mas de notória elegância conceitual, está baseada na constatação de que misturas de sinais independentes tendem a

apresentar distribuições de probabilidade mais próximas a uma distribuição gaussiana quando comparadas às variáveis originais, tendo como base o teorema do limite central. Neste sentido, como o vetor de estimativas das variáveis latentes $\mathbf{y}$ pode ser visto como o resultado de uma combinação linear de variáveis independentes, pois

$$\mathbf{y}(n) = \mathbf{W}^T\mathbf{x} = \mathbf{W}^T\mathbf{A}\mathbf{z}(n),$$

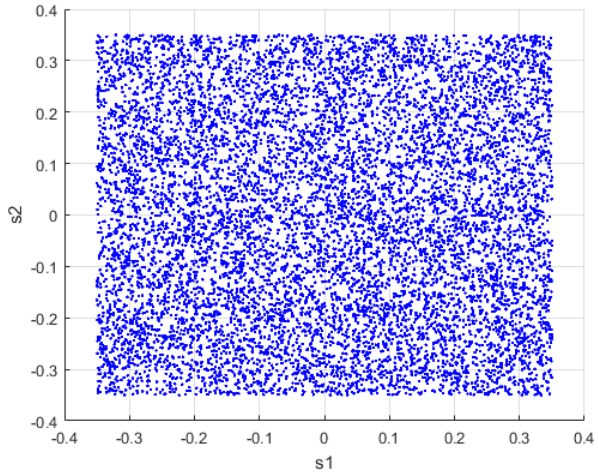
então, maximizar a não-gaussianidade dos elementos de $\mathbf{y}$ implicitamente favorece a obtenção de sinais cada vez menos "misturados". Em outras palavras, quanto mais conseguirmos separar individualmente as variáveis latentes por meio da matriz $\mathbf{W}$, mais afastada de uma distribuição gaussiana será a distribuição observada para cada elemento em $\mathbf{y}$.

Os algoritmos baseados em não-gaussianidade dependem, portanto, de uma medida que acabe por expressar o grau de similaridade entre uma distribuição de

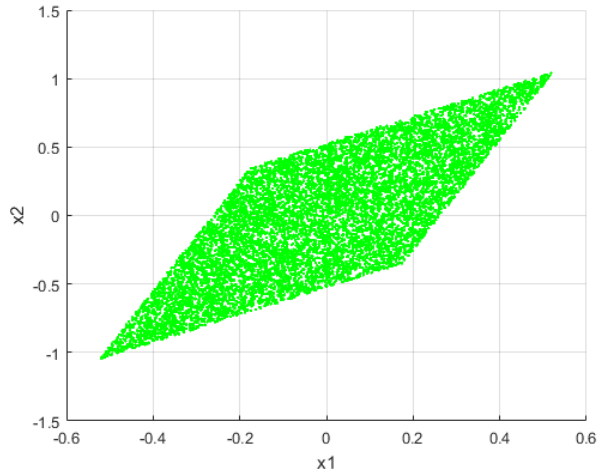
probabilidades qualquer e uma gaussiana. Tipicamente, duas funções custo são utilizadas para este propósito: (*i*) curtose e (*ii*) negentropia.

Além dos critérios já mencionados, existem outras estratégias propostas para ICA, como, por exemplo, as abordagens baseadas em máxima verossimilhança e na noção de descorrelação não-linear.

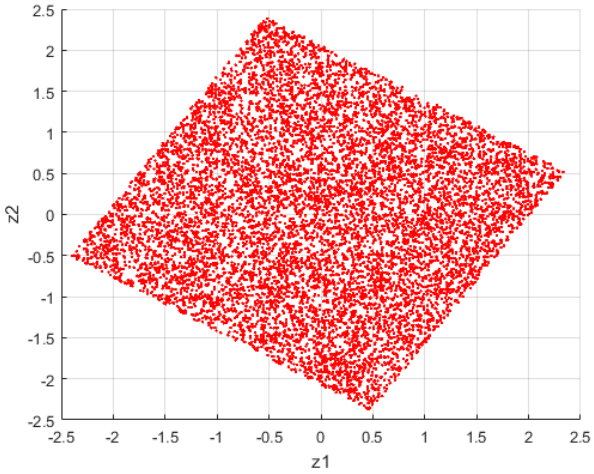
Exemplo: separação de duas fontes uniformes a partir de misturas lineares e instantâneas via ICA.



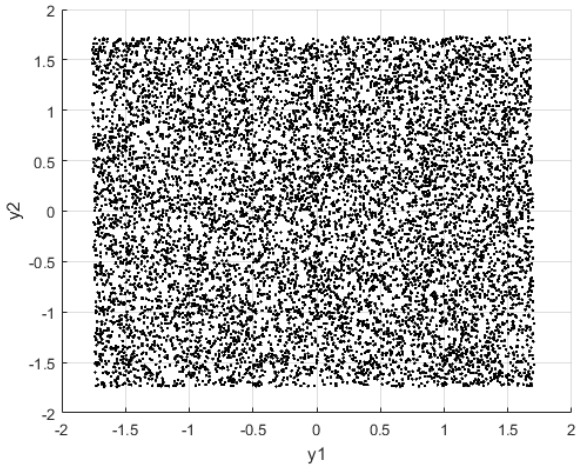
(a)



(b)



(c)



(d)

Figura 2: (a) Distribuição conjunta das variáveis latentes (uniforme); (b) distribuição conjunta das variáveis observadas; (c) distribuição conjunta dos dados após PCA; (d) distribuição conjunta dos sinais após ICA.

5. *Autoencoders*

As redes neurais autocodificadoras (AE, do inglês *autoencoders*), também chamadas de redes autoassociativas, são estruturas construídas com a finalidade de aprender a reproduzir em sua saída $\mathbf{y}$ o próprio dado de entrada $\mathbf{x}$ (BOURLARD & KAMP, 1988; HINTON & ZEMEL, 1994).

A estrutura básica de um AE consiste de duas partes (GOODFELLOW ET AL., 2016):

- Uma função de codificação $\mathbf{h} = f(\mathbf{x})$, que gera uma representação interna para o dado de entrada.
- Uma função de decodificação $\mathbf{y} = g(\mathbf{h})$, que faz o mapeamento da representação interna para a saída na tentativa de reconstruir a entrada.

Evidentemente, não se deseja obter um modelo que apenas desempenhe o papel da função identidade, ou seja, que simplesmente propague a entrada inalterada até a saída.

Por isso, são introduzidos alguns mecanismos no processo de treinamento e/ou na própria estrutura de tal modo a evitar esta solução trivial, forçando o modelo a gerar uma versão aproximada da entrada.

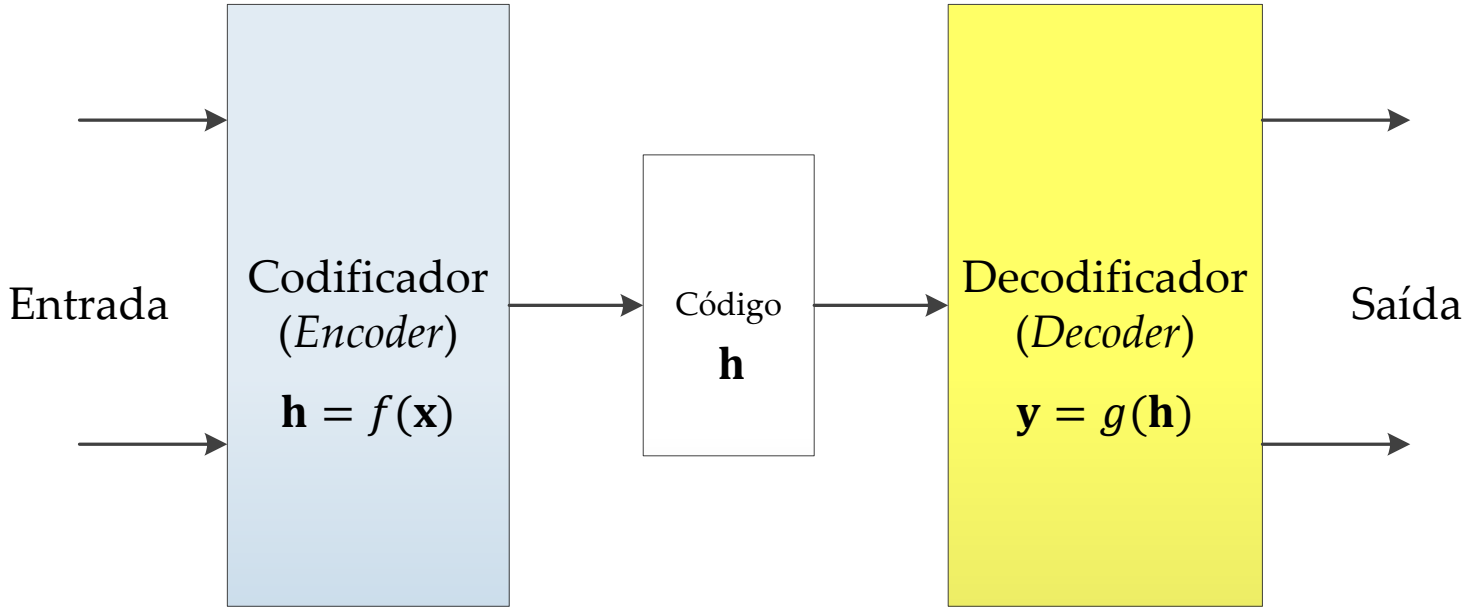


Figura. Estrutura geral de um *autoencoder*.

O processo de treinamento de um AE visa minimizar uma função custo $J(\mathbf{x}; g(f(\mathbf{x})))$ que penaliza o modelo conforme o grau de dissimilaridade entre $g(f(\mathbf{x}))$ e $\mathbf{x}$.

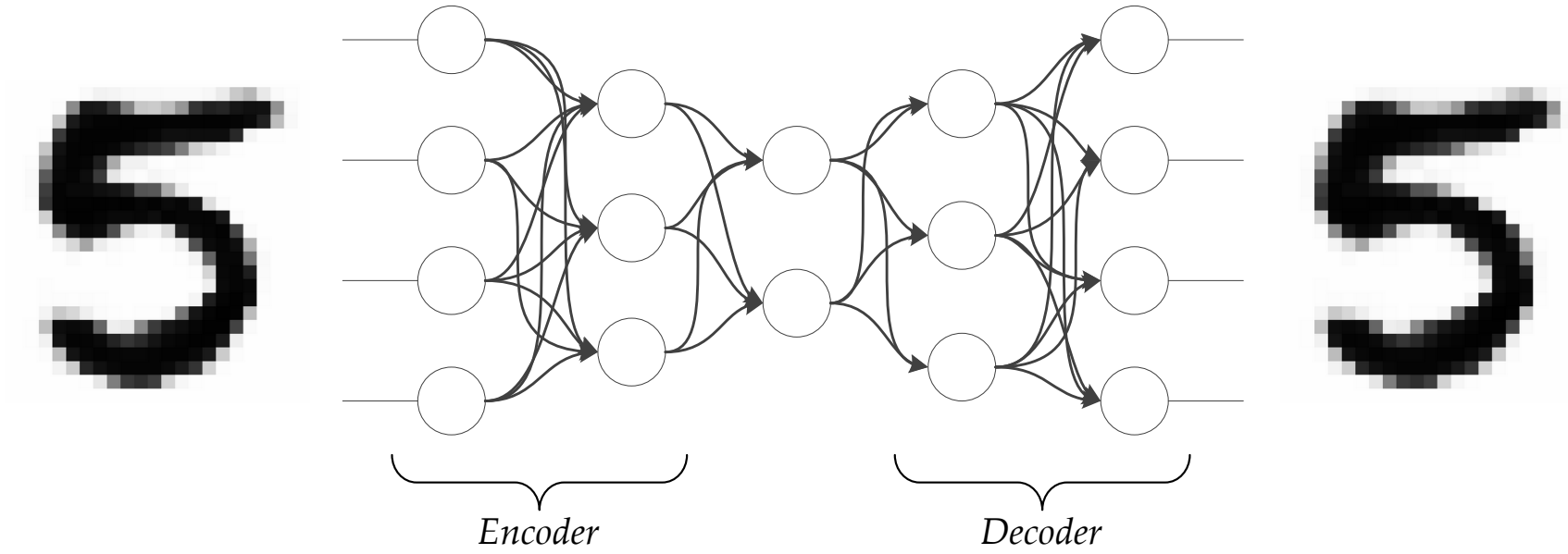


Figura. Exemplo de uma estrutura típica de rede neural multicamadas operando como um *autoencoder*.

Com isto, espera-se que o *autoencoder* aprenda a criar uma representação discriminativa dos dados no código gerado, a partir do qual seja possível reconstruí-los com precisão suficiente. Sendo assim, um possível uso de AEs é para realizar a extração de atributos e/ou a redução de dimensionalidade do dado de entrada.

5.1. *Regularized* AE

Na definição da função custo, são acrescentados termos que encorajam o modelo a exibir outras propriedades além da habilidade de copiar a entrada para a saída.

Exemplo: *Sparse autoencoders*.

- Além de minimizar o erro de reconstrução, deseja-se obter esparsidade na representação interna; neste caso, o critério a ser minimizado é dado na forma:

$$J(\mathbf{x}; g(f(\mathbf{x}))) + \Omega(\mathbf{h}),$$

onde $\Omega(\mathbf{h})$ representa a penalidade relacionada à esparsidade.

5.2. *Denoising* AE

Um *denoising autoencoder* (DAE) é treinado para reconstruir uma versão pura, livre de distorções, da entrada a partir de versões corrompidas deste sinal. O modelo, portanto, deve desfazer as distorções introduzidas pelo ruído em vez de simplesmente copiar a entrada (VINCENT ET AL., 2010).

O objetivo que se pretende atingir com um DAE não é propriamente a remoção do ruído em si. Na verdade, a ideia de *denoising* é interpretada como uma abordagem de treinamento que busca auxiliar o modelo a aprender a extrair atributos úteis.

Implicitamente, espera-se que ao tentar remover o ruído, o modelo seja capaz de descobrir uma representação interna cujos atributos capturem uma estrutura útil da distribuição dos dados de entrada.

5.3. *Contractive* AE

Outra estratégia para regularizar um *autoencoder*, conhecida como *contractive autoencoder* (CAE), consiste em inserir uma penalização na forma:

$$\Omega(\mathbf{h}; \mathbf{x}) = \lambda \sum \|\nabla_x h_i\|^2.$$

Isto força o modelo a aprender uma função (ou um código interno) que não varie abruptamente quando a entrada **x** sofre modificações suaves.

5.4. *Variational* e *Adversarial* AEs

Existem também outras propostas de *autoencoders* que se aproximam de modelos generativos, como o *variational* AE (DOERSCH, 2016) e o *adversarial* AE (MAKHZANI ET AL., 2016).

No processo de aprendizado, estas abordagens tentam aproximar distribuições complicadas que explicam os dados observados, fazendo um mapeamento da entrada para um modelo probabilístico interno (variáveis latentes), a partir do qual é possível gerar novos padrões de saída semelhantes aos dados de treinamento.

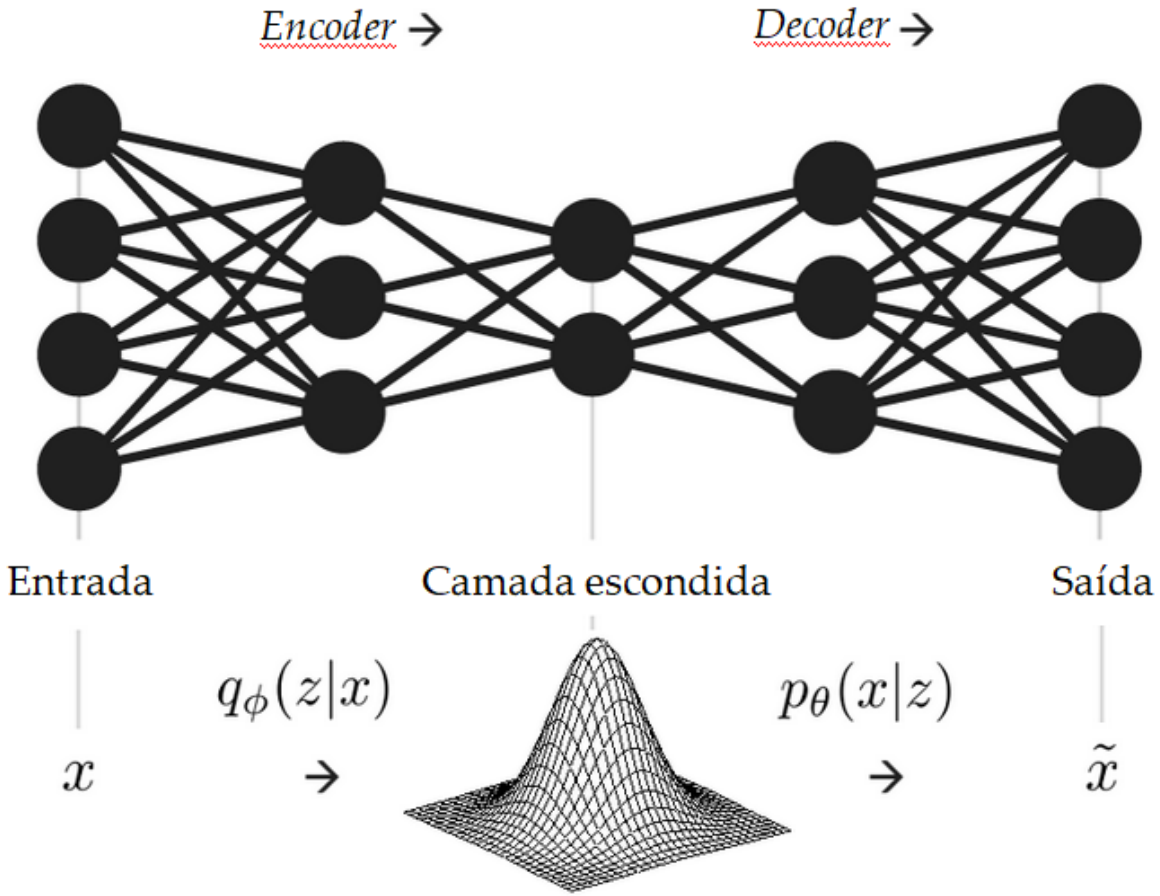


Figura. Ilustração da estrutura e operação de um *variational autoencoder*. Na perspectiva Bayesiana, o codificador se torna uma rede de inferência variacional, mapeando as entradas observadas em distribuições de probabilidade *a posteriori* no espaço latente, enquanto o decodificador corresponde a uma rede generativa, capaz de mapear as coordenadas latentes de volta a distribuições no espaço original dos dados.

6. Referências bibliográficas

ALPAYDIN, E. **Introduction to Machine Learning**. MIT Press. 3rd edition. 2014.

BAKIR, G. H.; WESTON, J.; SCHÖLKOPF, B. Learning to Find Pre-Images. Em S. Thrun, L. K. Saul e B. Schölkopf (Eds.), *Advances in Neural Information Processing Systems*, vol. 16, pp. 449-456, MIT Press, 2004.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. Springer, 2006.

BOURLARD, H.; KAMP, Y. Auto-Association by Multilayer Perceptrons and Singular Value Decomposition. *Biological Cybernetics*, vol. 59, pp. 291-294, 1988.

CARDOSO, J. High-Order Contrasts for Independent Component Analysis. *Neural Computation*, vol. 11, no. 1, pp.157-192, 1999.

DOERSCH, C. Tutorial on Variational Autoencoders. *arXiv*:1606.05908v2 [stat.ML], 2016.

DUDA, R. O., HART, P. E., STORK, D. G. **Pattern Classification**. John Wiley & Sons. 2nd edition, 2001.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016.

HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference and Prediction**. Springer. 2nd edition, 2006.

HAYKIN, S. **Neural Networks and Learning Machines**. Prentice Hall, 3rd edition, 2008.

HINTON, G. E.; ZEMEL, R. S. Autoencoders, Minimum Description Length, and Helmholtz Free Energy. *In Proc. of the Neural Information Processing Systems (NIPS'1993)*, 1994.

HYVÄRINEN, A.; KARHUNEN, J.; OJA, E. **Independent Component Analysis**. Wiley Interscience, 2001.

JOLLIFFE, T. **Principal Component Analysis**, Springer, 2nd edition, 2002.

LUENBERGER, D.G. **Linear and Nonlinear Programming**. 2nd edition, Addison-Wesley Publishing Company, 1984.

MAKHZANI, A.; SHLENS, J.; JAITLY, N.; GOODFELLOW, I.; FREY, B. Adversarial Autoencoders, *arXiv*:1511.05644v2 [cs.LG], 2016.

ROMANO, J. M. T.; ATTUX, R.; CAVALCANTE, C. C.; SUYAMA, R. **Unsupervised Signal Processing: Channel Equalization and Source Separation**. CRC Press, 2012.

SCHÖLKOPF, B.; SMOLA, A.; MÜLLER, K.-P. Nonlinear Component Analysis as a Kernel Eigenvalue Problem, *Neural Computation*, vol. 10, no. 5, pp. 1299-1319, 1998.

VINCENT, P.; LAROCHELLE, H.; LAJOIE, I.; BENGIO, Y.; MANZAGOL, P.-A. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion, *Journal of Machine Learning Research*, vol. 11, pp. 3371-3408, 2010.