Preditiva.ai

Aprendizado Não Supervisionado

Redução de Dimensionalidade: PCA, Autoencoder e t-SNE

Aprendizado Não Supervisionado

Qual a diferença?

Preditiva.ai

A principal diferença entre **modelos supervisionados** e **não supervisionados** é que os não supervisionados **não possuem uma variável target**.

Ou seja, a identificação de padrões se restringe as variáveis explicativas ou *features*. Essa ausência do target sem dúvida torna o processo de identificação dos padrões um pouco **mais desafiador**, mas veremos que existem diversas **técnicas desenvolvidas** especificamente para essa **abordagem**.

Entre essas técnicas temos:

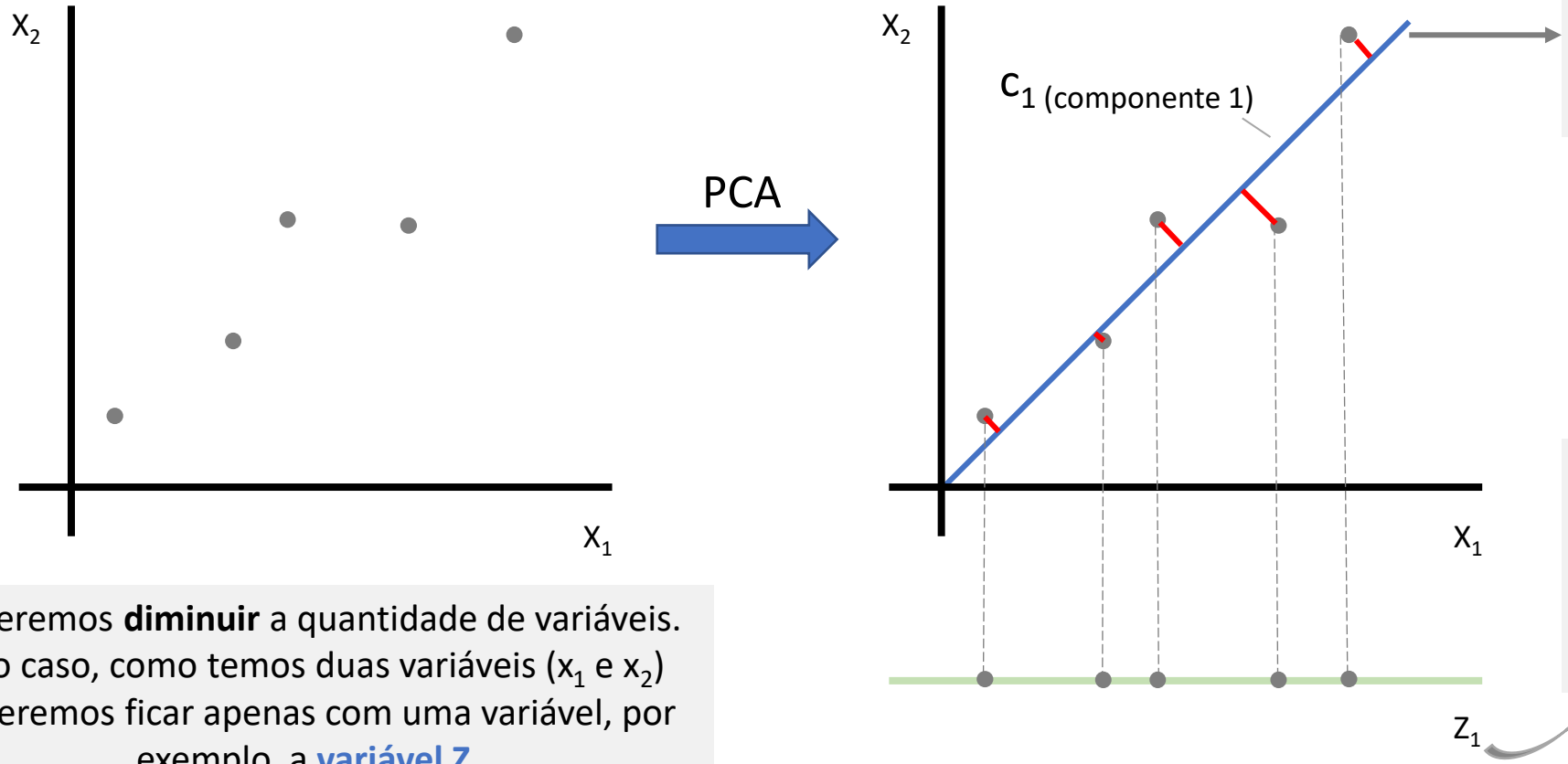
- **Algoritmos de Cluster**: são utilizados para agrupar e segmentar as observações a partir das suas características presentes nas *features*.
- **Redução de Dimensionalidade**: como o próprio nome diz, reduzem a dimensão do espaço das features (ou seja, reduz a quantidade de variáveis do dataset), muitas vezes concentrando a informação em um espaço dimensional menor do que o original.
- **Detecção de Anomalias**: são utilizados para capturar padrões que permitam identificar eventos raros.

Redução de Dimensionalidade

PCA: Intuição

Preditiva.ai

A ideia central de algoritmos de redução de dimensionalidade é reduzir a quantidade de variáveis (ou dimensões) de um conjunto de dados. Para entender a motivação do principal algoritmo deste tipo, o **PCA (Principal Component Analysis)**, vamos considerar um conjunto de variáveis X_1 e X_2.



Queremos **diminuir** a quantidade de variáveis. No caso, como temos duas variáveis (x_1 e x_2) queremos ficar apenas com uma variável, por exemplo, a **variável Z_1**.

Para isso, buscamos uma reta (ou **componente**) que **minimiza a distância dos pontos à reta**.

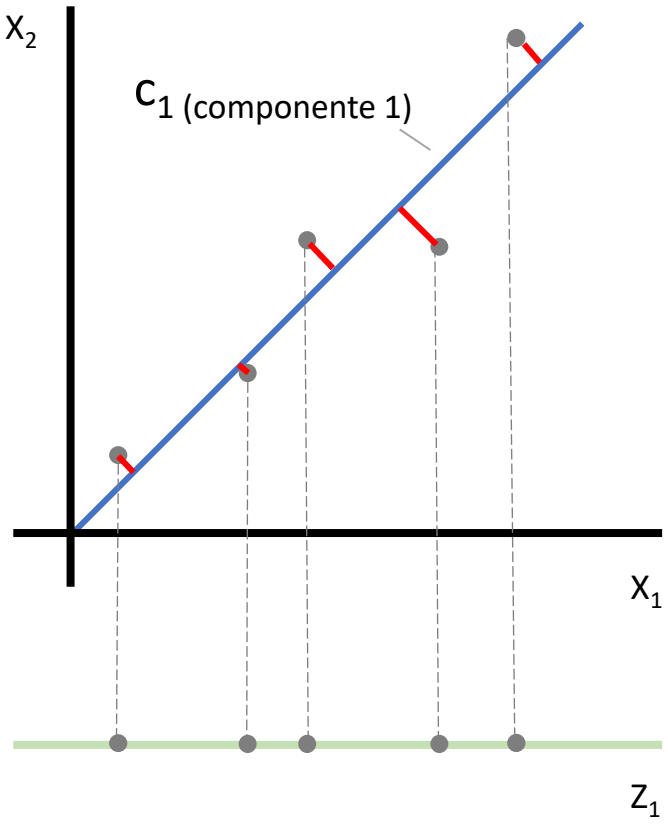
Desta forma, podemos usar esse componente para chegar à **uma variável Z_1 que seria a "redução" das variáveis x_1 e x_2**. A variável Z_1 é o resultado final do algoritmo.

Redução de Dimensionalidade

PCA: Algoritmo

Preditiva.ai

Para encontrar o componente, o PCA usa um procedimento de álgebra linear chamado **SVD (Singular Value Decomposition).** Esse procedimento recebe a matriz de variáveis (no caso, as variáveis x1 e x2) e retorna outras 3 matrizes conforme mostrado no algoritmo abaixo:



Passo 1) Fixamos X como a matriz dos variáveis a serem reduzidas *

Passo 2) Aplicamos o procedimento SVD à matriz X.

$$SVD(X) = U\ , S\ e\ V$$

$$\mathbf{V} = \begin{pmatrix} | & | & & | \\ \mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_n \\ | & | & & | \end{pmatrix}$$

A matriz V (chamada de **Matriz de Componentes Principais**) armazena os componentes que precisamos para fazer a redução de dimensionalidade que queremos.

Passo 3) Contruímos a variável Z da seguinte forma:

$$z = X * V(\mathrm{k})$$

A matriz V contém N colunas, sendo N a quantidade de variáveis da matriz X. **O valor de k significa o tamanho da dimensão da redução**. No exemplo, k = 1 pois queremos reduzer DUAS variáveis para UMA. Portanto, para encontrar Z devemos fazer o produto das matrizes X e a matriz das primeiras k colunas de V.

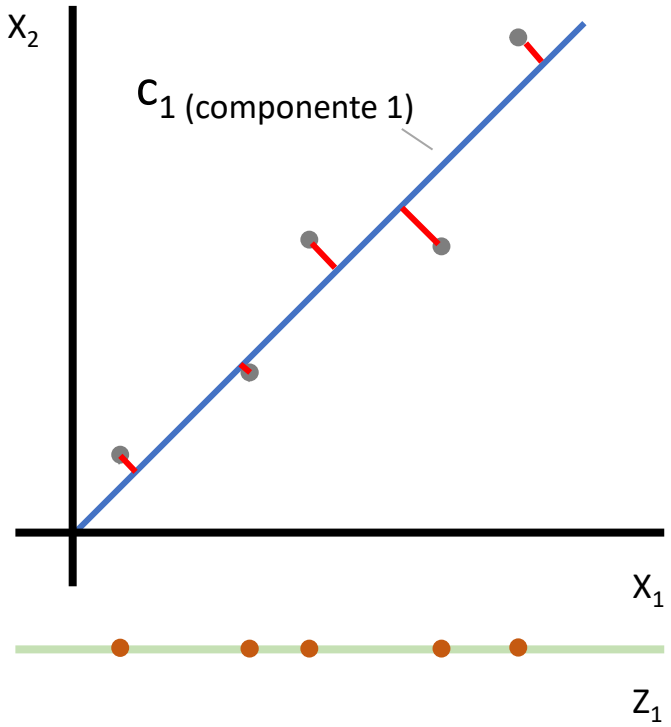
* A matriz X deve estar "centralizada", ou seja, sua média deve ser zero. Para fazer isso, basta subtrair de cada valor das variáveis a média da variável.

Redução de Dimensionalidade

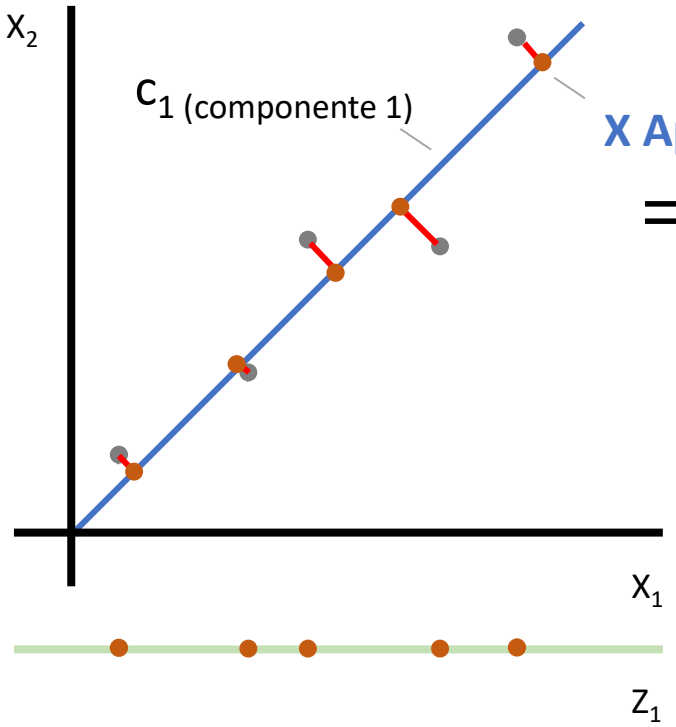
PCA: Algoritmo

Preditiva.ai

Existe um trade-off ao usar o PCA. Como vimos, o método busca reduzir a dimensionalidade do conjunto de dados, porém essa redução tem um preço: **a perda de informação**. Isso acontece pois ao reduzir as variáveis para uma variável Z, essa variável contém apenas uma "aproximação" do que era o conjunto de dados original. Veja um exemplo:



O que acontece ao tentarmos "**voltar**"para os dados originais?



X Aproximado

$$= V(\mathrm{k}) * Z$$

O método PCA encontra uma aproximação (redução) dos dados originais. Essa aproximação é armazenada na variável Z.

Com os dados da variável Z, o máximo que é possível fazer é **inferir** o valor dos dados originais usando os componentes principais. No exemplo, perceba que os dados de z (pontos laranjas) ficaram apenas próximos dos dados orginais (pontos cinzas).

Redução de Dimensionalidade

PCA: Variabilidade

Preditiva.ai

No PCA, essa perda de informação é chamada de **Retenção da Variabilidade** obtida pelos componentes principais. Essa variabilidade é calculada da seguinte forma:

Variabilidade retida pelos componentes $=$ $$1 - \frac{\frac{1}{m}\sum_{i=1}^{m} \left\| x_i - x_{aproximado} \right\|^2}{\frac{1}{m}\sum_{i=1}^{m} \left\| x_i \right\|^2}$$

Sendo **m** a quantidade de linhas do conjunto de dados

Ou ainda,

Variabilidade retida pelos primeiros k componentes $=$ $$1 - \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}}$$

Sendo **k** a quantidade componentes usados na redução e **n** a quantidade de variáveis do conjunto de dados original

Preditiva.ai

Demonstração

PCA

Arquivo: 7_2_RedDim_PCA_Demo.ipynb

Redução de Dimensionalidade

PCA

Preditiva.ai

Hands on

Rode o algoritmo PCA (com dois componentes principais) na base de Vinhos. É possível ver algum padrão?

Roteiro:

1. Importe a base “winequality-red.csv”;
2. Treine um PCA de 2 componentes. Verifique o percentual de retenção de variabilidade.
3. Plote as variáveis Z_1 e Z_2 em um gráfico de dispersão. Existem clusters bem definidos?
4. Treine um modelo de classificação para o target Quality >= 7 usando apenas as duas variáveis Z_1 e Z_2.
5. Verifique o desempenho do classificador com o PCA e sua versão sem o PCA. O que você pode concluir?

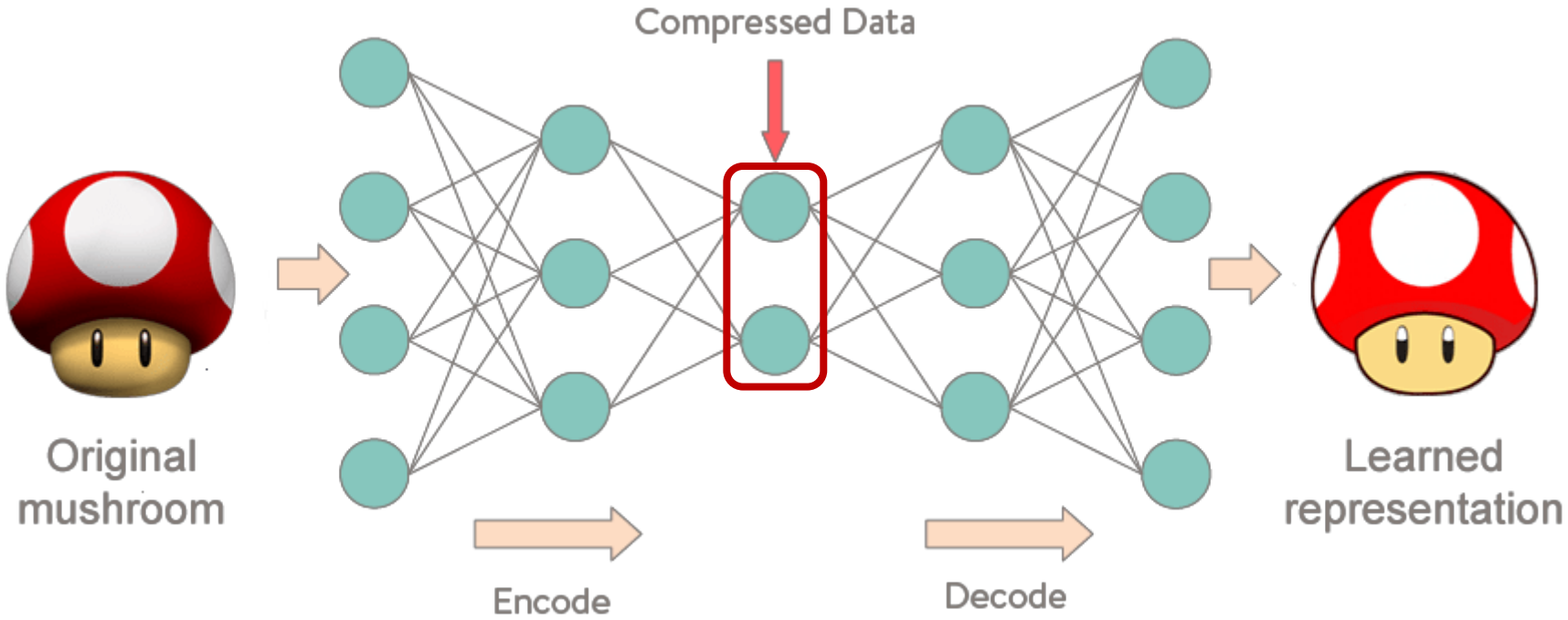
Redução de Dimensionalidade

AutoEncoder

Preditiva.ai

AutoEncoder é uma estrutura de Redes Neurais Artificiais bastante versátil que pode ser utilizada para algumas finalidades, entre elas a **Redução de Dimensionalidade**. Nessa tarefa entram em ação as mesmas 2 partes que vimos na Detecção de Anomalias: **Encode** e **Decode**.

A diferença é que para **Redução de Dimensionalidade** estamos interessados na **camada central**, que contém uma **representação comprimida dos inputs**.



Fonte: http://deeplearningbook.com.br/introducao-aos-autoencoders/

Preditiva.ai

Demonstração

AutoEncoder

Arquivo: 7_2_RedDim_AutoEncoder_tSNE_Demo.ipynb

Redução de Dimensionalidade

t-SNE

Preditiva.ai

O algoritmo **t-SNE**, ou t-Distributed Stochastic Neighbor Embedding, é uma técnica muito utilizada para **representar graficamente** um conjunto de dados de alta dimensionalidade em apenas 2 ou 3 dimensões.

Ao contrário dos outros algoritmos, a transformação da redução de dimensionalidade do **t-SNE não é reversível**. Por esse motivo, está técnica é utilizada para geração de gráficos em 2 ou 3 dimensões, e não como ***Feature Engineering*** para modelos.

Fonte: https://towardsdatascience.com/t-sne-python-example-1ded9953f26

Redução de Dimensionalidade

t-SNE

Preditiva.ai

A redução de dimensionalidade utilizando transformações do **t-SNE** são realizadas em 3 etapas:

1. Construção de uma **distribuição de probabilidades** de forma que observações com características semelhantes tem maior probabilidade de ficarem próximas, enquanto observações com características diferentes tem baixa probabilidade de ficarem próximas.

2. Definição de uma **distribuição de probabilidades** semelhante a do passo 1, mas agora considerando a representação de **menor dimensionalidade.**

3. **Otimização da distância** Euclidiana entre as 2 distribuições.

Preditiva.ai

Demonstração

t-SNE

Arquivo: 7_2_RedDim_AutoEncoder_tSNE_Demo.ipynb

Redução de Dimensionalidade

PCA vs. AutoEncoder vs. t-SNE

Preditiva.ai

A principal diferença entre o **PCA**, **AutoEncoder** e **t-SNE** é que as transformações utilizadas no **PCA** são **lineares** enquanto o **AutoEncoder** e **t-SNE** podem ser **não-lineares**.

Técnica	Transformações	Reversível	Velocidade Treinamento	Ortogonalidade das variáveis	Hiperparâmetros
PCA	Lineares	Sim	Rápida	Sim	Número de componentes
AutoEncoder	Lineares e Não-lineares	Sim	Lenta	Não	Estrutura da RNA
t-SNE	Lineares e Não-lineares	Não	Média	Não	Número de componentes e outros

Redução de Dimensionalidade

AutoEncoder e t-SNE

Preditiva.ai

Hands on

Desenvolva um modelo AutoEncoder com compressão em 2 dimensões na base de Vinhos. É possível ver algum padrão? E utilizando o t-SNE, é possível ver clusters?

Roteiro:

1. Importe a base “winequality-red.csv”;

2. Treine um AutoEncoder com 2 dimensões. Verifique o erro na reconstrução (*loss*).

3. Plote as dimensões D_1 e D_2 em um gráfico de dispersão. Existem clusters bem definidos?

4. Treine um modelo de classificação para o target Quality >= 7 usando apenas as duas variáveis D_1 e D_2.

5. Verifique o desempenho do classificador com o AutoEncoder, a versão com PCA e sua versão sem o PCA. O que podemos concluir?

Preditiva.ai