

PRÉ- PROCESSAMENTO

O GRANDE DIA

ELIMINAÇÃO MANUAL DE ATRIBUTOS

Quando um atributo não contribui para a estimativa do valor do atributo alvo, ele é considerado irrelevante.

**Deve mais ou
menos!**

O conjunto de dados final deve ser definido de acordo com a experiência de especialistas no domínio dos dados.

PRÉ- PROCESSAMENTO

- **Técnicas utilizadas para melhorar a qualidade dos dados;**
- **Eliminam ou minimizam os problemas como ruídos, outliers, valores incorretos, inconsistentes, duplicados ou ausentes.**
- **Podem ainda tornar os dados mais adequados para sua utilização por um determinado algoritmo.**

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_csv("hospital.csv", sep = ';')

df = df.drop(columns=['identificador', 'nome'])

df

```

	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnostico								
0	28	M	79	Concentradas	38.0	2	SP	Doente								
1	18	F	67	Inexistentes	39.5	4	MG	Saudavel								
2	49	M	92	Espalhadas	38.0	Identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnostico	
3	18	M	43	Inexistentes	38.5	0	4201	Joao	28	M	79	Concentradas	38.0	2	SP	Doente
4	21	F	52	Uniformes	37.6	1	3217	Maria	18	F	67	Inexistentes	39.5	4	MG	Saudavel
5	22	F	72	Inexistentes	58.0	2	4039	Luiz	49	M	92	Espalhadas	38.0	2	RS	Doente
6	19	F	87	Espalhadas	39.0	3	1920	Jose	18	M	43	Inexistentes	38.5	8	MG	Saudavel
7	34	M	67	Uniformes	38.4	4	4340	Claudia	21	F	52	Uniformes	37.6	1	PE	Doente
						5	2301	Ana	22	F	72	Inexistentes	58.0	3	RJ	Doente
						6	1322	Marta	19	F	87	Espalhadas	39.0	6	AM	Saudavel
						7	3027	Paulo	34	M	67	Uniformes	38.4	2	GO	Saudavel

INTEGRAÇÃO DE DADOS

- **Busca por atributos comuns nos conjuntos a serem combinados;**
- **Atributos utilizados para combinação deve(m) ter um valor único para cada objeto.**
- **CUIDADO: nome do atributo e atualização dos dados.**

AMOSTRAGEM DE DADOS

Algoritmos de AM podem ter dificuldade em lidar com um grande número de objetos.

Balanço entre eficiência computacional e acurácia (taxa de previsões corretas).

- Maior acurácia x menor eficiência computacional.

PROBLEMA: uma amostra pode não representar bem o problema que se deseja modelar.



AMOSTRAGEM DE DADOS

A amostra deve ser representativa do conjunto de dados original.

Diferentes amostras de uma mesma população podem gerar modelos diferentes.



Os dados devem obedecer a mesma distribuição estatística que gerou o conjunto de dados original.

**A MÉDIA DOS VALORES PARA CADA ATRIBUTO
DOS DADOS ORIGINAIS DEVE SER SEMELHANTE A
DA AMOSTRA.**

CONSELHO DO DIA:



Existem basicamente três abordagens para amostragem:

- Amostragem aleatória simples
- Amostragem estratificada
- Amostragem progressiva

AMOSTRAGEM ALEATÓRIA SIMPLES

Sem reposição de exemplos: os exemplos são extraídos do conjunto original para a amostra a ser utilizada.

- Cada exemplo pode ser selecionado apenas uma vez.

Com reposição de exemplos: a probabilidade de escolher qualquer objeto se mantém constante.

AMOSTRA ESTRATIFICADA

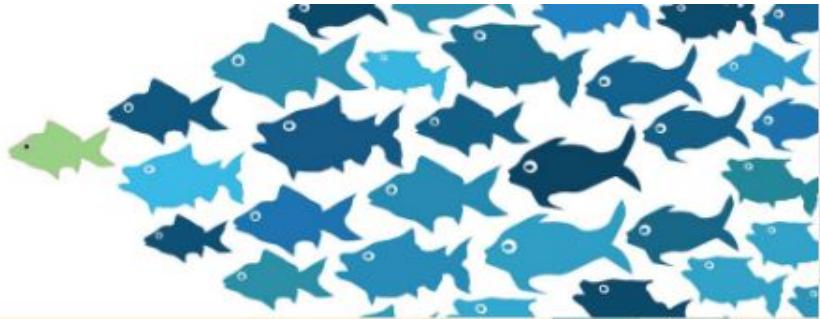
- **Usada quando as classes apresentam propriedades diferentes (Ex. Números de objetos bastante diferentes).**

- Manter o mesmo número de objetos para cada classe.
- Manter o número de objetos em cada classe proporcional ao número de objetos da classe no conjunto original.

AMOSTRA PROGRESSIVA

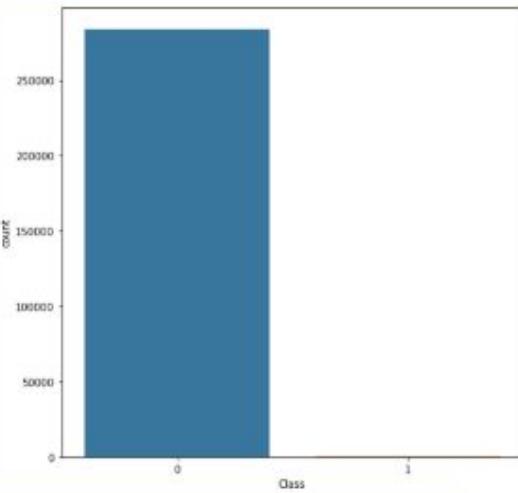
- **Começa com uma amostra pequena e aumenta progressivamente o tamanho da amostra extraída.**
- **A amostra vai aumentando enquanto a acurácia preditiva continuar a aumentar.**
- **Define-se a menor quantidade de dados necessária.**

DADOS DESBALANCEADOS

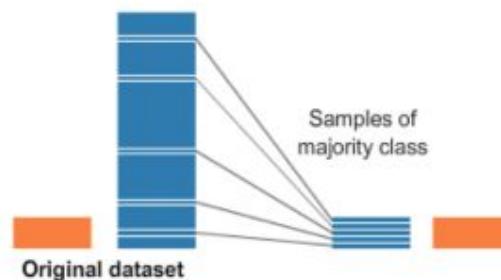


- Técnicas de balanceamento artificial:
 - Redefinir o tamanho do conjunto de dados
 - Utilizar diferentes custos de classificação para as diferentes classes
 - Induzir um modelo para uma classe

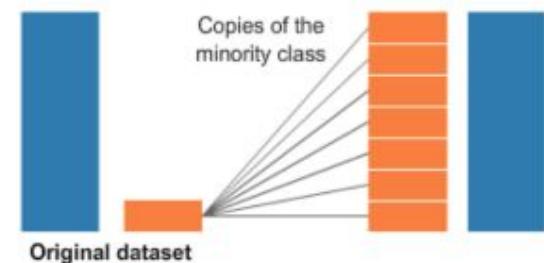
DADOS DESBALANCEADOS



Undersampling



Oversampling



DADOS DESBALANCEADOS

```
#Randomicamente seleciona 4 instâncias a partir da classe 'Doente'  
df.loc[df['diagnosticos'] == "Doente"].sample(n=4,random_state=2)
```

identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnosticos	
4	4340	Claudia	21	F	52	Uniformes	37.6	1	PE	Doente
5	2301	Ana	22	F	72	Inexistentes	58.0	3	RJ	Doente
2	4039	Luiz	49	M	92	Espalhadas	38.0	2	RS	Doente
0	4201	Joao	28	M	79	Concentradas	38.0	2	SP	Doente

As instâncias criadas comumente consideram valores de média +/- desvio padrão



Garanta que as instâncias criadas fazem sentido em relação ao esperado. Por exemplo: Uma população de 23,5 pessoas

Lembre sempre de manter a estatística descritiva da base de dados original!!

https://imbalanced-learn.readthedocs.io/en/stable/over_sampling.html#smote-adasyn

LIMPEZA DE DADOS

Problemas relacionados a qualidade dos dados:

- Dados ruidosos: possuem erros ou valores que são diferentes do esperado



DIFERENTE DE OUTLIER, que é um dado real!

- Inconsistente: não combinam ou contradizem valores de outros atributos do mesmo objeto

Que PODE
ser um dado
real!



LIMPEZA DE DADOS

- Dados ruidosos são informações que contêm erros, inconsistências ou ruídos que podem distorcer a análise e a interpretação dos dados.
- Esses dados podem surgir de várias fontes, como medições imprecisas, erros de entrada, falhas nos sensores ou interferências externas.

OBS.: Um dado ruidoso pode gerar um *outlier*



LIMPEZA DE DADOS

Problemas relacionados a qualidade dos dados:

- Redundantes (quando dois ou mais objetos têm os mesmos valores para todos os atributos ou dois ou mais atributos tem os mesmos valores para dois ou mais objetos)
- Incompletos (com ausência de valores para alguns dos atributos em parte dos dados)





LIMPEZA DE DADOS

	identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnostico
0	4201	Joao	28	M	NaN	Concentradas	38.0	2	SP	Doente
1	3217	Maria	18	F	67.0	Inexistentes	39.5	4	MG	Saudavel
2	4039	Luiz	49	M	92.0	Espalhadas	38.0	2	RS	Doente
3	1920	Jose	18	M	43.0	Inexistentes	38.5	8	MG	Saudavel
4	4340	Claudia	21	F	52.0	Uniformes	NaN	1	PE	Doente
5	2301	Ana	22	F	72.0	Inexistentes	58.0	3	RJ	Doente
6	2301	Ana	22	F	72.0	Inexistentes	58.0	3	RJ	Doente
7	2301	Ana	22	F	72.0	Inexistentes	58.0	3	RJ	Doente
8	1322	Marta	19	F	87.0	Espalhadas	39.0	6	AM	Saudavel
9	3027	Paulo	34	M	67.0	Uniformes	38.4	2	GO	Saudavel

```
print('Valores faltantes: ', df.isnull().sum())
print('Valores duplicados: ', df.duplicated())

Quantidade de valores faltantes:  identificador      0
nome                  0
idade                 0
sexo                  0
peso                  1
manchas                0
temperatura              1
internacoes              0
estado                  0
diagnostico                0
dtype: int64
Quantidade de valores duplicados:  0      False
1      False
2      False
3      False
4      False
5      False
6      True
7      True
8      False
9      False
dtype: bool
```



DADOS INCOMPLETOS

Eliminar os objetos com valores ausentes.

Definir e preencher manualmente valores para os atributos com valores ausentes.

Empregar algoritmos de AM que lidam internamente com valores ausentes (algoritmos indutores de árvores de decisão)

Utilizar algum método ou heurística para automaticamente definir valores para os atributos com valores ausentes.

DADOS INCOMPLETOS

Criar para o atributo um novo valor que indique que o atributo possuía um valor desconhecido.

Utilizar a média, moda ou mediana dos valores conhecidos para esse atributo.

Empregar um indutor para estimar o valor do atributo. É a mais popular!!!

- Utilização do valor utilizado em objetos semelhantes!!

DADOS INCOMPLETOS

```
#Substituir o valor faltante pela mediana  
median = df['peso'].median()  
df['peso'].fillna(median, inplace=True)
```

```
df
```

```
Valores faltantes: identificador      0  
nome          0  
idade          0  
sexo           0  
peso           1  
manchas        0  
temperatura    1  
internacoes    0  
estado          0  
diagnostico    0  
dtype: int64
```

	identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnostico
0	4201	Joao	28	M	72.0	Concentradas	38.0	2	SP	Doente
1	3217	Maria	18	F	67.0	Inexistentes	39.5	4	MG	Saudavel

DADOS INCOMPLETOS

```
# Elimina todas as linhas com dados ausentes  
df = df.dropna(how='any')  
  
df
```

	identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnostico
1	3217	Maria	18	F	67.0	Inexistentes	39.5	4	MG	Saudavel
2	4039	Luiz	49	M	92.0	Espalhadas	38.0	2	RS	Doente
3	1920	Jose	18	M	43.0	Inexistentes	38.5	8	MG	Saudavel
5	2301	Ana	22	F	72.0	Inexistentes	58.0	3	RJ	Doente
6	2301	Ana	22	F	72.0	Inexistentes	58.0	3	RJ	Doente
7	2301	Ana	22	F	72.0	Inexistentes	58.0	3	RJ	Doente
8	1322	Marta	19	F	87.0	Espalhadas	39.0	6	AM	Saudavel
9	3027	Paulo	34	M	67.0	Uniformes	38.4	2	GO	Saudavel

DADOS INCONSISTENTES

Possibilidades!!!

- Problema na anotação dos dados!

- Os atributos de entrada não explicam o atributo alvo!

	identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnostico
0	4201	Joao	28	M	67	Concentradas	38.0	2	SP	Doente
1	3217	Maria	18	F	67	Inexistentes	39.5	4	MG	Saudavel
2	4039	Lulz	49	M	92	Espalhadas	38.0	2	RS	Doente
3	1920	Jose	18	M	43	Inexistentes	38.5	8	MG	Saudavel
4	4340	Claudia	21	F	52	Uniformes	38.5	1	PE	Doente
5	4340	Claudia	21	F	52	Uniformes	38.5	1	PE	Saudavel
6	2301	Ana	22	F	72	Inexistentes	58.0	3	RJ	Doente
7	2301	Ana	22	F	72	Inexistentes	58.0	3	RJ	Doente
8	2301	Ana	22	F	72	Inexistentes	58.0	3	RJ	Doente
9	1322	Marta	19	F	87	Espalhadas	39.0	6	AM	Saudavel
10	3027	Paulo	34	M	67	Uniformes	38.4	2	GO	Saudavel

DADOS DUPLICADOS

```
# Elimina todas as linhas com dados duplicados  
df.drop_duplicates()
```

	identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes	estado	diagnostico
0	4201	Joao	28	M	NaN	Concentradas	38.0	2	SP	Doente
1	3217	Maria	18	F	67.0	Inexistentes	39.5	4	MG	Saudavel
2	4039	Luiz	49	M	92.0	Espalhadas	38.0	2	RS	Doente
3	1920	Jose	18	M	43.0	Inexistentes	38.5	8	MG	Saudavel
4	4340	Claudia	21	F	52.0	Uniformes	NaN	1	PE	Doente
5	2301	Ana	22	F	72.0	Inexistentes	58.0	3	RJ	Doente
8	1322	Marta	19	F	87.0	Espalhadas	39.0	6	AM	Saudavel
9	3027	Paulo	34	M	67.0	Uniformes	38.4	2	GO	Saudavel

REDUNDÂNCIA DE DADOS



- **Boosting = duplica-se a quantidade de exemplos difíceis de serem classificados.**
- **Redundância de atributos (idade x data de nascimento).**
- **Alta correlação entre atributos.**
OS ATRIBUTOS TRAZEM A MESMA INFORMAÇÃO EM RELAÇÃO AO ATRIBUTO ALVO - MANTENHA APENAS UM!!!



LEMBRANDO!!!

```
corr = df.corr()  
corr.style.background_gradient(cmap='coolwarm')
```

	identificador	idade	peso	temperatura	internacoes
identificador	1	0.48618	0.0620799	-0.319808	-0.817134
idade	0.48618	1	0.560835	-0.191917	-0.512349
peso	0.0620799	0.560835	1	0.0593191	-0.28261
temperatura	-0.319808	-0.191917	0.0593191	1	-0.035277
internacoes	-0.817134	-0.512349	-0.28261	-0.035277	1

Dados que contêm objetos que, aparentemente, não pertencem a distribuição que gerou os dados analisados.

- São identificados como observações que diferem de uma distribuição utilizada na modelagem dos dados.
 - São identificados como objetos pertencentes a níveis superficiais.



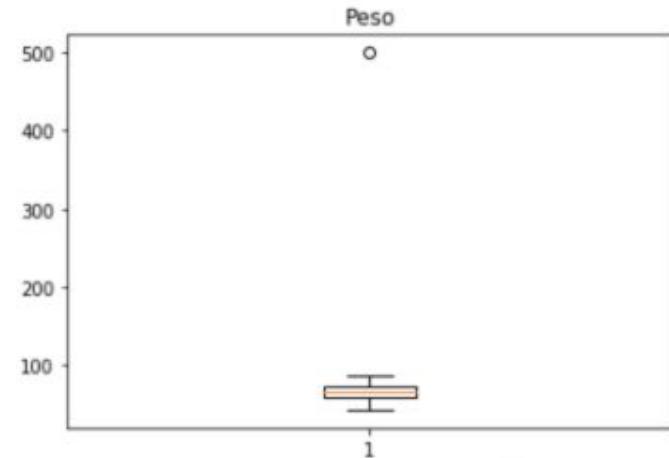
Esses dados, se forem reais, devem ser estudados, sabendo que existe a possibilidade de outros casos semelhantes surgirem



Não sendo reais, devem ser recuperados, eliminados ou estimados!!

```
df = pd.read_csv("hospital.csv", sep = ';')  
plt.boxplot(df['peso'])  
plt.title('Peso')
```

Text(0.5, 1.0, 'Peso')



TRANSFORMAÇÃO DE DADOS

- Conversão simbólico-numérico
 - Ausente: O Presente: 1
 - Menor valor: O Outro valor: 1
 - Cada valor nominal corresponde a uma sequência de c bits, onde c é igual ao número de possíveis valores ou categorias.



Necessário quando o método de AM exige dados numéricos!!!

TRANSFORMAÇÃO

ONE-HOT-ENCODING

CADA CATEGORIA DEVE SER RENOMEADA POR UM NÚMERO:

MUITO BOM, BOM, RUIM E MUITO RUIM (1, 2, 3 E 4)

ESSA TRANSFORMAÇÃO É CONHECIDA POR LABEL ENCODING OU INTEGER ENCODING

ESSA TRANSFORMAÇÃO É COMUMENTE USADA QUANDO AS CATEGORIAS REPRESENTAM UMA SEQUÊNCIA

EM CASOS DIFERENTES, USA-SE ONE HOT ENCODING, ONDE O ATRIBUTO É BINARIZADO E TRANSFORMADO EM UM CONJUNTO DE ATRIBUTOS (0S E 1S)

TRANSFORMAÇÃO

ONE-HOT-ENCODING

1	red,	green,	blue
2	1,	0,	0
3	0,	1,	0
4	0,	0,	1

Rome = [1, 0, 0, 0, 0, 0, ..., 0]
Paris = [0, 1, 0, 0, 0, 0, ..., 0]
Italy = [0, 0, 1, 0, 0, 0, ..., 0]
France = [0, 0, 0, 1, 0, 0, ..., 0]



```
from sklearn import preprocessing

le = preprocessing.LabelEncoder()
for column in df.columns:
    if df[column].dtypes == 'object':
        df[column] = le.fit_transform(df[column])

print(df)
```

	identificador	nome	idade	sexo	peso	manchas	temperatura	internacoes
0	4201	2	28	1	67	0	38.0	2
1	3217	5	18	0	67	2	39.5	4
2	4039	4	49	1	500	1	38.0	2
3	1920	3	18	1	43	2	38.5	8
4	4340	1	21	0	52	3	38.5	1
5	4340	1	21	0	52	3	38.5	1
6	2301	0	22	0	72	2	58.0	3
7	2301	0	22	0	72	2	58.0	3
8	2301	0	22	0	72	2	58.0	3
9	1322	6	19	0	87	1	39.0	6
10	3027	7	34	1	67	3	38.4	2

TRANSFORMAÇÃO DE DADOS

- Métodos de discretização permitem transformar atributos quantitativos em qualitativos.
- Os valores numéricos são transformados em intervalos ou categorias.

TRANSFORMAÇÃO DE ATRIBUTOS NUMÉRICOS

Quando os limites inferior e superior de valores dos atributos são muito diferentes ou estão em escalas diferentes.



Normalização = evita que um atributo predomine sobre outro, mas não existe garantia.

- Amplitude
- Distribuição

NORMALIZAÇÃO

$$X_{changed} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Por reescala =

define uma nova escala de valores, limites mínimo e máximo, para todos os atributos.

$$X_{changed} = \frac{X - \mu}{\sigma}$$

Padronização =

define um valor central e um valor de espalhamento comum para todos os atributos.

- Lida melhor com outliers.



NORMALIZAÇÃO

```
from sklearn.preprocessing import RobustScaler  
  
rs = RobustScaler()  
X = rs.fit_transform(df)  
  
print(X)  
  
[[ 1.2000000e+00  0.0000000e+00 -4.85436893e-02 -6.66666667e-01]  
 [-8.0000000e-01  0.0000000e+00  9.70873786e-02  6.66666667e-01]  
 [ 5.4000000e+00  3.4640000e+01 -4.85436893e-02 -6.66666667e-01]  
 [-8.0000000e-01 -1.9200000e+00  0.0000000e+00  3.3333333e+00]  
 [-2.0000000e-01 -1.2000000e+00  0.0000000e+00 -1.3333333e+00]  
 [-2.0000000e-01 -1.2000000e+00  0.0000000e+00 -1.3333333e+00]  
 [ 0.0000000e+00  4.0000000e-01  1.89320388e+00  0.0000000e+00]  
 [ 0.0000000e+00  4.0000000e-01  1.89320388e+00  0.0000000e+00]  
 [ 0.0000000e+00  4.0000000e-01  1.89320388e+00  0.0000000e+00]  
 [-6.0000000e-01  1.6000000e+00  4.85436893e-02  2.0000000e+00]  
 [ 2.4000000e+00  0.0000000e+00 -9.70873786e-03 -6.66666667e-01]]
```

NORMALIZAÇÃO

```
x_array = np.array(df['peso'])
normalized_X = preprocessing.normalize([x_array])

normalized_X

array([[0.1235992 , 0.1235992 , 0.92238207, 0.07932486, 0.09592774,
       0.09592774, 0.13282302, 0.13282302, 0.13282302, 0.16049448,
       0.1235992 ]])
```

```
from sklearn.preprocessing import RobustScaler

rs = RobustScaler()
X = rs.fit_transform(X)

X

array([[ 1.20000000e+00,  0.00000000e+00, -4.85436893e-02,
       -6.66666667e-01],
      [-8.00000000e-01,  0.00000000e+00,  9.70873786e-02,
       6.66666667e-01],
      [ 5.40000000e+00,  3.46400000e+01, -4.85436893e-02,
       -6.66666667e-01],
      [-8.00000000e-01, -1.92000000e+00,  0.00000000e+00,
       3.33333333e+00],
      [-2.00000000e-01, -1.20000000e+00,  0.00000000e+00,
       -1.33333333e+00],
      [-2.00000000e-01, -1.20000000e+00,  0.00000000e+00,
       -1.33333333e+00],
      [ 0.00000000e+00,  4.00000000e-01,  1.89320388e+00,
       0.00000000e+00],
      [ 0.00000000e+00,  4.00000000e-01,  1.89320388e+00,
       0.00000000e+00],
      [ 0.00000000e+00,  4.00000000e-01,  1.89320388e+00,
       0.00000000e+00],
      [-6.00000000e-01,  1.60000000e+00,  4.85436893e-02,
       2.00000000e+00],
      [ 2.40000000e+00,  0.00000000e+00, -9.70873786e-03,
       -6.66666667e-01]])
```

Vazamento de Dados

1. **Normalização e Padronização:** Realizar a normalização ou padronização dos dados deve ser feito após a divisão dos conjuntos.
2. **Tratamento de Dados Ausentes:** Impute ou remova dados ausentes depois da divisão.
3. **Codificação de Variáveis Categóricas:** A codificação deve ser realizada de forma que a variável categórica não revele informações do conjunto de teste ao conjunto de treinamento. Use técnicas como `One-Hot Encoding` após a divisão.

Vazamento de Dados

4. **Remoção de *Outliers*:** Identifique e remova *outliers* depois da divisão para garantir que o modelo não seja influenciado por dados extremos presentes no conjunto completo e no teste.
5. **Divisão Aleatória:** Realize a divisão de forma aleatória para garantir que ambos os conjuntos (treinamento e teste) sejam representativos da distribuição original dos dados. Guarde a semente de aleatoriedade!!
6. **Validação Cruzada:** Utilize validação cruzada (no conjunto de treinamento) para avaliar o modelo de forma robusta, garantindo que a avaliação não se baseie em um único conjunto de teste.
7. **Separação de Dados Temporais:** Se os dados forem temporais, mantenha a ordem temporal ao dividir os conjuntos, evitando que informações futuras sejam usadas para prever o passado.

REDUÇÃO DE DIMENSIONALIDADE

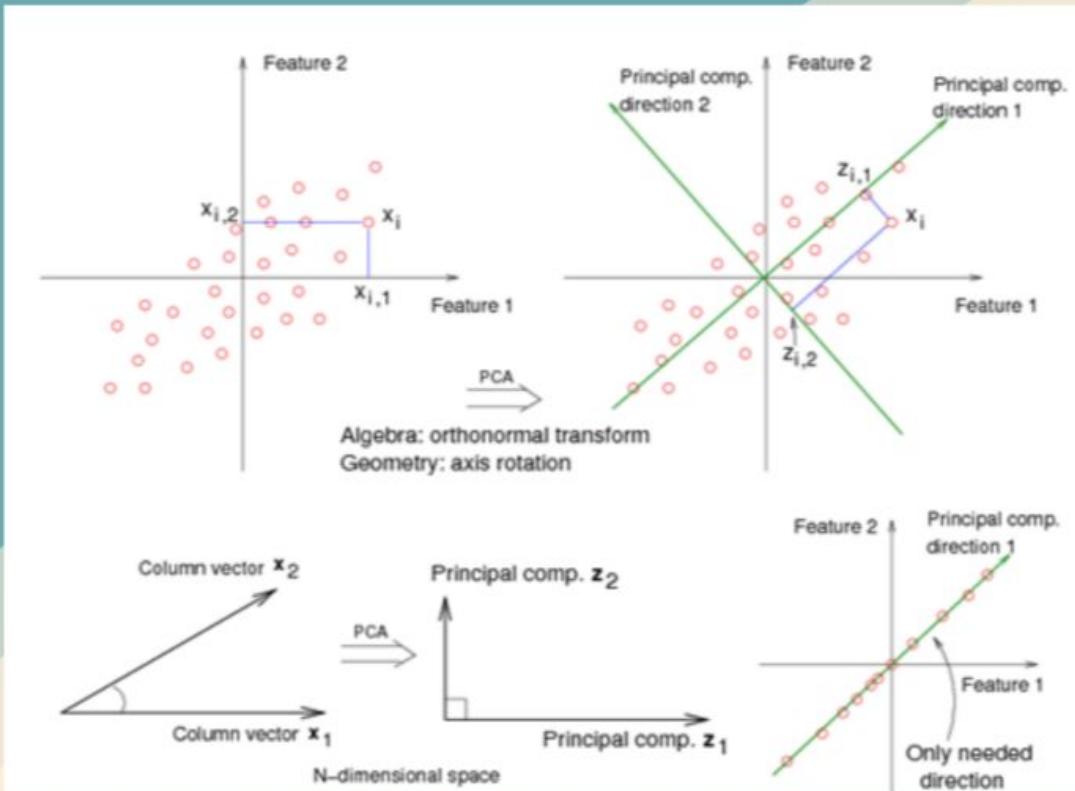


- Combina ou elimina parte dos atributos irrelevantes.
- Abordagens:
 - Agregação: novos atributos formados pela combinação de grupos de atributos
 - Seleção de atributos: mantém parte dos atributos

ANÁLISE DOS COMPONENTES PRINCIPAIS - PCA (PRINCIPAL COMPONENTS ANALYSIS)

- A técnica foi descrita por Pearson (1901);
- Uma descrição de métodos computacionais práticos veio com Hotelling (1933, 1936) que usou com o propósito determinado de analisar as estruturas de correlação.
- O PCA é uma técnica estatística de análise multivariada;
- Transforma linearmente um conjunto original de variáveis, inicialmente correlacionadas entre si, num conjunto substancialmente menor de variáveis não correlacionadas que contém a maior parte da informação do conjunto original.

PCA



https://www.bogotobogo.com/python/scikit-learn/scikit_machine_learning_Data_Compression_via_Dimensionality_Reduction_1_Principal_component_analysis%20_PCA.php

ANÁLISE DOS COMPONENTES PRINCIPAIS - PCA (PRINCIPAL COMPONENTS ANALYSIS)

- Os componentes principais apresentam propriedades importantes:
 - Cada componente principal é uma combinação linear de todas as variáveis originais;
 - São independentes entre si e estimados com o propósito de reter, em ordem de estimação, o máximo de informação, em termos da variação total contida nos dados.

Além de retirar a multicolinearidade das variáveis, reduz muitas variáveis a eixos que representam algumas delas, sendo estes eixos perpendiculares (ortogonais) explicando a variação dos dados de forma decrescente e independente.

Combinação linear: Cada componente principal é uma combinação linear das variáveis originais, ou seja, é formada ao multiplicar cada variável por um peso e somar esses valores. Por exemplo, se você tem variáveis $x_1, x_2 \dots x_n$ são os pesos que definem a contribuição de cada variável para o componente.

$$PC_1 = a_1x_1 + a_2x_2 + \dots + a_nx_n$$

onde os coeficientes $a_1, a_2 \dots a_n$ são os pesos que definem a contribuição de cada variável para o componente

Autovalores e autovetores: Para encontrar esses pesos, o PCA calcula a matriz de covariância dos dados e resolve um problema de álgebra linear para encontrar os autovetores e autovalores dessa matriz.

- Os autovetores indicam as direções no espaço dos dados onde a variância é máxima – ou seja, as direções dos componentes principais.
- Os autovalores correspondem à quantidade de variância que cada autovetor (componente) explica nos dados. O primeiro componente principal é o autovetor com o maior autovalor, capturando a maior parte da variação dos dados.

PCA



Bisavó: Ouvi dizer que você está estudando "Pee-Cee-Aa". Eu imagino o que isso seja...

Você: Ah, é apenas um método de resumir alguns dados. Olha, temos algumas garrafas de vinho aqui em cima da mesa. Podemos descrever cada vinho por sua cor, por sua força, por sua idade e assim por diante. Podemos compor uma lista completa de características diferentes de cada vinho em nossa adega. Mas muitos deles medirão propriedades relacionadas e, portanto, serão redundantes. Nesse caso, poderemos resumir cada vinho com menos características! É isso que o PCA faz.

PCA



Avô: Isso é interessante! Portanto, essa coisa do PCA verifica quais características são redundantes e as descarta?

Você: Excelente pergunta, vovô! Não, o PCA não está selecionando algumas características e descartando outras. Em vez disso, constrói algumas novas características que acabam resumindo bem nossa lista de vinhos. É claro que essas novas características são construídas usando as antigas; por exemplo, uma nova característica pode ser calculada como a idade do vinho menos o nível de acidez do vinho ou alguma outra combinação como essa (chamamos de combinações lineares). De fato, o PCA encontra as melhores características possíveis, aquelas que resumem a lista de vinhos e somente são possíveis (entre todas as combinações lineares concebíveis). É por isso que é tão útil.

PCA



Mãe: Hmmm, isso certamente parece bom, mas não tenho certeza se entendi. O que você realmente quer dizer quando diz que essas novas características do PCA "resumem" a lista de vinhos?

Você: Acho que posso dar duas respostas diferentes para esta pergunta. A primeira resposta é que você está procurando algumas propriedades (características) do vinho que diferem fortemente entre os vinhos. De fato, imagine que você tenha uma propriedade igual para a maioria dos vinhos. Isso não seria muito útil, seria? Os vinhos são muito diferentes, mas sua nova propriedade faz com que todos pareçam iguais! Este certamente seria um resumo ruim. Em vez disso, o PCA procura propriedades que mostrem o máximo de variação possível entre os vinhos.

PCA



A segunda resposta é que você procura as propriedades que permitem prever ou "reconstruir" as características originais do vinho. Mais uma vez, imagine que você tenha uma propriedade que não tenha relação com as características originais; se você usar apenas essa nova propriedade, não há como reconstruir as originais! Novamente, isso seria um resumo ruim. Portanto, o PCA procura propriedades que permitam reconstruir as características originais da melhor maneira possível. Surpreendentemente, verifica-se que esses dois objetivos são equivalentes e, portanto, o PCA pode matar dois coelhos com uma cajadada só.

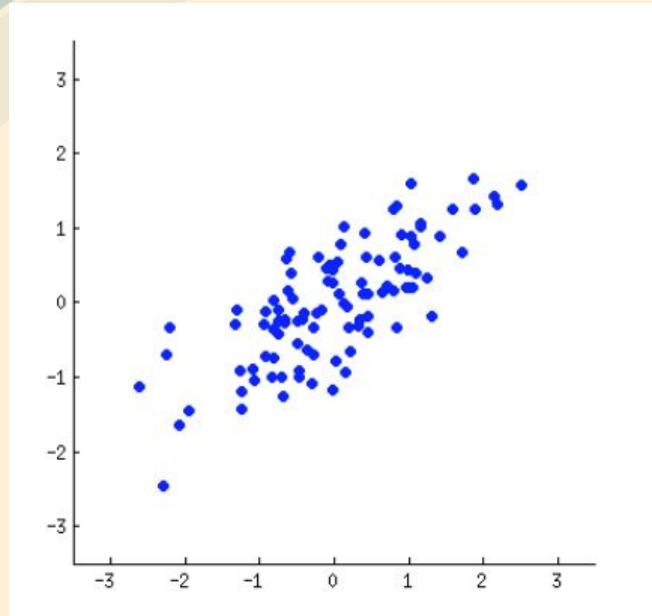


PCA



Namorado: Mas, querida, esses dois "objetivos" do PCA parecem tão diferentes! Por que eles seriam equivalentes?

Você: Hmmm. Talvez eu deva fazer um pequeno desenho (pega um guardanapo e começa a rabiscar). Vamos escolher duas características do vinho, talvez a escuridão do vinho e o teor alcoólico - não sei se elas estão correlacionadas, mas vamos imaginar que sim. Aqui está a aparência de um gráfico de dispersão de diferentes vinhos:



PCA

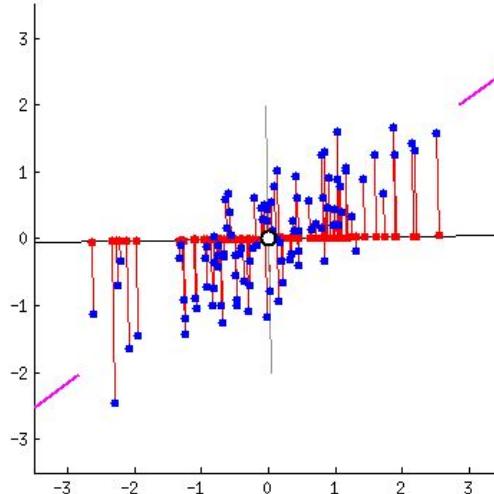


Cada ponto nesta "nuvem de vinho" mostra um vinho em particular. Você vê que as duas propriedades (x e y nesta figura) estão correlacionadas. Uma nova propriedade pode ser construída traçando uma linha através do centro dessa nuvem de vinho e projetando todos os pontos nessa linha. Esta nova propriedade será dada por uma combinação linear $w_1x + w_2y$, onde cada linha corresponde a alguns valores particulares de w_1 e w_2 .

PCA



Agora olhe aqui com muito cuidado - eis como essas projeções se parecem com linhas diferentes (pontos vermelhos são projeções dos pontos azuis):



PCA



Como eu disse antes, o PCA encontrará a linha "melhor" de acordo com dois critérios diferentes do que é o "melhor". Primeiro, a variação dos valores nessa linha deve ser máxima. Preste atenção em como a "dispersão" (que chamamos de "variação") dos pontos vermelhos muda enquanto a linha gira; você pode ver quando atinge o máximo? Segundo, se reconstruirmos as duas características originais (posição de um ponto azul) a partir da nova (posição de um ponto vermelho), o erro de reconstrução será dado pelo comprimento da linha vermelha de conexão. Observe como o comprimento dessas linhas vermelhas muda enquanto a linha gira; você pode ver quando o comprimento total atinge o mínimo?

PCA



Se você observar esta animação por algum tempo, notará que "a variação máxima" e "o erro mínimo" são atingidos ao mesmo tempo, ou seja, quando a linha aponta para os tiques magentas que marquei nos dois lados da nuvem de vinho . Esta linha corresponde à nova propriedade do vinho que será construída pela PCA. A propósito,

PCA significa "análise de componentes principais" e essa nova propriedade é chamada "primeiro componente principal". E, em vez de dizer "propriedade" ou "característica", geralmente dizemos "característica" ou "variável"

PCA



Filha prodígio e gênia: Muito bom, mamãe! Acho que posso entender por que os dois objetivos produzem o mesmo resultado: é essencialmente por causa do teorema de Pitágoras, não é? De qualquer forma, ouvi dizer que o PCA está de alguma forma relacionado a autovetores e autovalores; onde eles estão nessa foto?

Você: Observação brilhante. Matematicamente, a dispersão dos pontos vermelhos é medida como a distância quadrada média do centro da nuvem de vinho a cada ponto vermelho; como você sabe, é chamado de variação. Por outro lado, o erro total de reconstrução é medido como o comprimento quadrado médio das linhas vermelhas correspondentes. Mas como o ângulo entre as linhas vermelhas e a linha preta é sempre de 90°, a soma dessas duas quantidades é igual à distância quadrada média entre o centro da nuvem de vinho e cada ponto azul; esse é precisamente o teorema de Pitágoras.

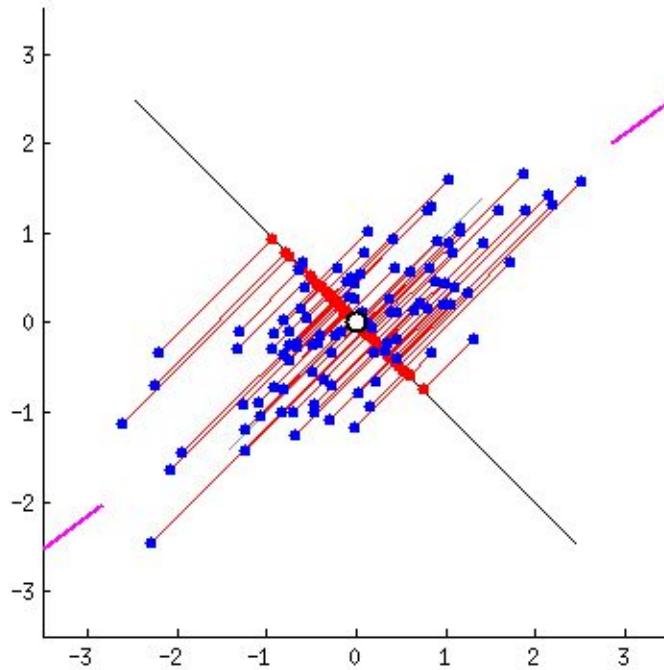
PCA



É claro que essa distância média não depende da orientação da linha preta; portanto, quanto maior a variação, menor o erro (porque a soma é constante).

A propósito, você pode imaginar que a linha preta é uma haste sólida e cada linha vermelha é uma mola. A energia da mola é proporcional ao seu comprimento ao quadrado (isso é conhecido na física como lei de Hooke), de modo que a haste se orientará de modo a minimizar a soma dessas distâncias ao quadrado. Fiz uma simulação de como será, na presença de algum atrito viscoso:

PCA



PCA



Em relação aos autovetores e autovalores. Você sabe o que é uma matriz de covariância; no meu exemplo, é uma matriz 2×2 que é dada por

$$\begin{pmatrix} 1.07 & 0.63 \\ 0.63 & 0.64 \end{pmatrix}.$$

O que isso significa é que a variação da variável x é 1,07, a variação da variável is é 0,64 e a covariância entre elas é 0,63. Por ser uma matriz quadrada simétrica, pode ser diagonalizada escolhendo um novo sistema de coordenadas ortogonais, dado por seus vetores próprios (aliás, isso é chamado de teorema espectral); os autovalores correspondentes serão localizados na diagonal. Nesse novo sistema de coordenadas, a matriz de covariância é diagonal e tem a seguinte aparência:

PCA



$$\begin{pmatrix} 1.52 & 0 \\ 0 & 0.19 \end{pmatrix},$$

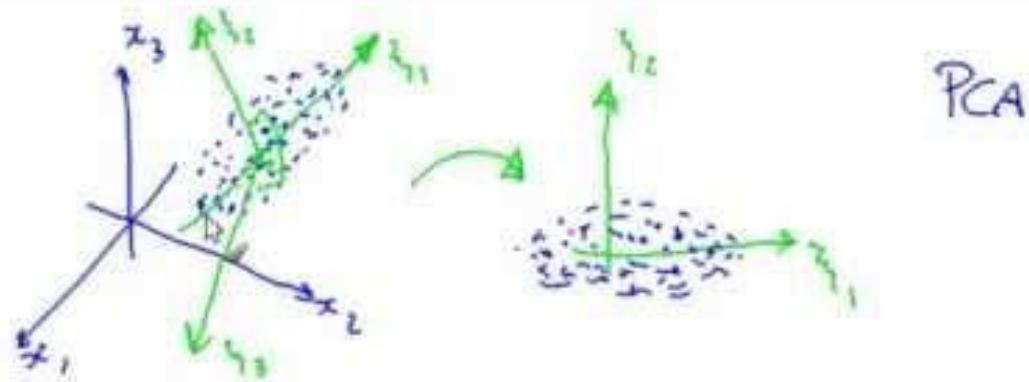
significando que a correlação entre pontos agora é zero. Torna-se claro que a variação de qualquer projeção será dada por uma média ponderada dos valores próprios (estou apenas esboçando a intuição aqui). Conseqüentemente, a variação máxima possível (1,52) será alcançada se simplesmente fizermos a projeção no primeiro eixo de coordenadas. Segue-se que a direção do primeiro componente principal é dada pelo primeiro vetor próprio da matriz de covariância.

Você também pode ver isso na figura rotativa: há uma linha cinza ortogonal à preta; juntos, eles formam um quadro de coordenadas rotativo. Tente perceber quando os pontos azuis não se correlacionam nesse quadro rotativo. A resposta, novamente, é que isso acontece exatamente quando a linha preta aponta para a magenta. Agora posso dizer como os encontrei: eles marcam a direção do primeiro vetor próprio da matriz de covariância, que neste caso é igual a (0,81,0,58).

ANÁLISE DOS COMPONENTES PRINCIPAIS - PCA (PRINCIPAL COMPONENTS ANALYSIS)

As desvantagens são:

- A sensibilidade a outliers,
- Não recomendada quando se tem duplas ausências (muitos zeros na matriz) e dados ausentes,
- Quando se tem mais variáveis do que unidades amostrais, ao reduzir o número de variáveis, há perda da informação de variabilidade das variáveis originais;
- PCA nem sempre funciona (às vezes, mesmo com a redução, ainda continua grande). É o caso de variáveis originais pouco correlacionadas!



PCA

$$\left\{ \begin{array}{l} \hat{y}_{11} = r_{11}x_1 + r_{12}x_2 + \dots + r_{1n}x_n \\ \hat{y}_{12} = r_{21}x_1 + r_{22}x_2 + \dots + r_{2n}x_n \\ \hat{y}_{13} = r_{31}x_1 + r_{32}x_2 + \dots + r_{3n}x_n \\ \vdots \\ \hat{y}_m = r_{m1}x_1 + r_{m2}x_2 + \dots + r_{mn}x_n \end{array} \right.$$

E continuação...

PCA

```
import numpy as np
from pandas import DataFrame

x = np.array([2.5, 0.5, 2.2, 1.9, 3.1, 2.3, 2.0, 1.0, 1.5, 1.1])
y = np.array([2.4, 0.7, 2.9, 2.2, 3.0, 2.7, 1.6, 1.1, 1.6, 0.9])

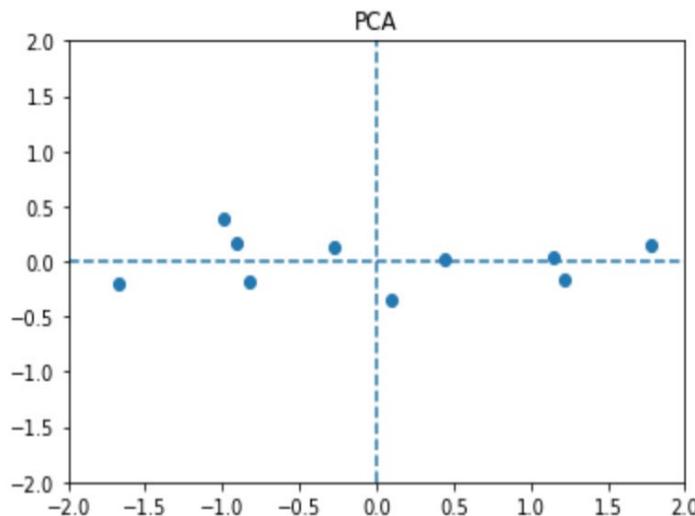
Z = np.c_[x - x.mean(), y - y.mean()]
df = DataFrame(Z, columns=['x', 'y'])
df.index.name = 'Medidas'
df.T
```

Medidas	0	1	2	3	4	5	6	7	8	9
x	0.69	-1.31	0.39	0.09	1.29	0.49	0.19	-0.81	-0.31	-0.71
y	0.49	-1.21	0.99	0.29	1.09	0.79	-0.31	-0.81	-0.31	-1.01

```
from sklearn.decomposition import PCA

pca = PCA(n_components = 2, copy = True) #executando o PCA
X = pca.fit_transform(Z)

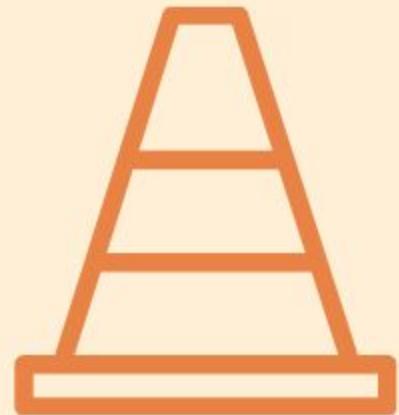
fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
ax.set_xlim([-2,2])
ax.set_ylim([-2,2])
ax.axhline(linestyle='dashed') #linha no eixo x
ax.axvline(linestyle='dashed') #linha no eixo y
_ = ax.set_title("PCA")
```



PCA

**PARA ESCOLHER AS
COMPONENTES, AS MAIS
IMPORTANTES SAO AS QUE
POSSUEM MAIOR VARIÂNCIA!!**

**O NÚMERO DE COMPONENTES
É DEFINIDO PELO USUÁRIO!!**

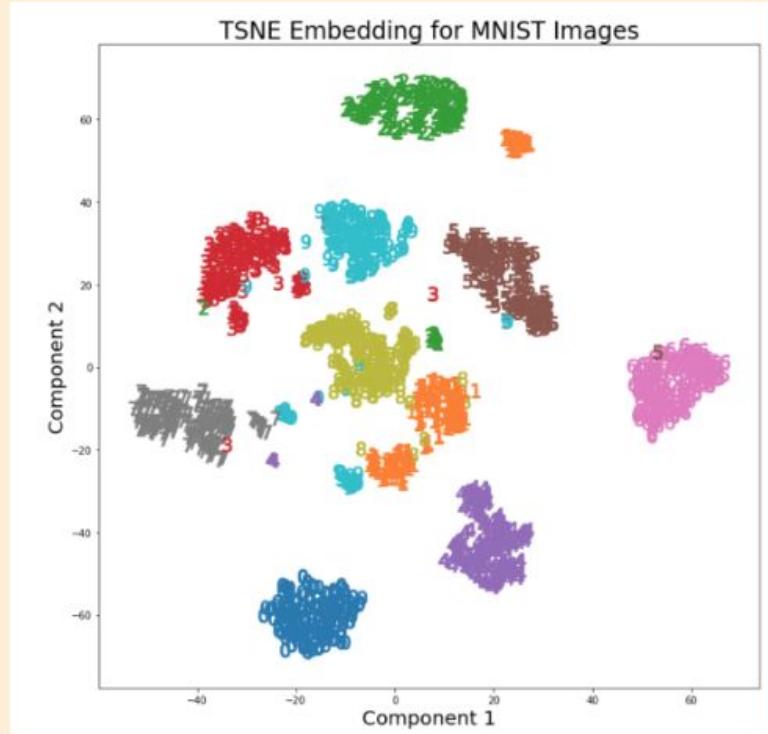


T-SNE

T-distributed Stochastic Neighbor Embedding (t-SNE) é um algoritmo de aprendizagem de máquina para visualização;

Desenvolvido por Laurens van der Maaten e Geoffrey Hinton;

É uma técnica de redução de dimensionalidade não linear para dados de altas dimensões para visualização em um espaço de poucas dimensões (duas ou três dimensões).



Identifica atributos importantes

Reduz a necessidade de memória e tempo de processamento
e os custos de coleta de dados e com isso aumenta o acesso a novas tecnologias

Facilita a visualização dos dados

Melhora o desempenho de várias técnicas de AM

Elimina atributos irrelevantes e reduz ruído

Simplifica o modelo gerado e torna mais fácil sua compreensão

Seleção de Atributos

TÉCNICAS DE SELEÇÃO DE SUBCONJUNTO

- A seleção de um subconjunto é um processo computacionalmente mais custoso que a ordenação dos atributos.
- Primeiro ordena-se os dados e em seguida seleciona-se o subconjunto.

TÉCNICAS DE SELEÇÃO DE SUBCONJUNTO

Geração para trás (backward generation): começa com todos os atributos e remove um a um;

Geração para a frente (forward generation): começa sem nenhum e inclui um atributo por vez;

Geração bidirecional (bidirectional generation): a busca pode começar em qualquer ponto e atributos podem ser adicionados ou removidos.

TÉCNICAS DE SELEÇÃO DE SUBCONJUNTO

Geração estocástica (random generation):
o ponto de partida da busca e atributos a
serem removidos ou adicionados são
decididos de forma estocástica.