Aula 02- Conjunto de dados

20443 – Tópicos em Informática X
Mineração de Dados
2018/2 - Turma A
Prof. Dr. Murilo Naldi

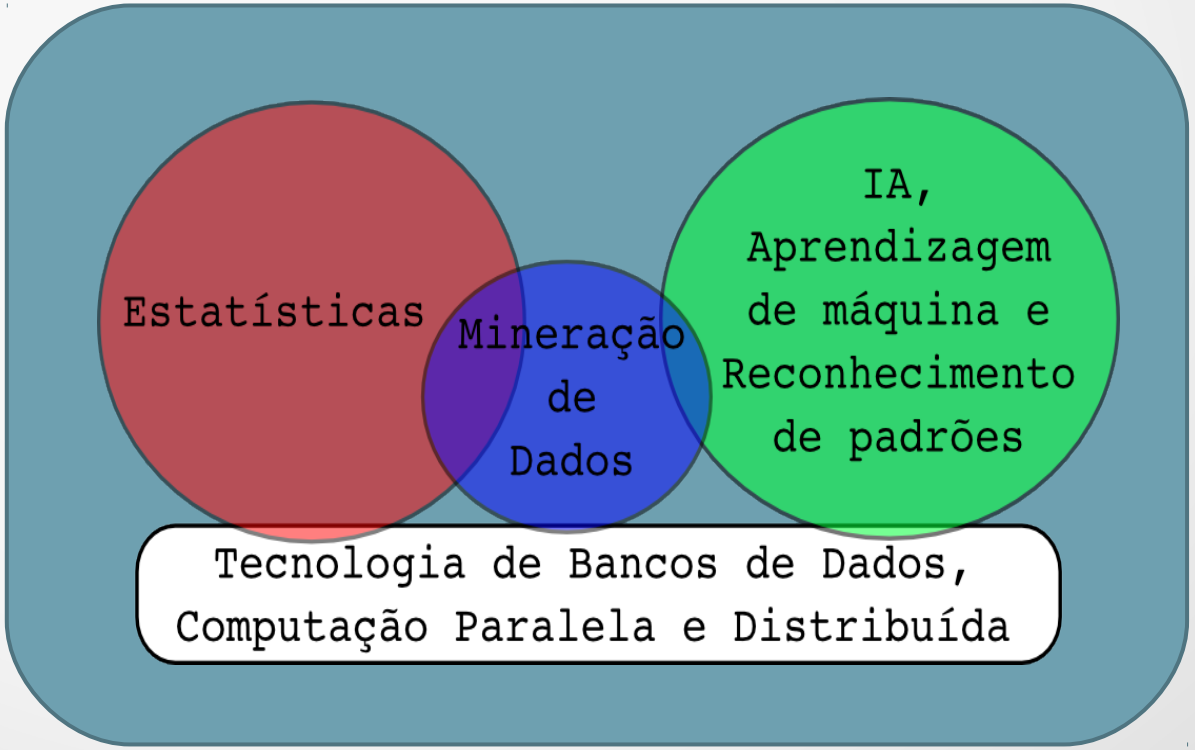
naldi@dc.ufscar.br

Agradecimentos

- Parte do material utilizado nesta aula foi cedido pelos professores André C.P.L.F de Carvalho e Ricardo J.G.B. Campello e, por esse motivo, o crédito deste material é deles
- Parte do material utilizado nesta aula foi disponibilizado por M. Kumar no endereço:
  - www-users.cs.umn.edu/~kumar/dmbook/index.php
- Agradecimentos a Intel Software e a Intel IA Academy pelo material disponibilizado e recursos didáticos

Aula Anterior

- Visão geral do processo de descoberta do conhecimento e mineração de dados
- Importância de cada etapa
- Origens da mineração de dados
- Aplicações principais



Nesta aula

- Conjunto de dados
  - Tipos de dados
  - Exemplos de conjuntos
  - Problemas com dados
- Explorando os dados
  - Estatísticas resumidas
  - Visualização

Conjunto de dados

- Dados são a essência da área de mineração de dados
- Gerados por diversas áreas do conhecimento, por exemplo:
  - Negócios
  - Genética
  - Economia
  - Medicina
  - Internet

Ilustração de problema de MD

- Em um projeto de mineração participam uma médica, uma estatística e um profissional de mineração de dados
- A médica envia dados sobre pacientes, sem tempo de fazer a mínima explanação
- O profissional abre o conjunto de dados se tem uma surpresa:

012	232	33,5	0	10,7
020	121	16,9	2	210,1
027	165	24,0	0	427,6
...	...	...	...	...

Ilustração de problema de MD

- O profissional aplica as técnicas de mineração diretamente nos dados e não observa nenhum resultado estranho ao prever o último atributo
- Dias depois, ele leva os resultados até uma reunião da equipe do projeto

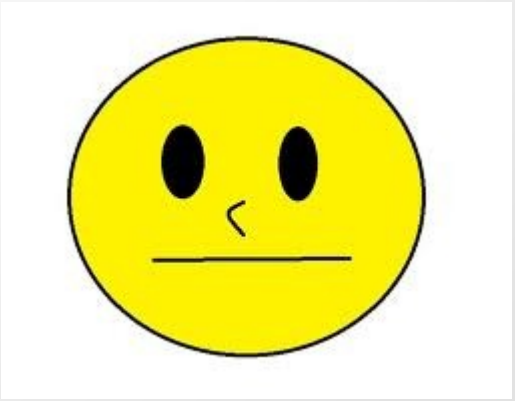
012	232	33,5	0	10,7
020	121	16,9	2	210,1
027	165	24,0	0	427,6
...	...	...	...	...

Ilustração de problema de MD

- No reunião, conversando com a estatística ele descobre que:
  - O resultado é melhor quando o campo 5 é normalizado
  - O campo 4 deveria ter valores de 1 a 10 e 0 quando não tinha valor presente. Por um erro, utilizaram 0 para definir 10
  - Os campos 2 e 3 eram praticamente os mesmos
  - O campo 1 é o ID e foi ordenado de acordo com o campo 5, o que gera a falsa relação encontrada pelo profissional de MD

Ilustração de problema de MD

- Em resumo, os resultados obtidos sobre tais dados não possuíam significado genuíno
- Portanto, é preciso compreender os dados antes de começar a minerá-los!



Tipos de atributos

- Atributos qualitativos ou categóricos:
- Nominais: representam características que não possuem ordem intrínseca.
  - Exemplos: cor dos olhos, profissão.
- Ordinais: representam qualidades/propriedades que refletem uma sequência ou ordem.
  - Exemplos: quente e frio, ou ruim, regular e bom.

Tipos de atributos

- Atributos quantitativos ou numéricos:
- Contínuos: número infinito e não enumerável de valores.
  - Exemplo: altura, peso.
- Discretos: número finito ou infinito porém enumerável de valores.
  - Exemplo: número de rodas, número de pessoas, números ímpares.

Exercício

- Quais são os tipos de atributos dos dados abaixo?

ID	Proprietário	E. Civil	Renda	Investe
1	Sim	Solteiro	1.500,00	Pouco
2	Não	Casado	812,00	Muito
3	Não	Solteiro	2.345,67	Não
4	Sim	Casado	4.768,00	Muito
5	Não	Divorciado	734,00	Não
6	Não	Casado	3.900,00	Pouco
7	Sim	Divorciado	2.100,00	Muito

Tipos de dados

- Os três tipos de dados mais comuns em tarefas de mineração de dados são:
  - Dados de registro
  - Dados baseados em grafos
  - Dados ordenados

Dados de registro

- Composto de uma coleção de registros, organizados verticalmente
- Exemplos:
  - dados de registro
  - matriz de dados
  - dados de transação
  - matriz de documentos

Exemplo de registros

ID	Proprietário	E. Civil	Renda	Investe
1	Sim	Solteiro	1.500,00	Pouco
2	Não	Casado	812,00	Muito
3	Não	Solteiro	2.345,67	Não
4	Sim	Casado	4.768,00	Muito
5	Não	Divorciado	734,00	Não
6	Não	Casado	3.900,00	Pouco
7	Sim	Divorciado	2.100,00	Muito

Matriz de documentos

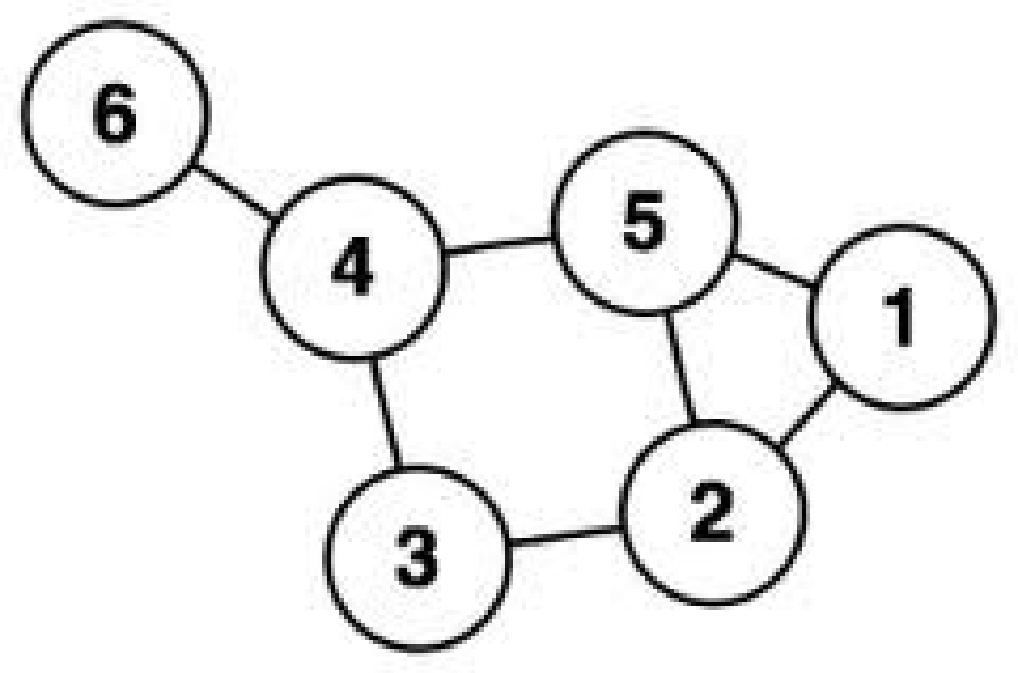
- Uma das formas mais tradicionais de tratar documentos de texto é utilizando o método *bag-of-words*
  - Consiste em “contar” o número de aparições de cada palavra em cada texto
    - *term-frequency*
  - Existem diversas variações da medida

Matriz de documentos

Documento	Abacate	Avião	Beterraba	Casa	Dados
1	213	0	35	0	0
2	18	0	123	0	0
3	0	0	0	0	0
4	0	7	0	3	7
5	0	2	0	5	15
6	14	0	0	17	0
7	0	0	12	0	0

Dados baseados em grafos

- Grafo pode ser uma representação poderosa para dados
- O grafo é capaz de:
  - capturar o relacionamento entre os objetos
  - contém os dados dentro do próprio grafo



Conjunto de dados Iris

- Consiste de informações sobre 150 flores Iris, 50 de 3 espécies, com 5 atributos:
- Comprimento de sépala
- Largura de sépala
- Comprimento de pétala
- Largura de pétala
- 3 Classes, 50 objetos cada
  - (Setosa, Multicolor, Virgínica)

Obtido de flowerinfo.org

Usando Pandas para ler Iris

- Código

```
# Localização do arquivo

filepath = 'data/Iris_Data.csv'

# Importando os dados
data = pd.read_csv(filepath)
# Imprimindo as cinco primeras linhas
print(data.iloc[:5])
```

- Saída

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Usando Pandas obter novos dados

- Código

```
# Calculando a área da sépala
data['sepal_area'] = data.sepal_length *
data.sepal_width

# Imprimindo algumas linhas e colunas
print(data.iloc[:5, -3:])
```

- Saída

	petal_width	species	sepal_area
0	0.2	Iris-setosa	17.85
1	0.2	Iris-setosa	14.70
2	0.2	Iris-setosa	15.04
3	0.2	Iris-setosa	14.26
4	0.2	Iris-setosa	18.00

Levantando estatísticas

- Código

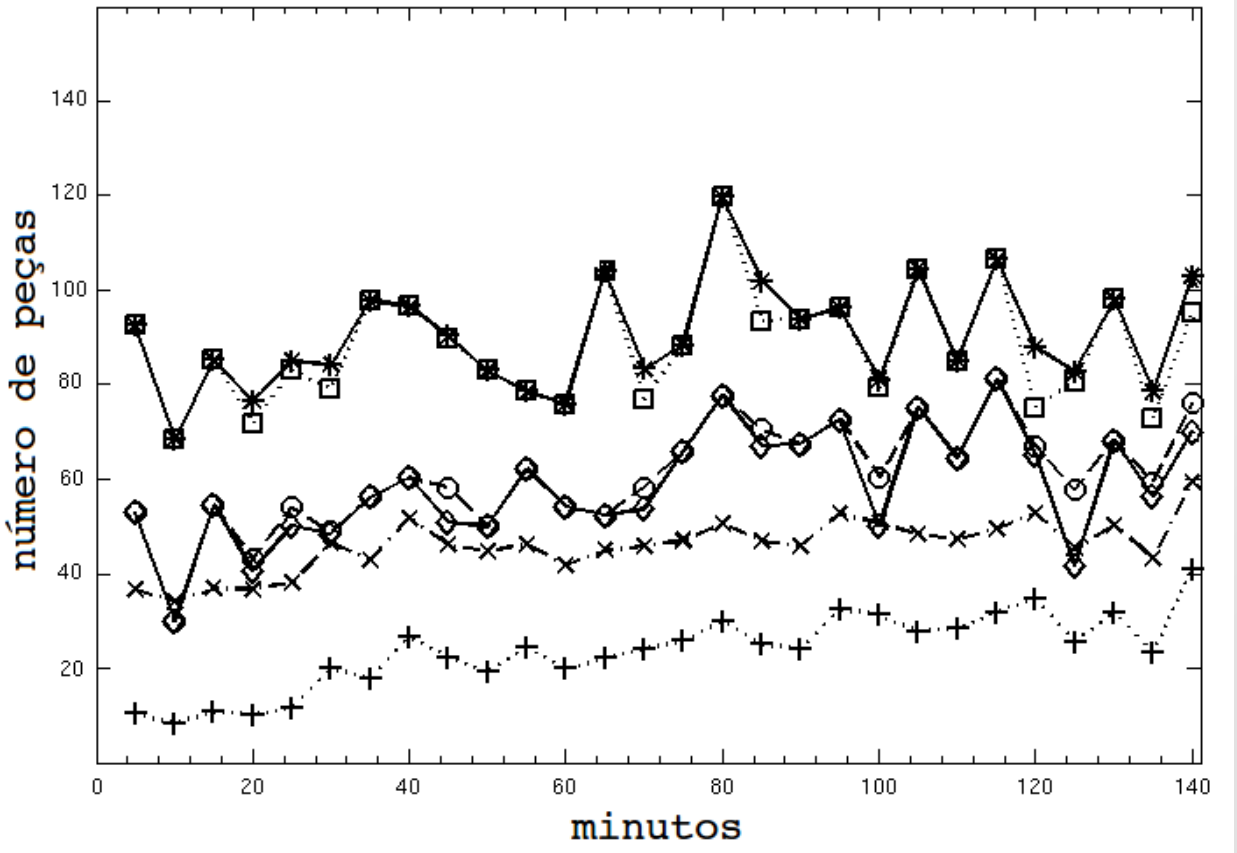
```
# Usando o método size com DataFrame
# Para séries, usar o método .value_counts
group_sizes = (data.groupby('species').size())
print(group_sizes)
```

- Saída

```
species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

Dados ordenados

- Os atributos têm relacionamentos que envolvem ordenação de tempo e espaço
- Exemplos:
  - Dados sequenciais, séries temporais



Qualidade dos dados

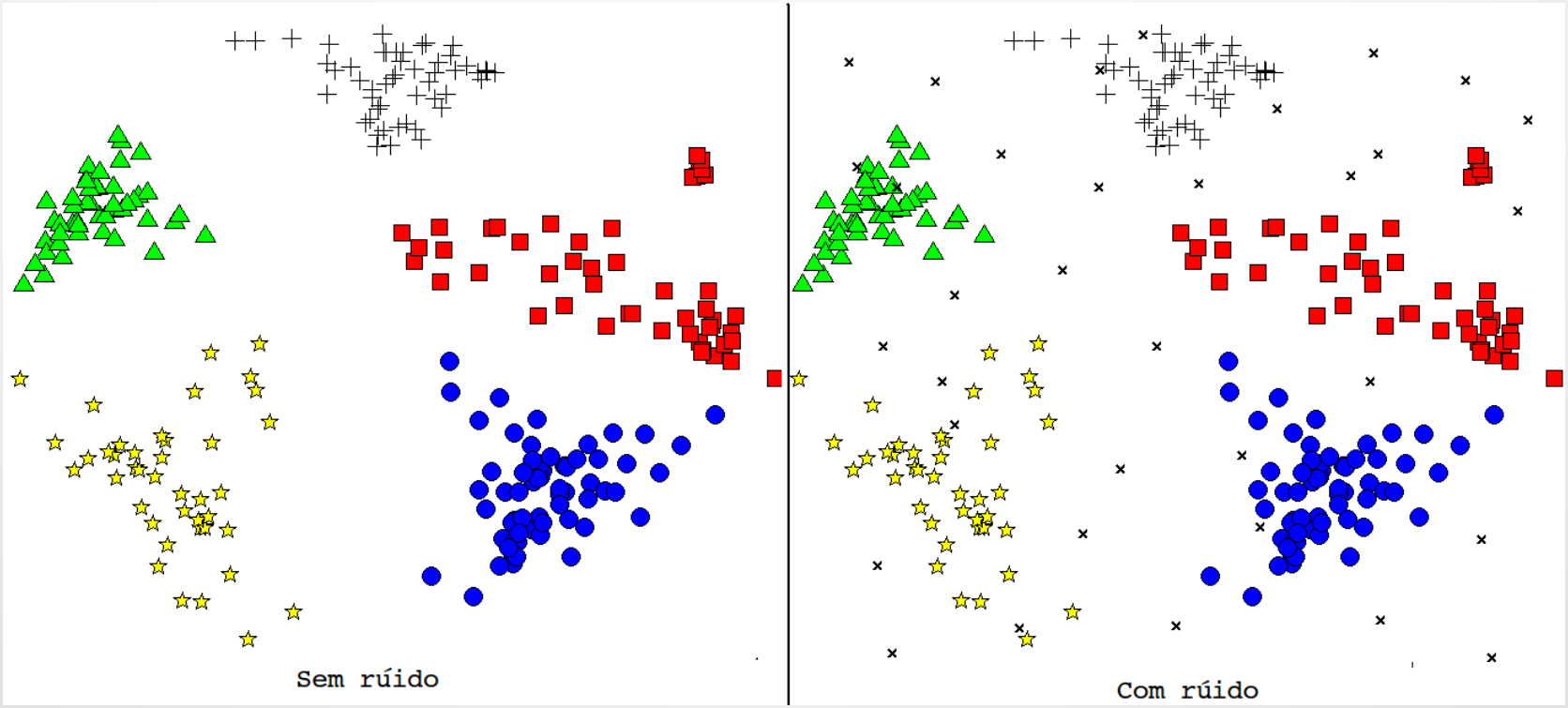
- A maioria dos dados são coletados para um propósito diferente de mineração
- Nem sempre é possível evitar problemas com a qualidade dos dados
- Duas possíveis alternativas:
  - Detecção e correção de problemas
  - Uso de algoritmos que toleram a baixa qualidade dos dados

Erros de medição e coleta

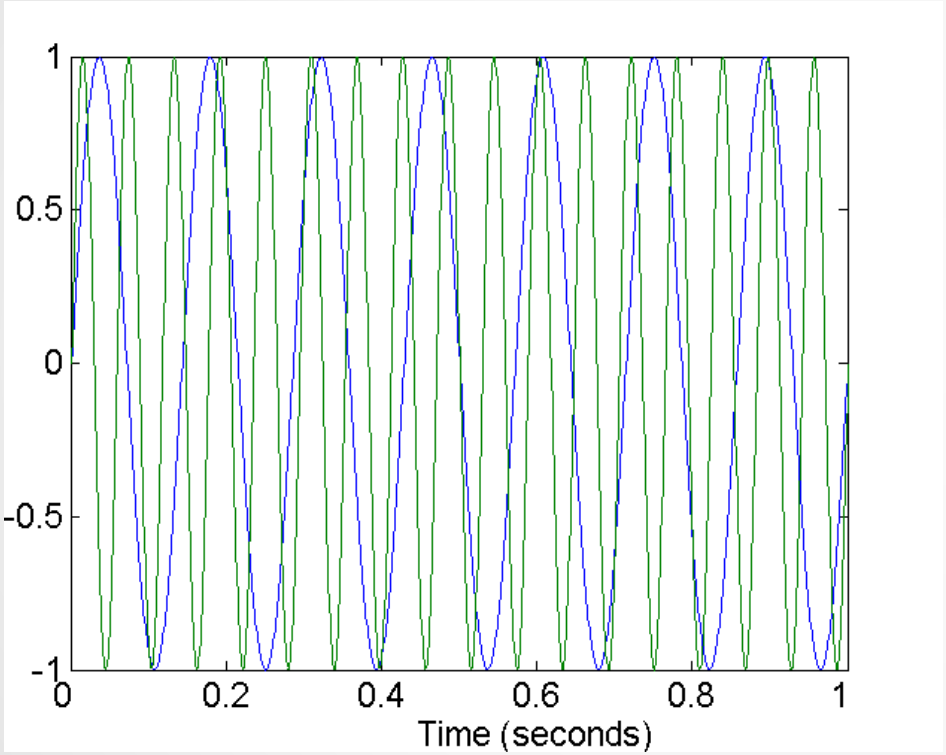
- Os mais comuns são:
  - Omissão de objetos ou valores de atributos
  - Inclusão inapropriada de objetos
  - Erros resultantes de medição
  - Erros de entrada
- Podem ser evitados com:
  - Programas que detectam e forçam a entrada de dados corretos
  - Várias medidas

Ruídos e artefatos

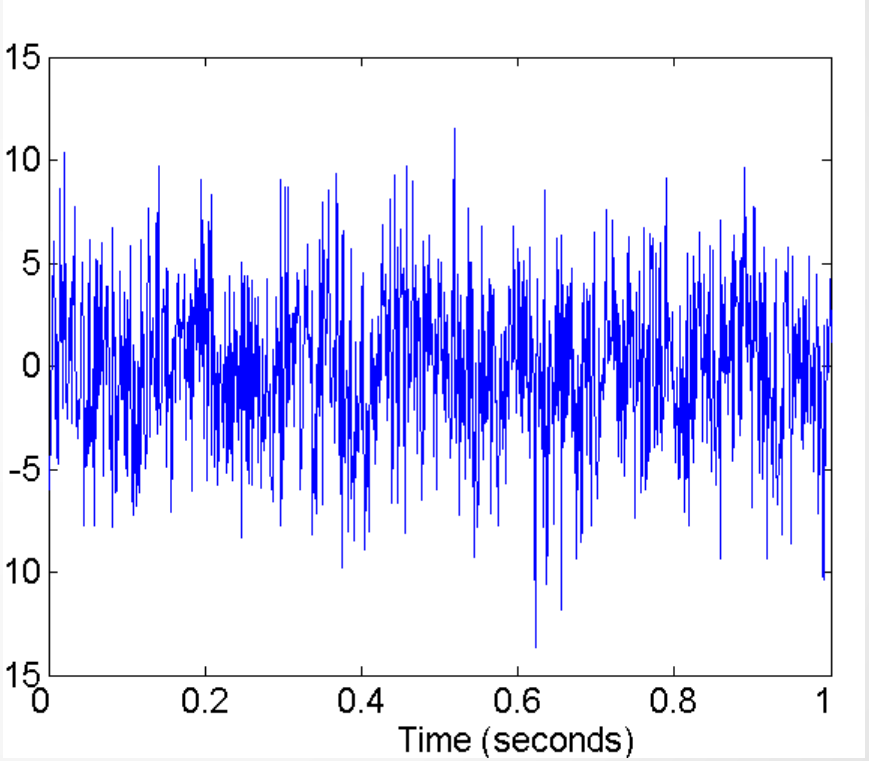
- Ruído é componente aleatório de um erro de medição
- Artefato são distorções determinísticas



Ruídos e artefatos



Duas senóides



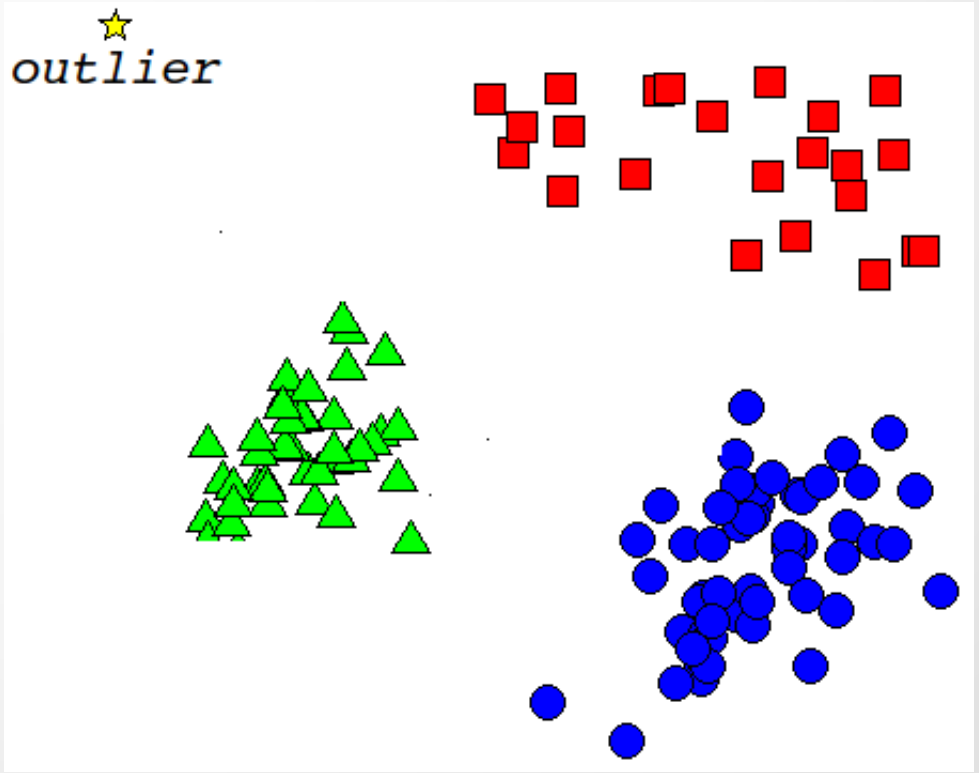
Duas senóides com ruído

Ruídos e artefatos

Exemplo de fotografias com artefato
Fotos por Antônio Bartocci Queiroz

Externos

- Externos (*outliers*) são objetos que possuem características diferentes da maioria dos outros objetos do conjunto
- Também é usado
  - para valores que sejam incomuns a um determinado atributo



Externos

- Externo não é (causado por) um erro.
  - Assume-se que ele é um objeto legítimo
- Externos podem ser importantes, pois podem indicar:
  - Comportamentos anômalos
  - Intrusão
  - Excessões

Valores faltantes

- E comum encontrar valores faltantes em conjuntos de dados
  - Especialmente em procedimentos caros
- Preciso lidar com esses objetos
  - Muitas técnicas não são capazes de trabalhar com valores faltantes
- O que fazer?

Valores faltantes

- Dentre as possíveis soluções, podemos:
  - Eliminar objetos ou atributos
    - Se o conjunto for grande o suficiente
  - Estimar os valores faltantes
    - Técnicas de *inputação*
  - Ignorar valores faltantes (caso especial)
    - Nem sempre é possível, devido a restrições da técnica aplicada

Valores inconsistentes

- Quando um valor não é consistente com o atributo em questão
- Devem ser detectados e eliminados
- Exemplos:
  - Idade: masculino ou 234 anos
  - Altura: -12
  - Cor: depende

Dados duplicados

- É possível que haja objetos semelhantes ou idênticos no mesmo conjunto de dados
- Neste caso, é preciso que tais objetos sejam combinados
- Objetos duplicados (multiplicados) podem resultar na redução de eficiência do algoritmo de mineração

Padronização

- Consiste em uma transformação aplicada nos valores dos atributos
- Normalização
  - Transformar em um vetor normado, ou seja, em um vetor de comprimento unitário (1).
  - Projetar os dados em um novo intervalo
    - Exemplo: para padronizar um número *x* que está no intervalo [$menor_1$,$maior_1$] para o intervalo [$menor_2$,$maior_2$] usamos

$$n(x) = (x - menor_1) \times \frac{maior_2 - menor_2}{maior_1 - menor_1} + menor_2$$

Explorando dados

- Uma boa forma de conhecer os dados é explorando suas características
- A exploração dos dados consiste em uma investigação preliminar dos dados com o intuito de compreender melhor suas características
- Exemplo:
  - Estatísticas resumidas
  - Visualização

Estatísticas de Resumo

- São informações que capturam diversas características de uma grande quantidade de valores
- Exemplos:
  - Frequência e moda
  - Medidas de localização (média, mediana)
  - Porcentagem
  - Medidas de dispersão (variação e variância)

Levantando estatísticas

- Código

```
# Média sobre o DataFrame
print(data.mean())
# Média sobre série
print(data.petal_length.median())
# Moda calculada sobre série
print(data.petal_length.mode())
```

- Saída

```
sepal_length    5.843333
sepal_width     3.054000
petal_length    3.758667
petal_width     1.198667
sepal_area     17.806533
dtype: float64
```

- Saída

```
4.35

0    1.5
dtype: float64
```

Levantando estatísticas

- Código

```
# Desvio padrão, variância, e desvio padrão da média
print(data.petal_length.std(), data.petal_length.var(),
data.petal_length.sem())
# Retorna quartis
print(data.quantile(0))
```

- Saída

```
1.764420419952261 7 3.113179418344515 6 0.1440643240210084
sepal_length    4.3
sepal_width     2.0
petal_length    1.0
petal_width     0.1
sepal_area      10.0
Name: 0, dtype: float64
```

Descrevendo os dados

- Código

```
# Desvio padrão, variância, e desvio padrão da média
print(data.describe())
```

- Saída

	sepal_length	sepal_width	petal_length	petal_width	sepal_area
count	150.0000	150.0000	150.0000	150.0000	150.00000
mean	5.843333	3.054000	3.758667	1.198667	17.806533
std	0.828066	0.433594	1.764420	0.763161	3.368693
min	4.300000	2.000000	1.000000	0.100000	10.000000
25%	5.100000	2.800000	1.600000	0.300000	15.645000
50%	5.800000	3.000000	4.350000	1.300000	17.660000
75%	6.400000	3.300000	5.100000	1.800000	20.325000
max	7.900000	4.400000	6.900000	2.500000	30.020000

Variância

- A variância é preferida dentre as medidas de dispersão pois representa melhor valores que estão dentro da faixa (diferença entre o valor máximo e mínimo de um atributo).

$$s_x^2 = variância(x) = \frac{1}{a-1}\sum_{i=1}^{a}(x_i - \bar{x})^2$$

Múltiplas variáveis

- Estatísticas de resumo podem ser aplicadas em múltiplas variáveis
- Para variáveis contínuas, a dispersão dos dados é capturada pela matriz de co-variância
- A co-variância de dois atributos é a medida do grau no qual dois atributos variam juntos
  - Não padronizada
    - Caso contrário, possuiria esperança zero e variância 1.

Co-variância

- A co-variância dos atributos de índice i e j é dada por

$$s_{ij} = co - variância(x_i, x_j)$$

- E a co-variância é dada por

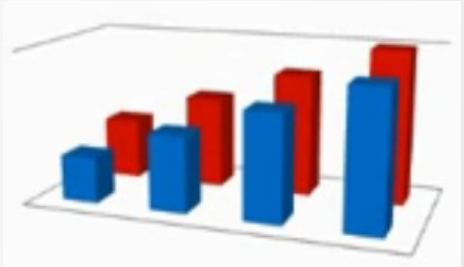
$$co - variância(x, y) = \frac{1}{a-1} \sum_{i=1}^{a} (x_i - \bar{x})(y_i - \bar{y})$$

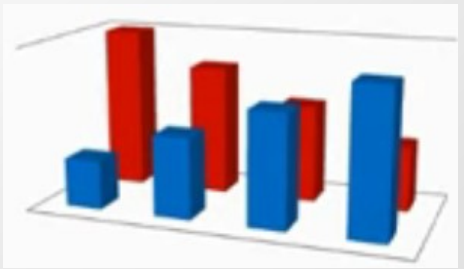
Propriedades da Co-variância

- Para quaisquer variáveis aleatórias X, Y, Z e uma constante c, temos:
  - co-variância(X,X) = variância(X)
  - co-variância(X,Y) = co-variância(Y,X)
  - co-variância(cX,Y) = c . co-variância(X,Y)
  - co-variância(X, Y+Z) = co-variância(X,Y) + co-variância(X,Z)

Propriedades da Co-variância

- Se X e Y são variáveis aleatórias independentes, então:
  - *co-variância*(X,Y) = 0
  - Porém *co-variância*(X,Y) = 0 não implica em independência!
- Sinal positivo indica que X e Y se desenvolvem juntas
  - Negativo implica no contrário





Obtendo a covariância Iris

- Código

```
# Análise da covariância entre os atributos Iris
print(data.cov())
```

- Saída

	sepal_length	sepal_width	petal_length	petal_width	sepal_area
sepal_length	0.685694	-0.039268	1.273682	0.516904	1.906238
sepal_width	-0.039268	0.188004	-0.321713	-0.117981	0.942732
petal_length	1.273682	-0.321713	3.113179	1.296387	2.178896
petal_width	0.516904	-0.117981	1.296387	0.582414	0.965009
sepal_area	1.906238	0.942732	2.178896	0.965009	11.348090

Covariância e Correlação

- Não é possível julgar o grau de relacionamento entre dois atributos apenas pela covariância
  - Pois ela não é padronizada
  - Qual valor é suficientemente alto?

- Nestes casos a correlação é preferida

Correlação

- Não é possível julgar o grau de relacionamento entre dois atributos apenas pela covariância
  - Pois ela não é padronizada
- Nestes casos a correlação é preferida, dada por:

$$correlacão(s_i, x_j) = \frac{co - variância(s_i, x_j)}{s_i \cdot x_j}$$

Obtendo a correlação Iris

- Código

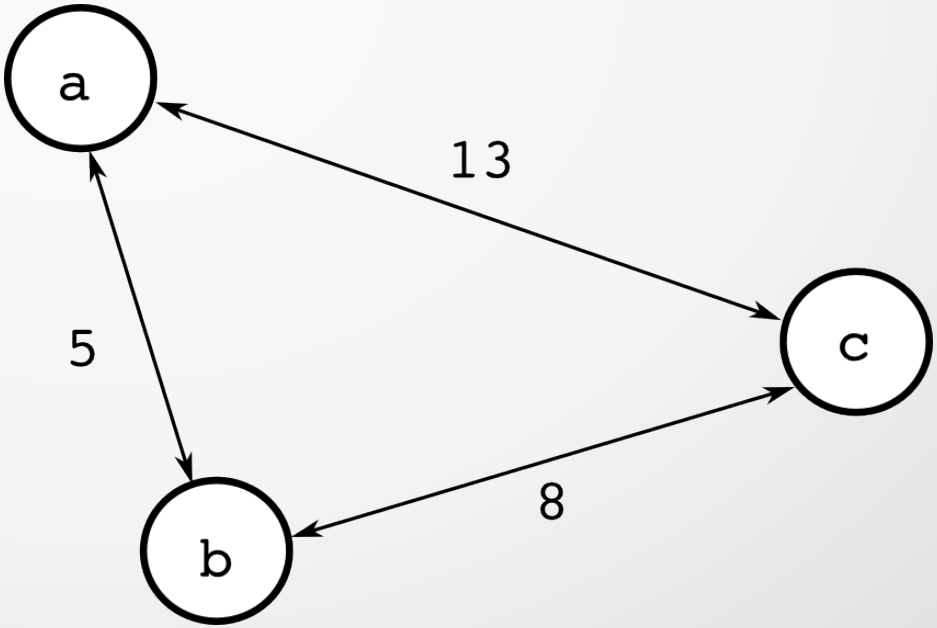
```
# Análise da correlação entre os atributos Iris
print(data.corr())
```

- Saída

	sepal_length	sepal_width	petal_length	petal_width	sepal_area
sepal_length	1.000000	-0.109369	0.871754	0.817954	0.683362
sepal_width	-0.109369	1.000000	-0.420516	-0.356544	0.645421
petal_length	0.871754	-0.420516	1.000000	0.962757	0.366584
petal_width	0.817954	-0.356544	0.962757	1.000000	0.375365
sepal_area	0.683362	0.645421	0.366584	0.375365	1.000000

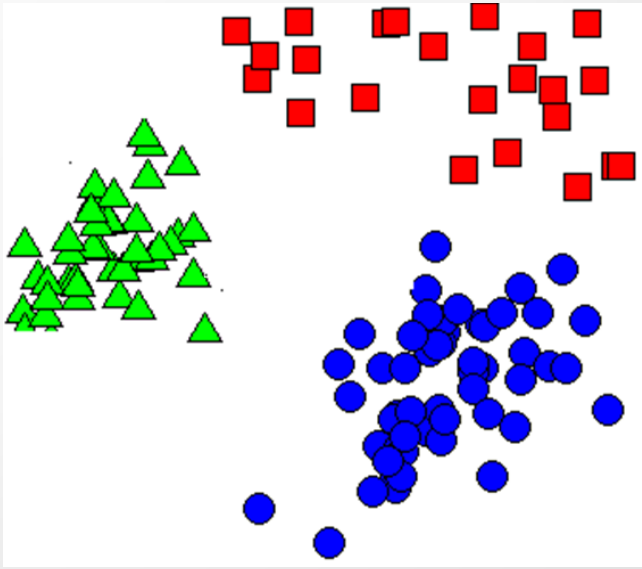
Medidas de similaridade

- Também conhecidas como medidas de semelhança ou proximidade
- Consiste em medidas utilizadas para mensurar quão similar um objeto do conjunto de dados é de outro
- Quando usado para medir diferenças, é chamado de medida de dissimilaridade



Medidas de similaridade

- Também utilizadas para calcular similares em geral (entre grupos, centroides, objetos vizinhos)
- Essenciais para
  - tarefas de mineração
- Exemplo:
  - Agrupamento
  - Vizinhos mais próximos
  - Detecção de anomalia



Medidas para atributos qualitativos

- Atributos binários
- Sendo $\mathbf{x}_i$ e $\mathbf{x}_j$ objetos, e $na^{\{ab\}}$ o número de atributos em $\mathbf{x}_i$ que sejam *a* e em $\mathbf{x}_j$ sejam *b*
- Podemos utilizar a medida de casamento simples, dada por:

$$s(\mathbf{x}_i, \mathbf{x}_j) = \frac{n_a^{\{11\}} + n_a^{\{00\}}}{n_a^{\{00\}} + n_a^{\{01\}} + n_a^{\{10\}} + n_a^{\{11\}}}$$

Medidas para atributos qualitativos

- Nominais ou ordinais
  - É preciso definir um índice de discordância para um atributo *z*

$$\delta_z(\mathbf{x}_i, \mathbf{x}_j)$$

- Poderia ser 1 se os valores forem iguais ou 0 se forem diferentes
- No caso de ordinais, é preciso estabelecer uma ordem entre os valores

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sum_{z=1}^{n_a} \delta_z(\mathbf{x}_i, \mathbf{x}_j)$$

Medidas para atributos quantitativos

- Métricas de Minkowski
- Baseadas na equação:

$$d(\mathbf{x}_i, \mathbf{x}_j) = (\sum_{z=1}^{n_a} w_z^{\rho} |\mathbf{x}_i(z) - \mathbf{x}_j(z)|^{\rho})^{1/\rho}$$

- Em que w_z é o peso do z-ésimo atributo e p é um parâmetro positivo
- Quando é $p = 1$, a medida é chamada distância de Manhattan ou de blocos
- Quando é $p = 2$, a medida é chamada distância Euclidiana

Medidas de correlação

- Medidas baseadas em informações sobre a magnitude relativa de diferentes atributos
  - A separação angular é calculada por meio do cosseno do ângulo formado entre dois vetores que representam os objetos

$$\phi(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{z=1}^{n_a} \mathbf{x}_i(z)\mathbf{x}_j(z)}{\sqrt{\sum_{z=1}^{n_a} \mathbf{x}_i(z)^2 \sum_{z=1}^{n_a} \mathbf{x}_j(z)^2}}$$

$$d(\mathbf{x}_i, \mathbf{x}_j) = \frac{1 - \phi(\mathbf{x}_i, \mathbf{x}_j)}{2}$$

Medidas de correlação

- Coeficiente de correlação de Pearson:
  - útil quando se deseja medir a tendência de duas sequências para as quais os valores de magnitude absolutos não são relevantes, apenas os valores relativos

$$\overline{\mathbf{x}}_i = \frac{1}{n_a} \sum_{z=1}^{n_a} \mathbf{x}_i(z)$$

$$\phi(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{z=1}^{n_a} (\mathbf{x}_i(z) - \overline{\mathbf{x}}_i)(\mathbf{x}_j(z) - \overline{\mathbf{x}}_j)}{\sqrt{\sum_{z=1}^{n_a} (\mathbf{x}_i(z) - \overline{\mathbf{x}}_i)^2 \sum_{z=1}^{n_a} (\mathbf{x}_j(z) - \overline{\mathbf{x}}_j)^2}}$$

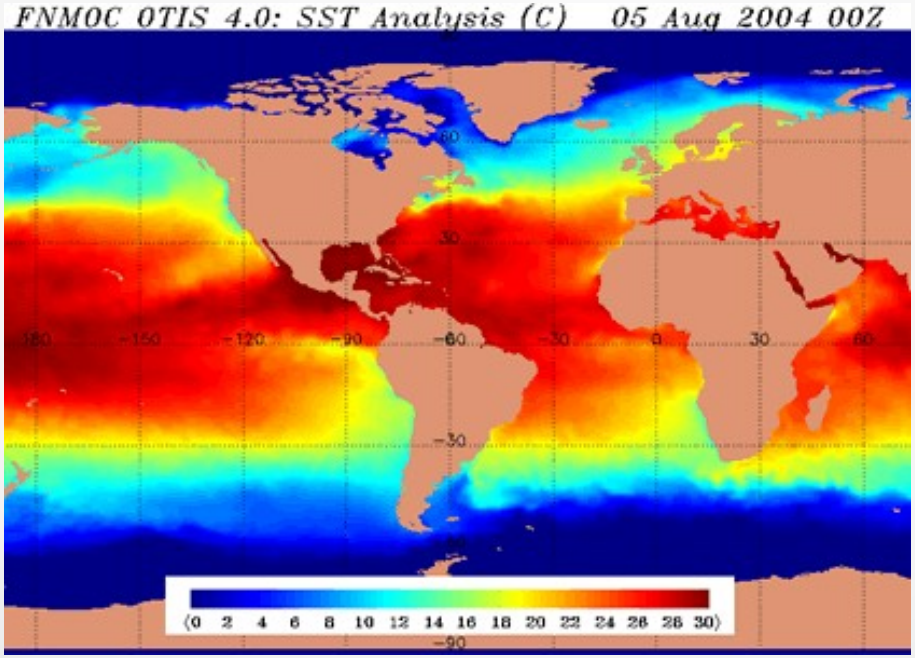
$$d(\mathbf{x}_i, \mathbf{x}_j) = \frac{1 - \phi(\mathbf{x}_i, \mathbf{x}_j)}{2}$$

Visualização

- Visualização é uma exibição dos dados me forma de gráfico ou tabela
- Para isso é preciso converter os dados em formas visuais, de forma que suas principais características sejam apresentadas
- A principal motivação da visualização é que o expectador possa absorver rapidamente grandes quantidades de informações

Exemplo

- A figura abaixo apresenta a temperatura do mar no mês de agosto de 2004
- Nela estão presentes milhares de valores



Fonte: www.zamg.ac.at

Mapeando dados

- Dados são geralmente representados de três formas:
  - Se apenas um atributo é categorizado por objeto, então os dados são agrupados segundo este atributo (tabela de tabulação cruzada e gráficos de barras)
  - Se os objetos são representados por mais de um atributo, ele pode ser exibido como linha em uma tabela ou gráfico
  - Pode ser apresentado graficamente como um ponto em um espaço bi ou tri-dimensional

Organização

- A forma em que os dados são organizados influenciam em sua interpretação

	1	2	3	4	5
1	0	1	0	1	1
2	1	0	1	0	0
3	0	1	0	1	1
4	1	0	1	0	0
5	0	1	0	1	1
6	1	0	1	0	0
7	0	1	0	1	1
8	1	0	1	0	0

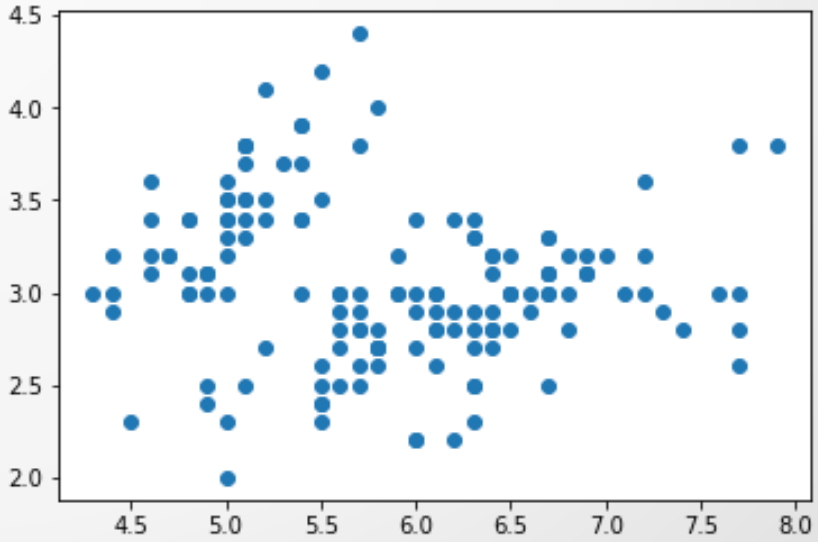
	1	3	2	5	4
4	1	1	0	0	0
2	1	1	0	0	0
6	1	1	0	0	0
8	1	1	0	0	0
5	0	0	1	1	1
3	0	0	1	1	1
1	0	0	1	1	1
7	0	0	1	1	1

Análise de atributos

- Dados que possuem 2 ou 3 atributos podem ser plotados em gráficos
- Quando possível, pode-se selecionar um subconjunto de atributos para análise

- Código

```
# Comparando comprimento com
largura da sépala
import matplotlib.pyplot as plt
plt.plot(data.sepal_length,
data.sepal_width,ls='', marker='o')
```

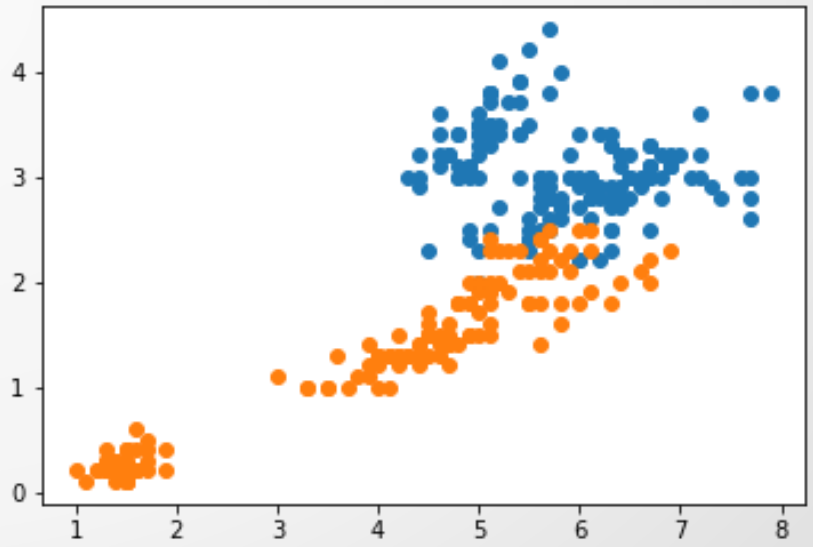


Análise de atributos

- Também é possível sobrepor comparações
  - Por exemplo, comparar pétala e sépala ao mesmo tempo por meio do seu tamanho

- Código

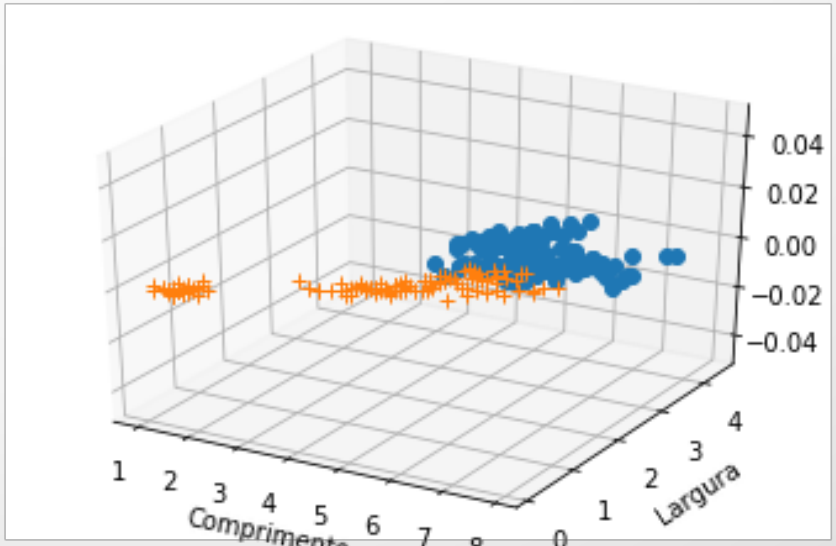
```
# Comparando comprimento com
largura da sépala e pétala
plt.plot(data.sepal_length,
data.sepal_width, ls ='', marker='o',
label='sepal')
plt.plot(data.petal_length,
data.petal_width, ls ='', marker='o',
label='petal')
```



Análise de atributos

- Código

```
# Comparando comprimento com largura da sépala e pétala
em três dimensões
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
plt.plot(data.sepal_length, data.sepal_width,ls ='',
marker='o', label='sepal')
plt.plot(data.petal_length, data.petal_width,ls ='',
marker='+', label='petal')
ax.set_xlabel('Comprimento')
ax.set_ylabel('Largura')
plt.show()
```

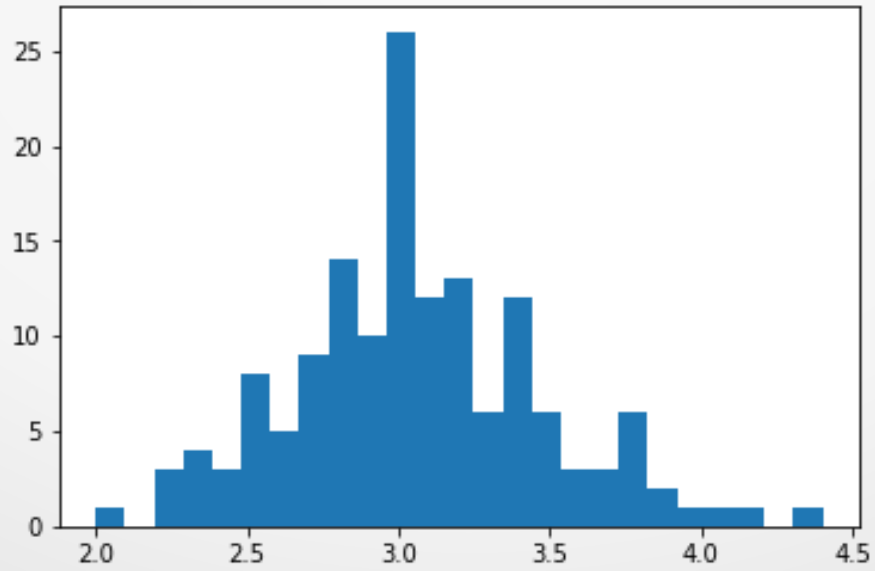


Histograma

- Contém a contagem de valores ou a frequência relativa
  - Divide os valores em cestos

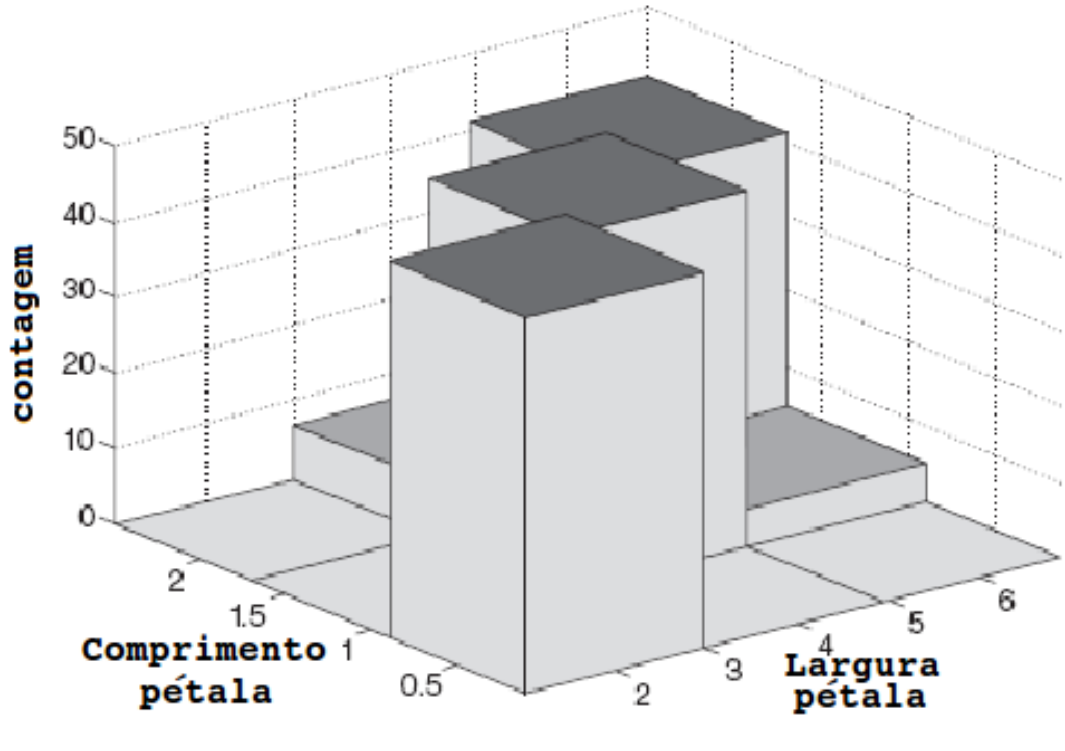
- Código

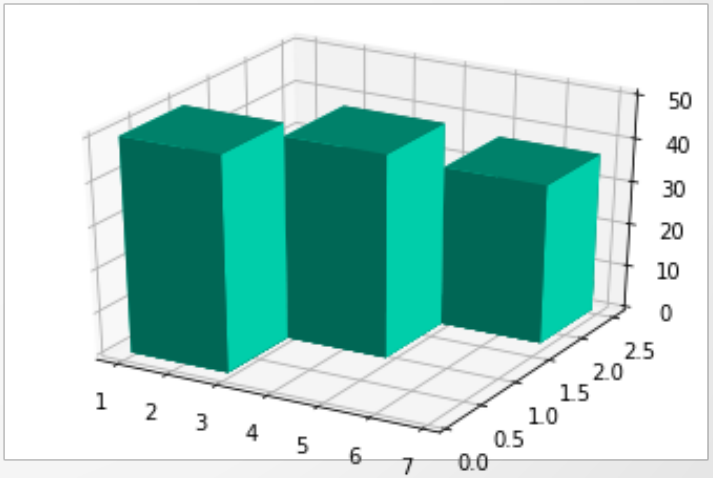
```
#  Histograma com largura de sépala
plt.hist(data.sepal_width, bins=25)
```



Histograma

- Histogramas podem ser utilizados para comparar múltiplos atributos ao mesmo tempo





Configurando gráfico

- Código

```
# Configurando barras e rótulos de 10 objetos
ig, ax = plt.subplots()
ax.barh(np.arange(10), data.sepal_length.iloc[:10])
ax.set(xlabel='Centímetros', ylabel='Número',
       title='Comprimento de Sépala')
```

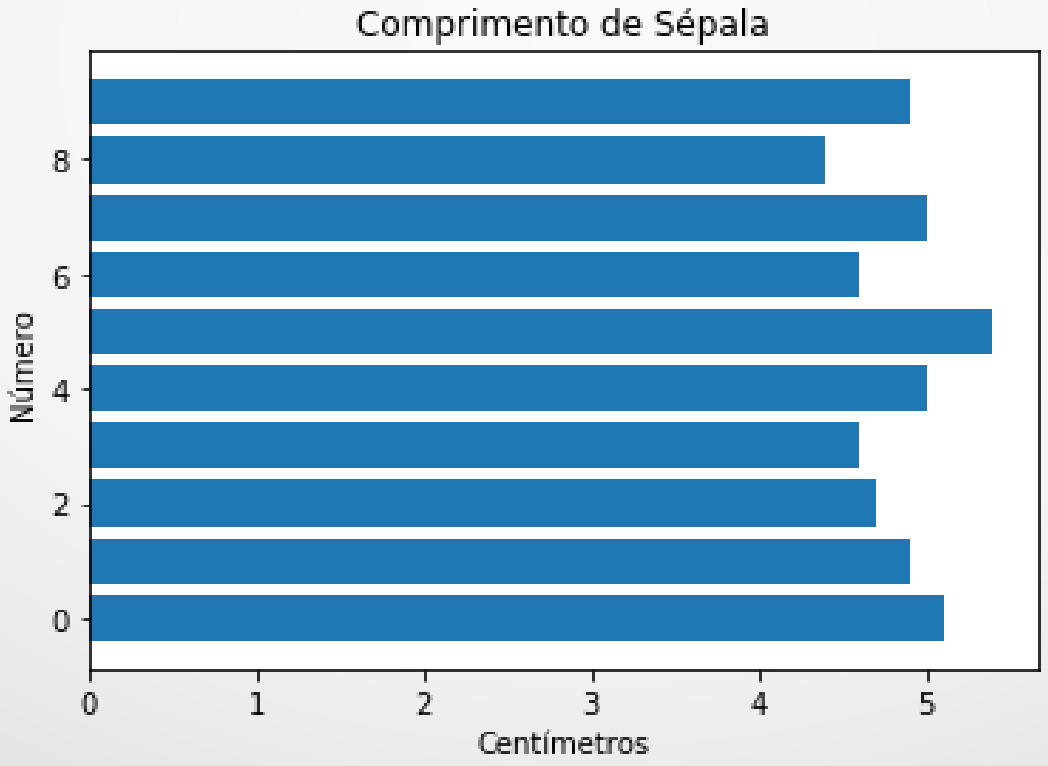


Gráfico de setores

- Usado para comparar valores relativos a um atributo
- Muito simples, pouco informativo
  - Menos preferido em trabalhos técnicos
- Código

```
#Gráfico de setores sobre Iris
import seaborn as sns
labels = data['species'].unique()
sizes = data.groupby('species').size()
colors = ['gold', 'lightcoral',
'lightskyblue']
explode = (0.1, 0, 0)
plt.pie(sizes, explode=explode,
labels=labels,
colors=colors,autopct='%1.1f%%',
shadow=True, startangle=140)
```

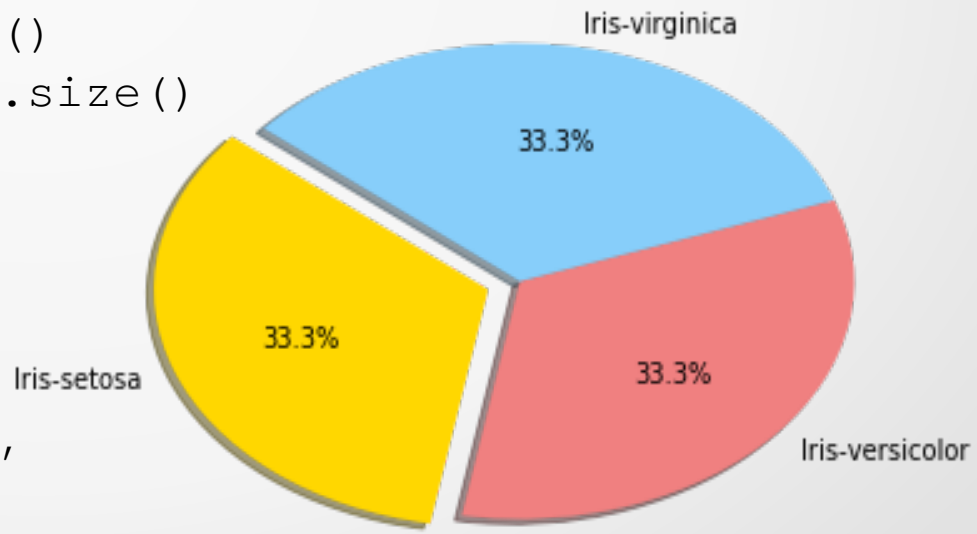


Diagrama caule e folhas

- Usados para fornecer informações sobre a distribuição dos dados contínuos ou inteiros unidimensionais
- São um tipo de histograma, em os atributos são divididos em intervalo de valores os números indicam os valores representados

Caule	Folha
3	7
4	2 8 9
5	3 5 7 8 9
6	0 2 2 3 4 5 6 8 9
7	0 1 2 3 4 5 5 6 7 7 8 8 9 9
8	0 0 1 3 4 4 5 6 7 8 9
9	0 0 2 3 5 8 9

Diagrama caule e folhas

- Comprimento de sépala em ordem crescente

43 44 44 44 45 46 46 46 46 47 47 48 48 48 48 48 49 49 49 49 49 49 50 50
50 50 50 50 50 50 50 50 51 51 51 51 51 51 51 51 51 52 52 52 52 53 54 54
54 54 54 54 55 55 55 55 55 55 55 56 56 56 56 56 56 57 57 57 57 57 57 57
57 58 58 58 58 58 58 58 59 59 59 60 60 60 60 60 60 61 61 61 61 61 61 62
62 62 62 63 63 63 63 63 63 63 63 64 64 64 64 64 64 64 65 65 65 65 65
66 66 67 67 67 67 67 67 67 67 68 68 68 69 69 69 69 70 71 72 72 72 73 74
76 77 77 77 77 79

- Gráfico caule e folha do comprimento de sépala

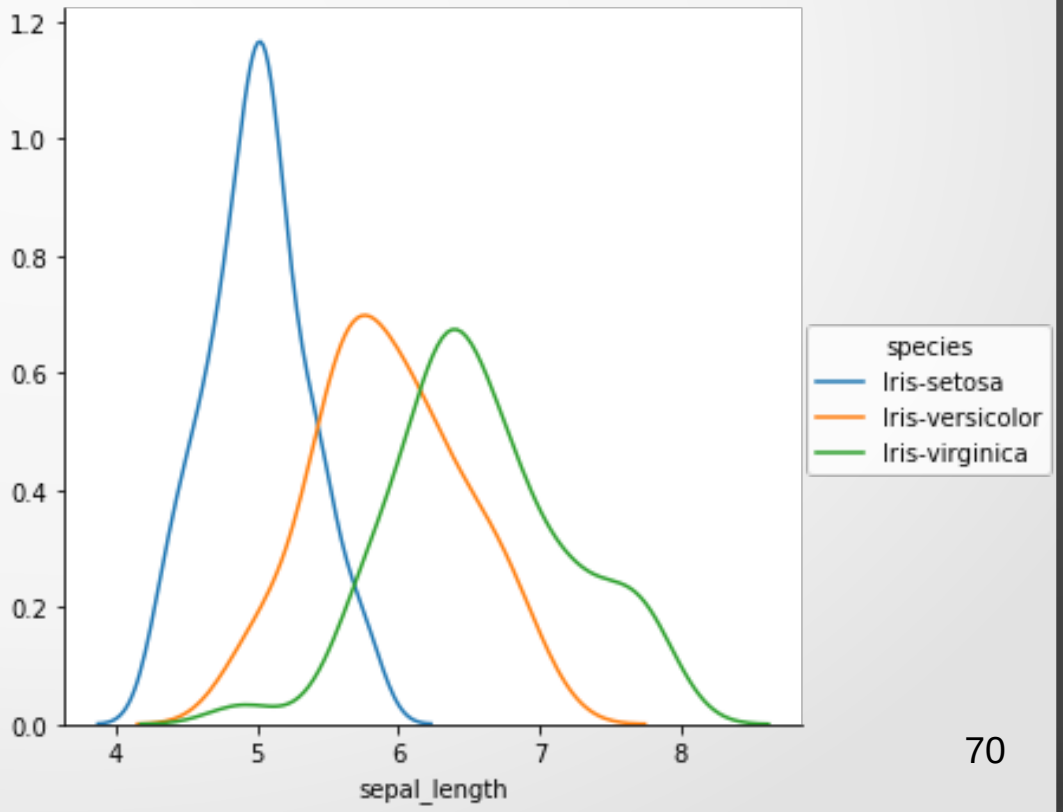
4 : 3444566667788888999999
5 : 000000000011111111222234444445555555666666777777788888899999
6 : 000000111111222233333333444444455555667777777888999999
7 : 0122234677779

Gráfico de densidade

- Utilizado para estudo das características dos dados, sua distribuição e relação
  - Semelhante aos histogramas, mas sem cestos e podem ser sobrepostos no mesmo gráfico

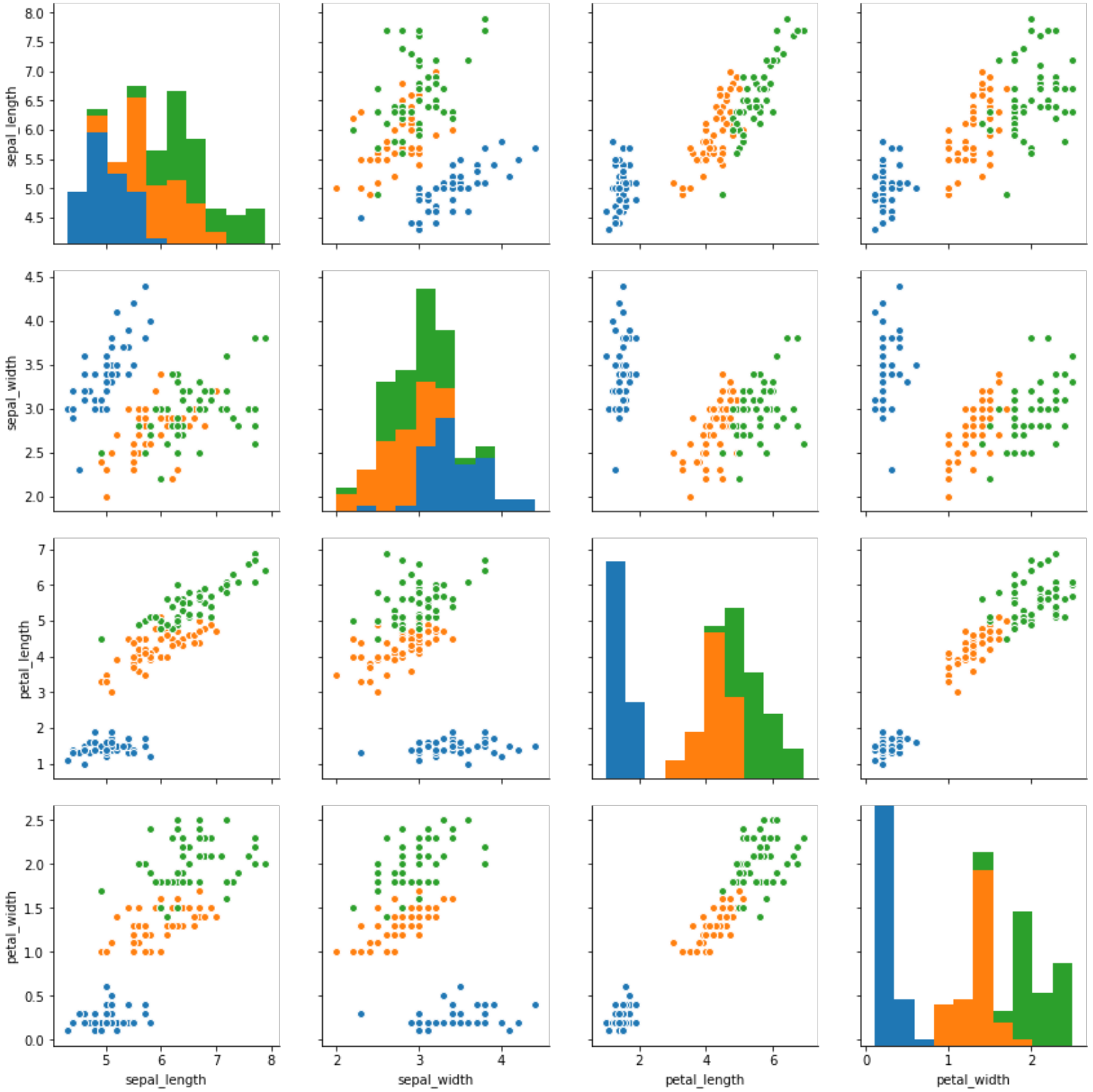
- Código

```
#Gráfico de densidade para o atributo comprimento de sépala
sns.FacetGrid(data,hue="species", size=5).map(sns.kdeplot, "sepal_length").add_legend()
```



Gráficos de dispersão

- Usados para duas finalidades principais:
  - Mostram graficamente o relacionamento entre dois atributos (avaliação do grau de correlação linear)
  - Quando as classes são conhecidas, podem ser utilizados para investigar o grau no qual os atributos separam as classes



#Gŕaficos de dispersão

sns.pairplot (data, hue= 'species', size=3)

Gráfico de caixas

- Gráficos de caixas são utilizados para permitir a visualização da distribuição dos valores de um único atributo numérico
- As extremidades indicam 10% e 90% dos dados, enquanto os limites indicam 25% e 75% dos dados

- Código

```
# Boxplot usando SeaBorn
sns.boxplot(x="species",
y="sepal_length", data=data)
```

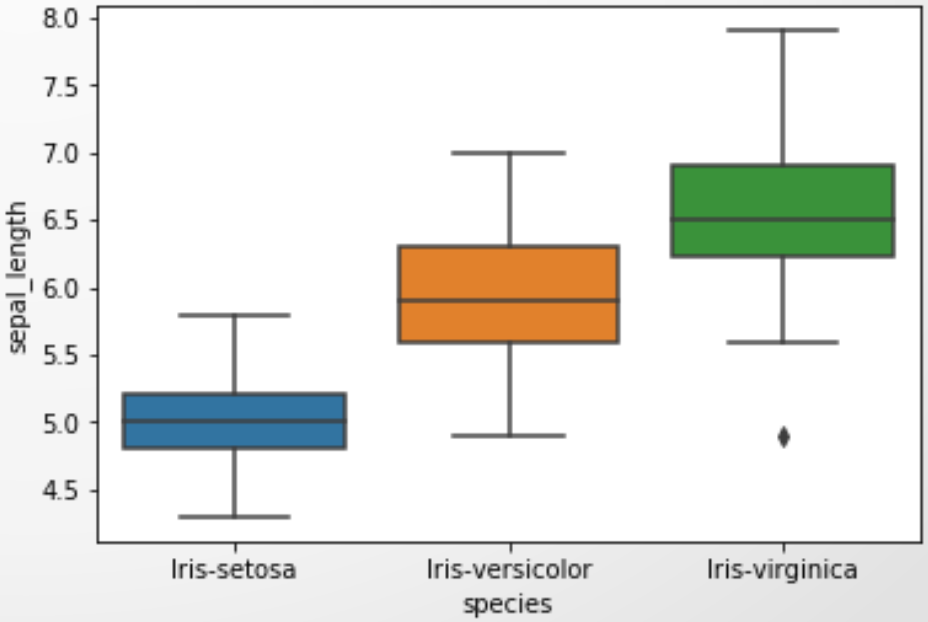


Gráfico de caixas

- É possível sobrepor gráficos usando SeaBorn
- Neste caso, foi feito um gráfico de dispersão sobre o gráfico de caixas

- Código

```
# Boxplot usando SeaBorn
sns.boxplot(x="species",
y="sepal_length", data=data)
ax = sns.stripplot(x="species",
y="sepal_length", data=data,
jitter=True, edgecolor="black")
```

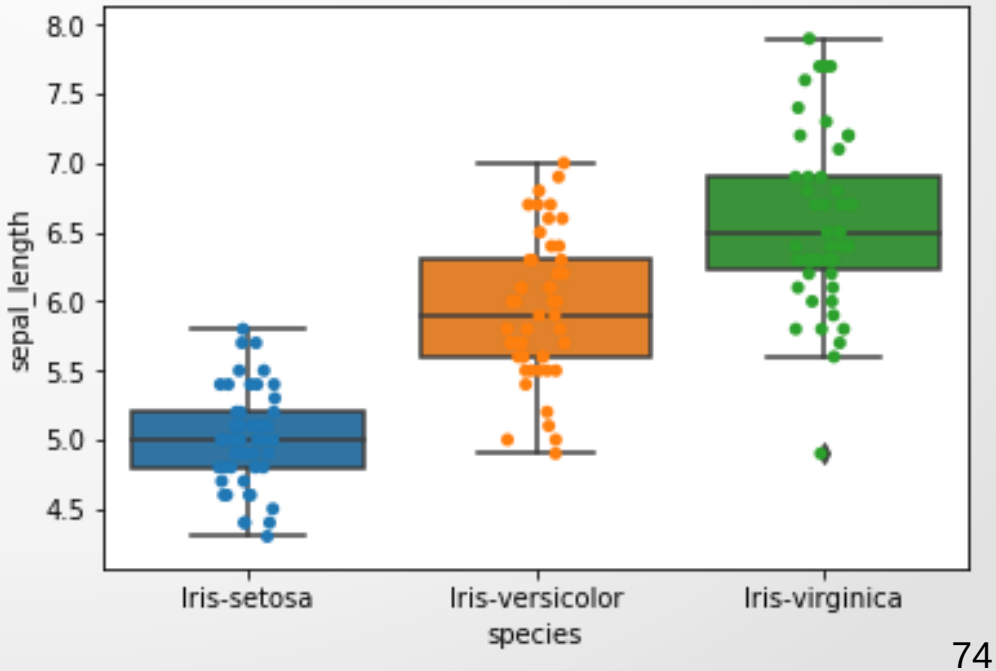
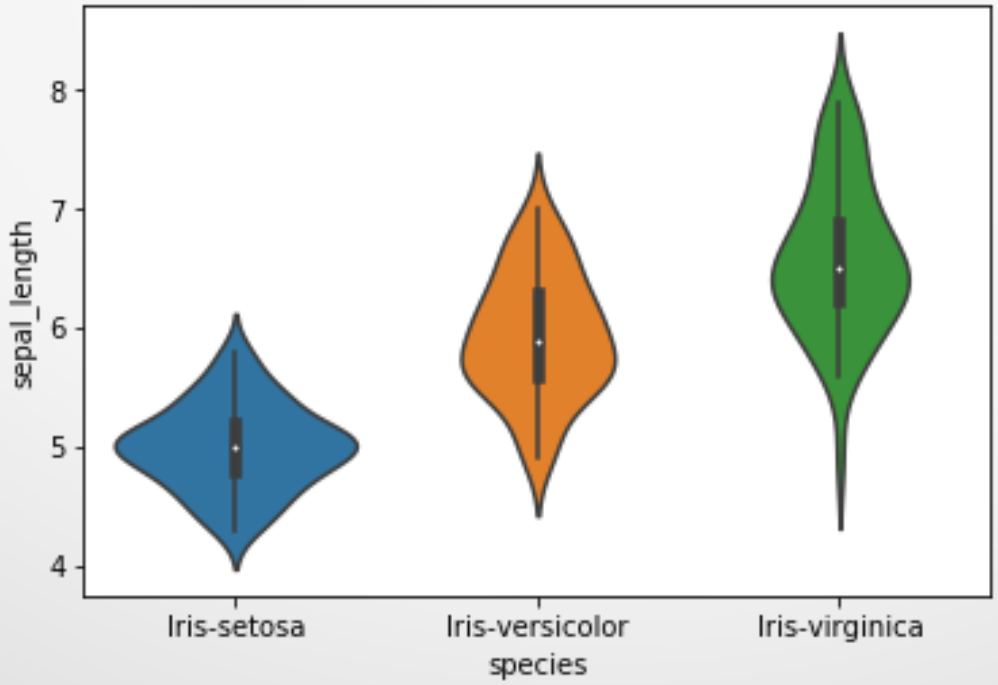


Gráfico de violinos

- Outro tipo interessante é o gráfico de violino
  - Ele possui as vantagens do gráfico de caixas e do gráfico de dispersão juntas!

- Código

```
# Violion plot do Seaborn
sns.violinplot(x="species", y="sepal_length", data=data, size=6)
```



Visualização de dados de alta dimensão

- As técnicas apresentadas anteriormente são voltadas para poucos atributos
- Gráficos de matrizes podem ser utilizados para visualização de múltiplos atributos
- Normalmente, os objetos são normalizados para evitar que um atributo domine o gráfico
- Especialmente útil quando os objetos são organizados de acordo com suas classes

Gráfico de calor

- Representa a matriz de números em forma de cores
  - Pode ser usada para expressar poucos valores

- Código

```
#Aplicando mapa de calor para a correlação entre atributos Iris
corr = data.corr()
sns.heatmap(corr, xticklabels=corr.columns.values, yticklabels=corr.columns.values)
```

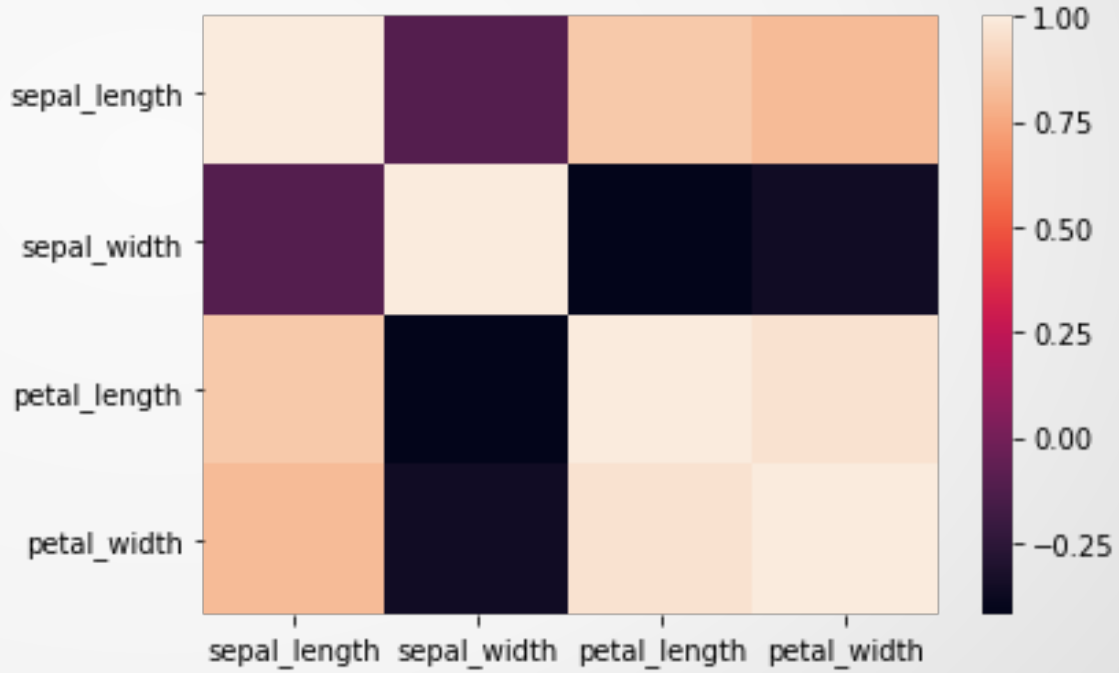


Gráfico Matrizes com Dendrograma

- Ou muitos dados!
  - Agrupamento hierárquico com dendrograma
- Código

```
#Aplicando agrupamento hierárquico com dendrograma
iris = sns.load_dataset("iris")
species = iris.pop("species")
g = sns.clustermap(iris)
```

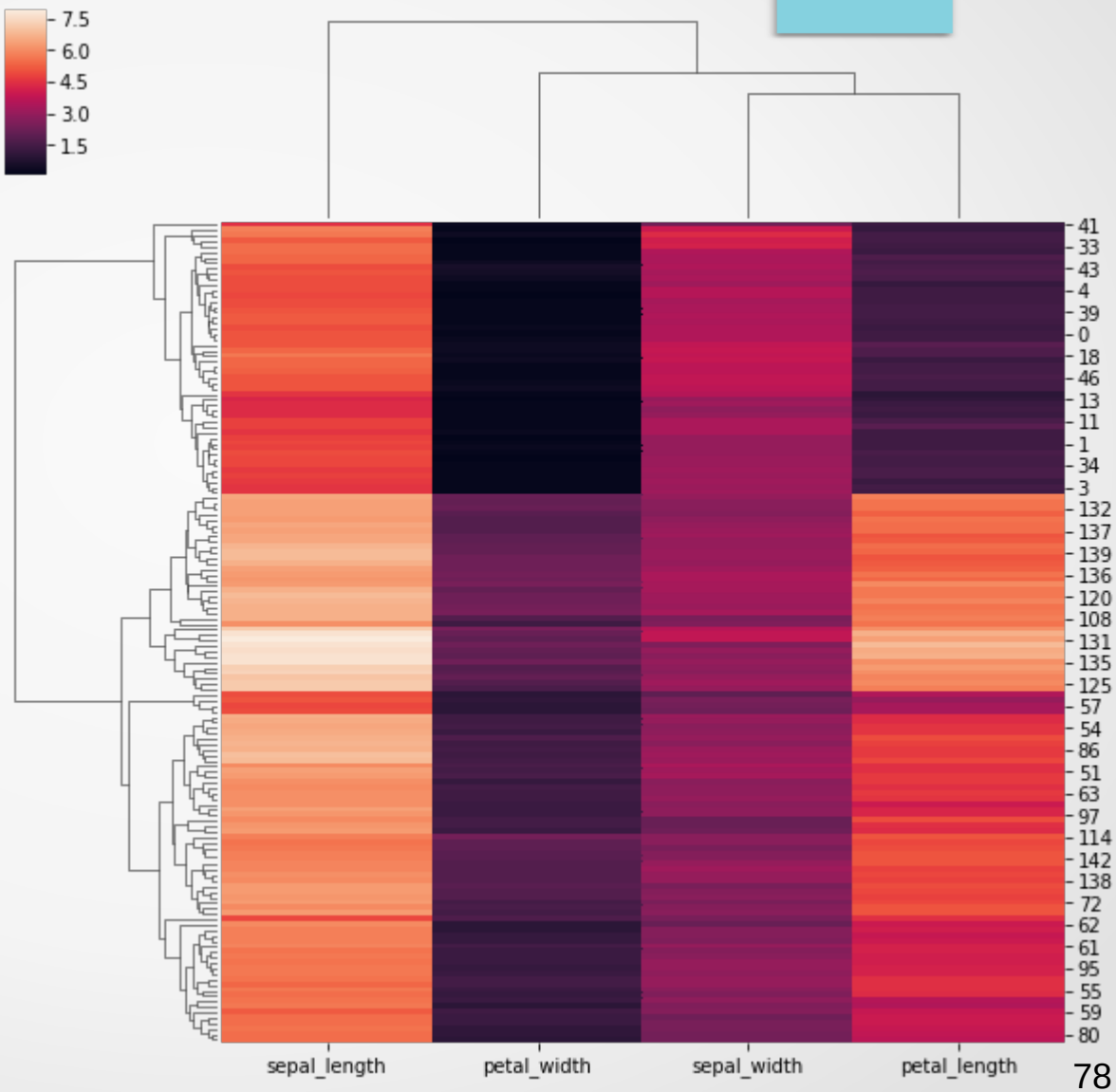
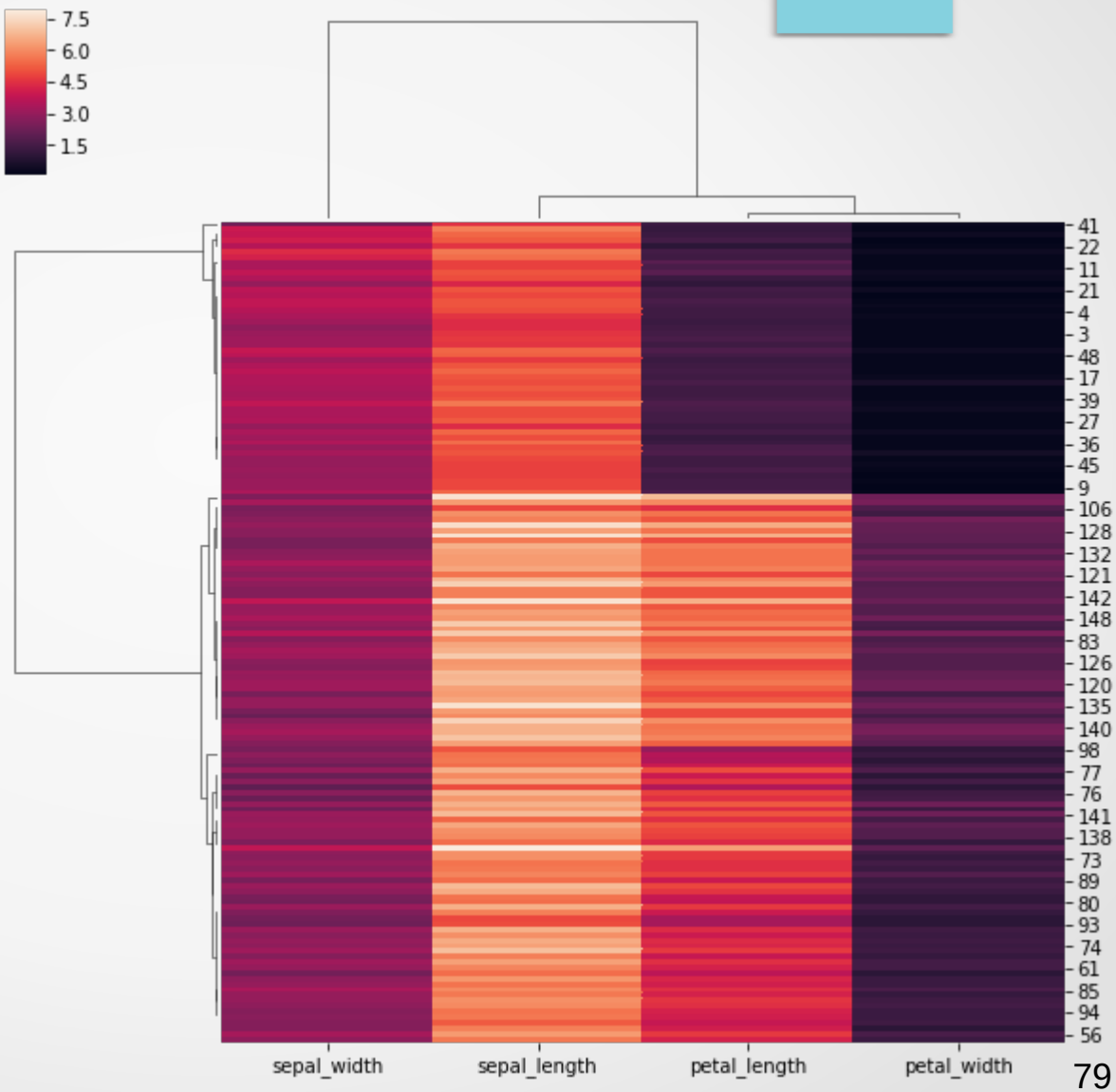


Gráfico Matrizes com Dendrograma

- Ou muitos dados!
  - Agrupamento hierárquico com dendrograma
  - Agora, com correlação no lugar da similaridade

- Código

```
#Aplicando agrupamento hierárquico com dendrograma
sns.clustermap(iris, metric="correlation")
```

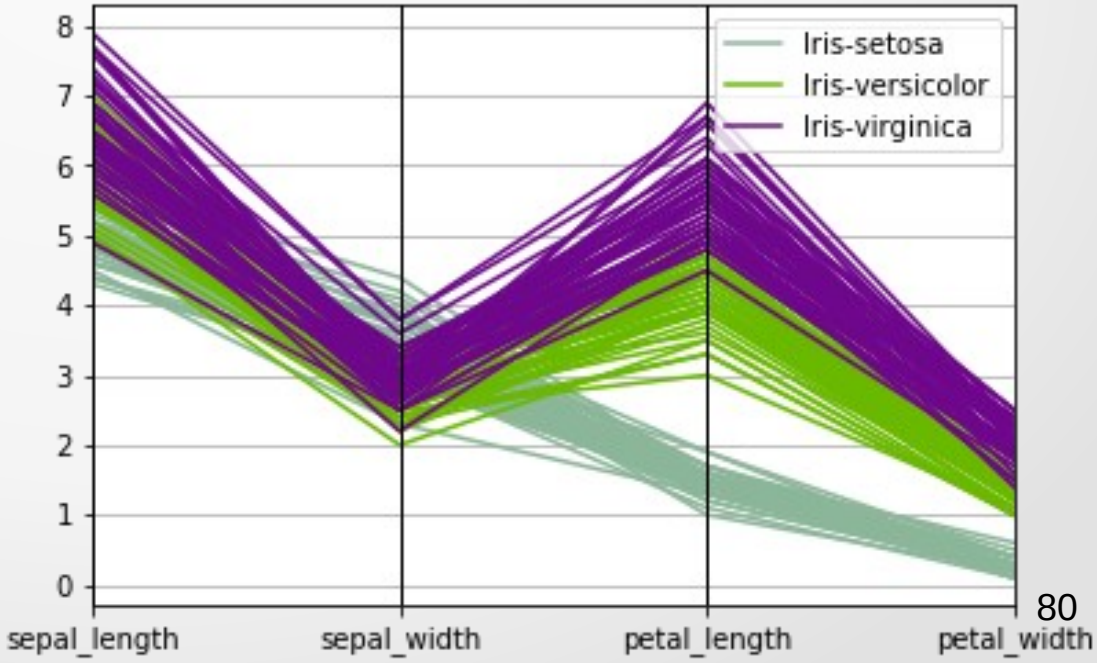


Coordenadas Paralelas

- Neste tipo de gráfico, cada atributo possui um eixo paralelo entre si em vez de perpendicular
  - Cada objeto é representado por uma linha
  - As linhas podem se agrupar representando classes
  - A ordem dos atributos é importante

- Código

```
#Coordenadas Paralelas
pd.plotting.parallel_coordinates(data, "species")
```

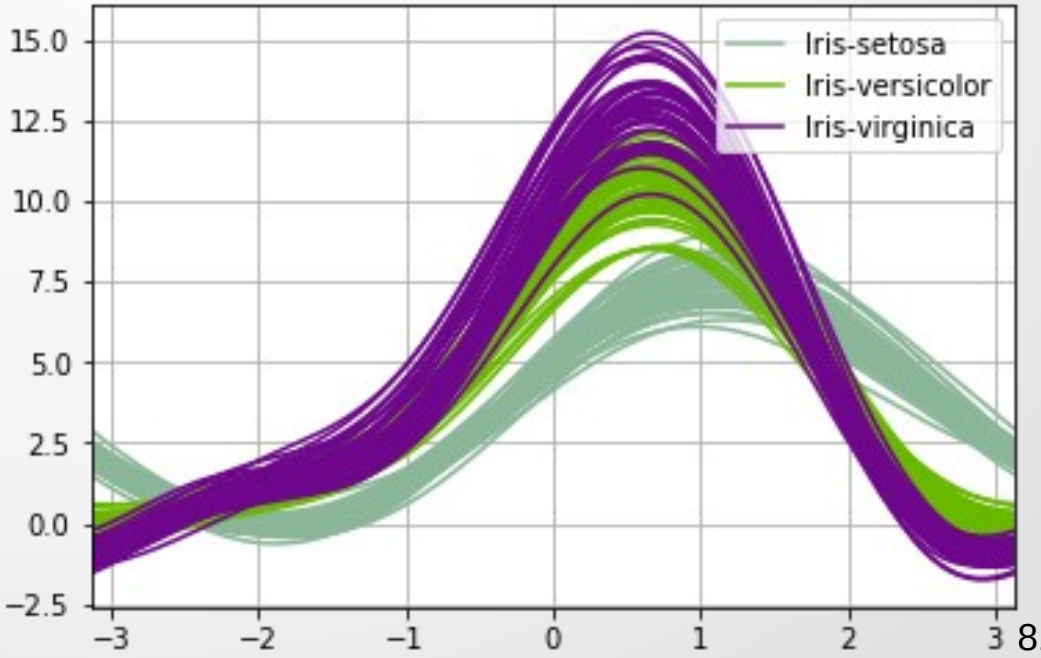


Andrews' Curves

- Proposto por Andrews, D. (1972)
- Utiliza os atributos dos dados como os coeficientes de uma série de Fourier
- Curvas (objetos) similares se agrupam
  - Externos se destacam

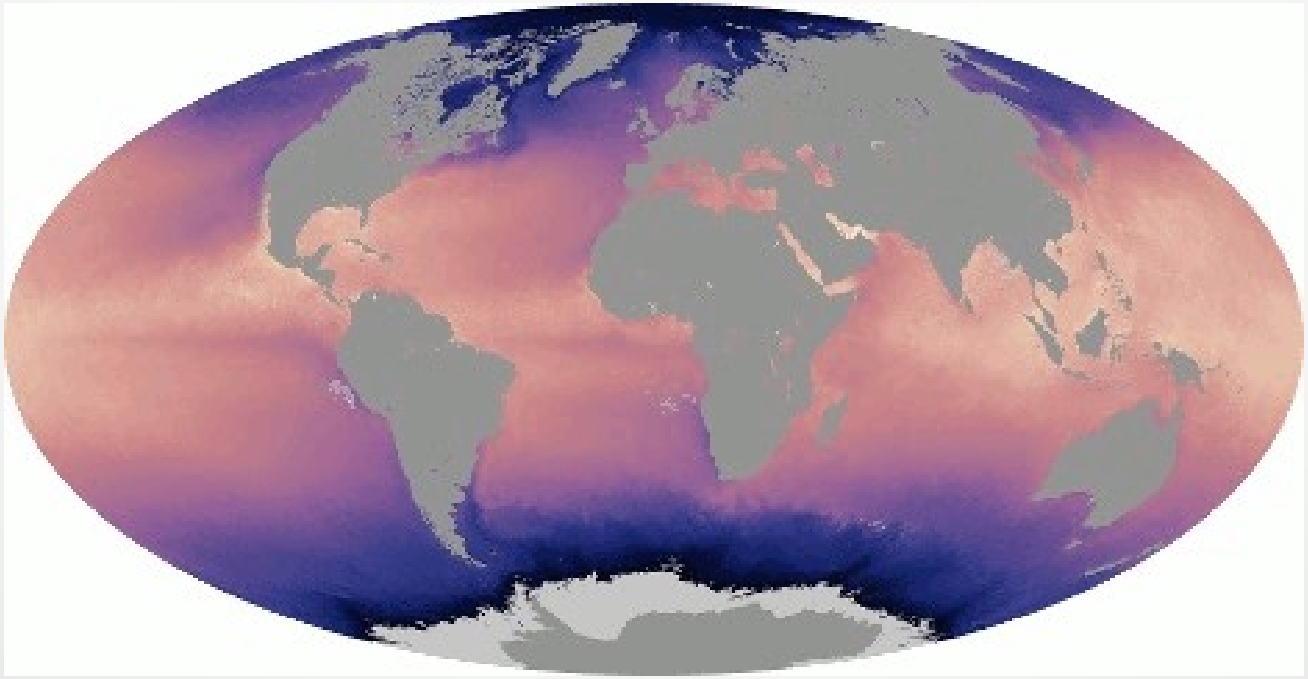
- Código

```
#Coordenadas Paralelas
pd.plotting.andrews_curves(data, "species")
```



Animações

- Uma abordagem para lidar com sequências de dados, envolvendo o tempo ou não
- Contudo, é mais difícil ver detalhes



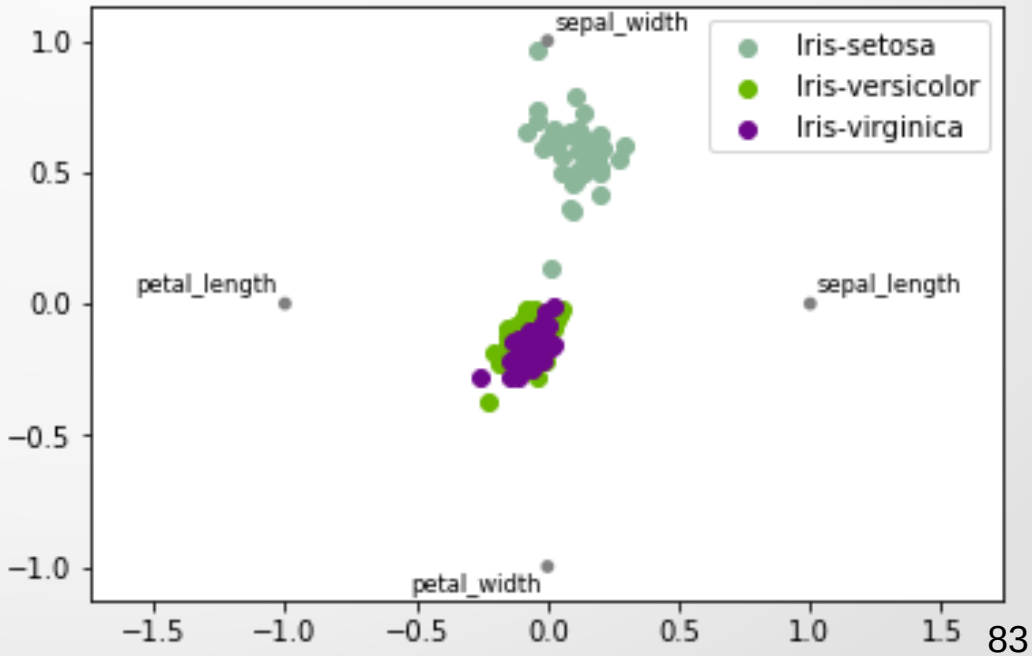
Temperatura do mar entre julho 2002 e agosto 2003. Fonte: NASA

Gráficos dados multivariados

- Radviz é uma forma de visualizar dados multivariados.
  - Cada ponto representa um atributo (normalizado)
  - Outros pontos são objetos
- Bom para ver a influência dos atributos nas classes

- Código

```
#Coordenadas Paralelas
pd.plotting.radviz(data,
"species")
```



Coordenadas Estrela

- Cada valor de atributo é convertido em uma fração que representa sua distância entre os valores mínimos e máximo do atributo e ligados

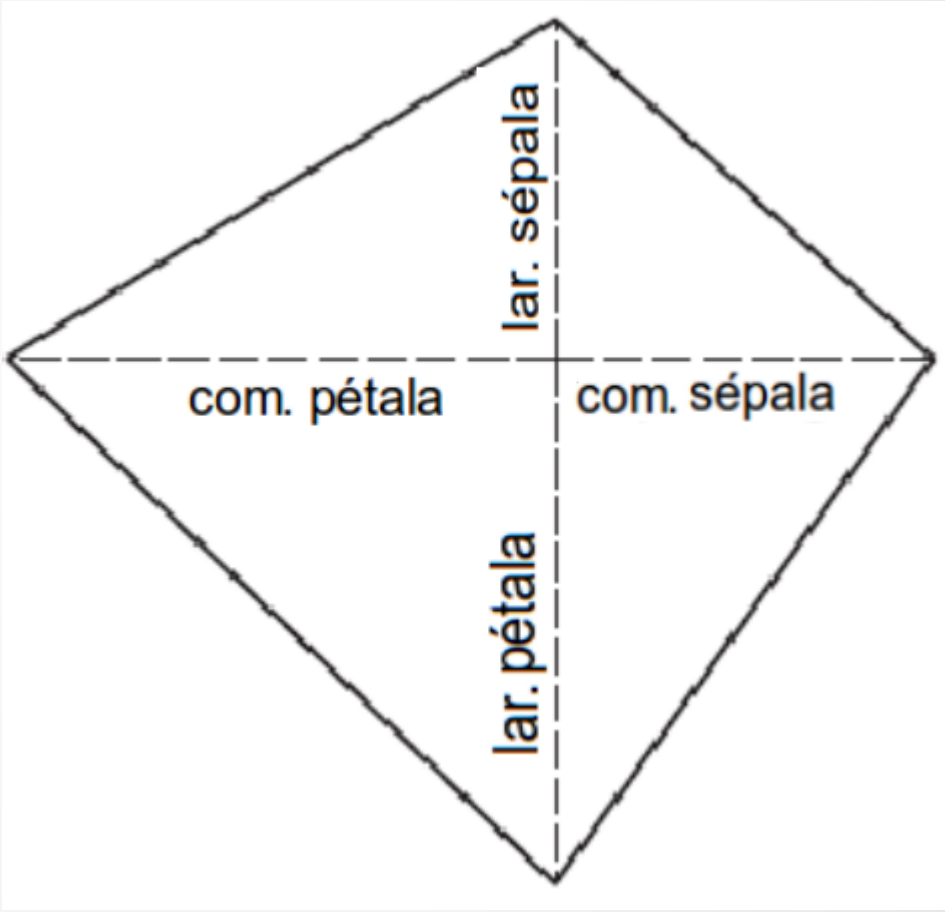
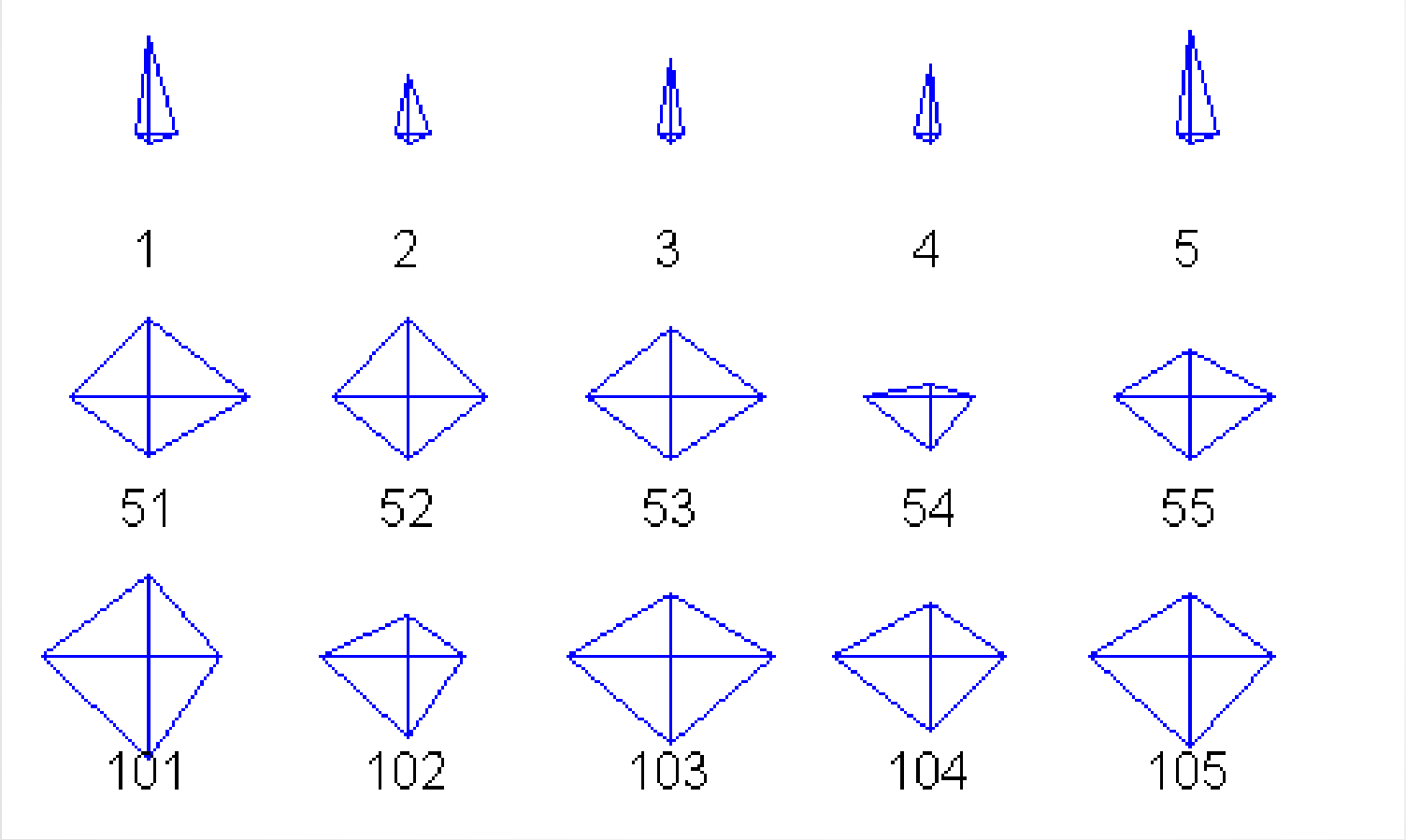


Gráfico estrela para objeto 150 do conjunto Íris
Adaptado de Tan et. al. 2009

Coordenadas Estrela



Setosa

Versicolor

Virginica

Gráfico de radar

- Semelhante as coordenadas estrela
  - Geralmente utiliza círculos para indicar valores

- Código

```
# Categorias e número de variáveis
categories=list(data)[0:4]
N = len(categories)

# Impressão do primeiro padrão do conjunto Iris
# É preciso repetir o primeiro valor para forma gráfico circular
values=data.loc[0].drop('species').values.flatten().tolist()
values += values[:1]
values

# Monta o ângulo de cada exigo no gráfico
angles = [n / float(N) * 2 * pi for n in range(N)]
angles += angles[:1]

# Inicialia o gráfico
ax = plt.subplot(111, polar=True)

# Desenha um eixo por variável e insere os rótulos
plt.xticks(angles[:-1], categories, color='grey', size=8)

# Desenha os rótulos em Y
ax.set_rlabel_position(0)
plt.yticks([2,4,6], ["2","4","6"], color="grey", size=7)
plt.ylim(0,8)

# Plota os dados
ax.plot(angles, values, linewidth=1, linestyle='solid')

# Preenche a área interna
ax.fill(angles, values, 'b', alpha=0.1)
```

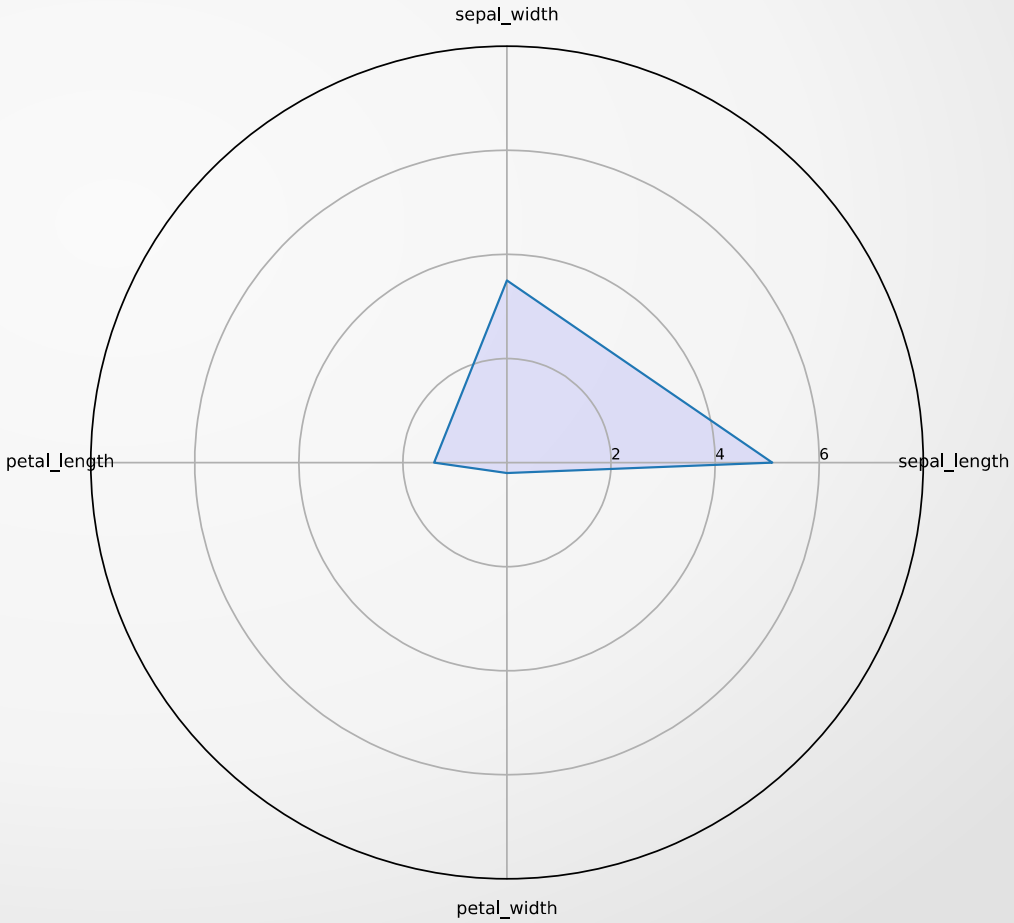
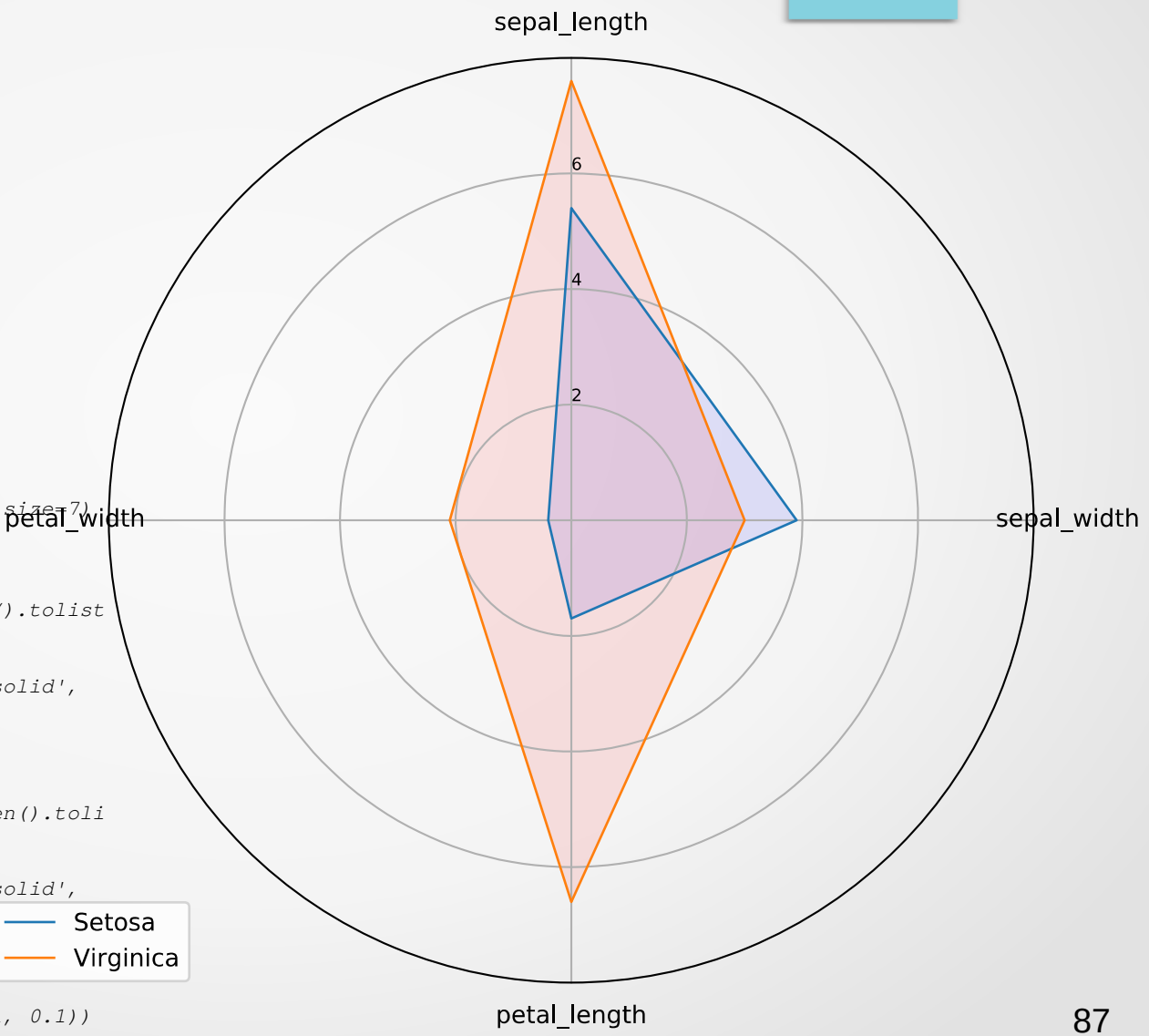


Gráfico de radar

- Código

```
# Categorias e número de variáveis
categories=list(data)[0:4]
N = len(categories)

# Monta o ângulo de cada exigo no gráfico
angles = [n / float(N) * 2 * pi for n in range(N)]
angles += angles[:1]

# Inicializa gráfico
ax = plt.subplot(111, polar=True)

# Primeiro eixo no topo para visualização
ax.set_theta_offset(pi / 2)
ax.set_theta_direction(-1)

# Desenha um eixo por variável e rotula
plt.xticks(angles[:-1], categories)

# Desenha rótulos de no eixo Y
ax.set_rlabel_position(0)
plt.yticks([2,4,6], ["2","4","6"], color="grey", size=7)
plt.ylim(0,8)

# Plotando gráfico 1, exemplo número 5
values=data.loc[5].drop('species').values.flatten().tolist
()
values += values[:1]
ax.plot(angles, values, linewidth=1, linestyle='solid',
label="Setosa")
ax.fill(angles, values, 'b', alpha=0.1)

# Plotando gráfico 2, exemplo número 105
values=data.loc[105].drop('species').values.flatten().toli
st()
values += values[:1]
ax.plot(angles, values, linewidth=1, linestyle='solid',
label="Virginica")
ax.fill(angles, values, 'r', alpha=0.1)

# Coloca legenda
plt.legend(loc='upper right', bbox_to_anchor=(0.1, 0.1))
```



OLAP

- On-Line Analytical Processing (OLAP) foi proposto por E.F. Codd, o pai do banco de dados relacional
- Bancos de dados relacionais colocam os dados em tabelas, enquanto OLAP usa um array multi-dimensional
  - Tais representações já existiam em estatística e outras áreas
- Existe uma série de análises de dados e operações de exploração que são facilitadas com este tipo de representação

Criando um array multi-dimensional

- A conversão de dados em um array multi-dimensional consiste em duas etapas:
  - Primeira etapa: identificar quais atributos devem ser as dimensões e qual atributo deve ser o atributo alvo cujos valores serão as entradas do array multi-dimensional
  - Segunda etapa: achar o valor de cada entrada no array multi-dimensional pela soma dos valores (do atributo alvo) ou contagem de todos os objetos cujos valores de atributos correspondem a entrada

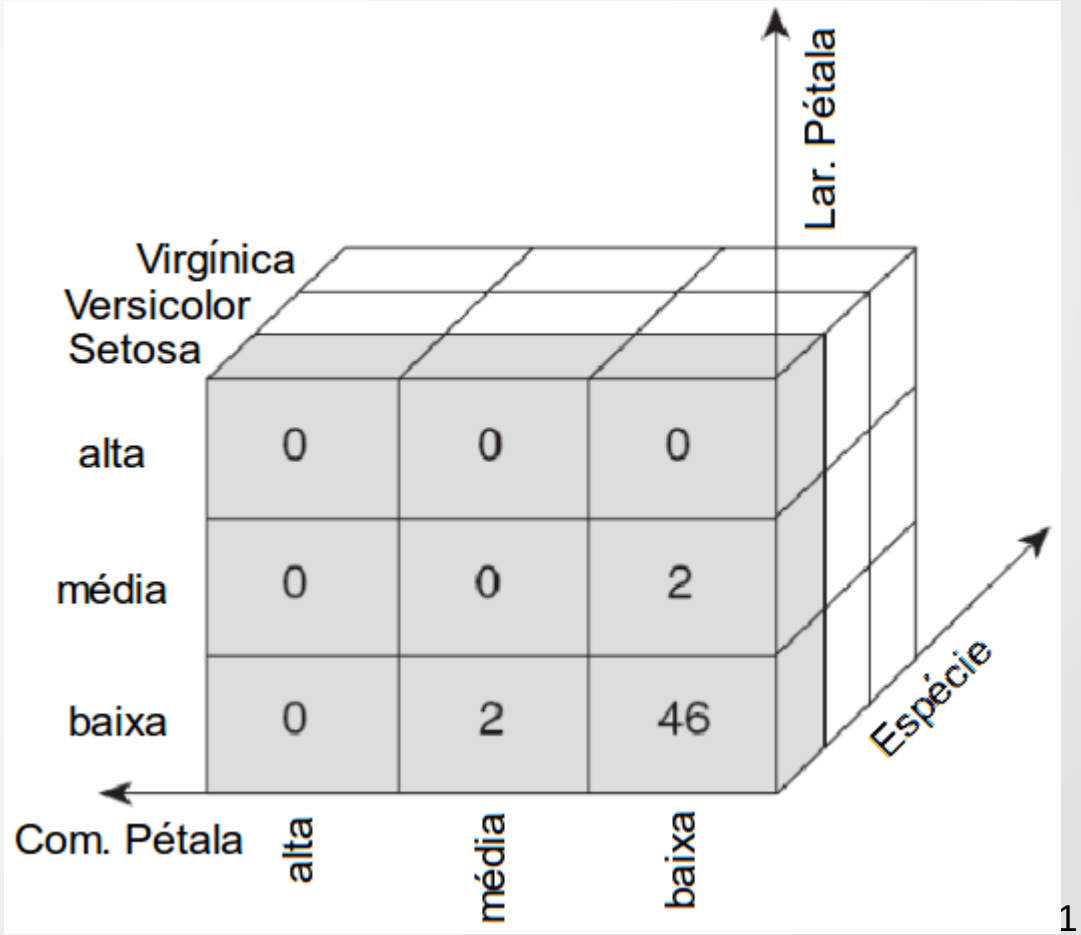
Exemplo

- Mostramos como os atributos comprimento pétala, largura de pétala e espécie podem ser convertido para um array multi-dimensional
- Primeiro, a largura e comprimento pétala são discretizados para valores categóricos: baixa, média e alta

Com. Pétala	Lar. Pétala	Espécie	Contagem
baixa	baixa	Setosa	46
baixa	média	Setosa	2
média	baixa	Setosa	2
média	média	Versicolor	43
média	alta	Versicolor	3
média	alta	Virgínica	3
alta	média	Versicolor	2
alta	média	Virgínica	3
alta	alta	Versicolor	2
alta	alta	Virgínica	44

Array multi-dimensional do conjunto Iris

- Cada tupla com lar. de pétala, com. de pétala, e espécie identifica um elemento
- O valor de contagem correspondente é atribuído
- Todas as tuplas não especificadas recebem 0



Tabulações Cruzadas

Setosa	Largura		
Comprimento	baixa	média	alta
baixa	46	2	0
média	2	0	0
alta	0	0	0

Virgínica	Largura		
Comprimento	baixa	média	alta
baixa	0	0	0
média	0	0	3
alta	0	3	44

Multicolor	Largura		
Comprimento	baixa	média	alta
baixa	0	0	0
média	0	43	3
alta	0	2	2

Exemplo Cubo Iris

- Sendo uma agregação uma soma de objetos, o cubo de dados do conjunto Iris possui:
- 3 agregações
  - com 2 dimensões
- 3 agregações
  - com 1 dimensão
- 1 agregação com
  - 0 dimensões
- Python Cubes