Python para Ciência de Dados

NumPy, Matplotlib, Pandas Introdutório

Luiz Alberto

Ciência da Computação

May 25, 2019

Importação de Dados

- Arquivos flat (e.g., .txt ou csv)
- Arquivos de outros software (e.g, Excel ou Matlab)
- Banco de dados relacionais

Lendo arquivos textos

```
1 import pandas as pd
2 nome = 'ciencia-de-dados.txt'
3 arquivo = open(nome, mode='r') # modo de leitura
4 texto = arquivo.read()
5 arquivo.close()
```

Imprimindo arquivos textos

```
1  import pandas as pd
2  nome = 'ciencia-de-dados.txt'
3  arquivo = open(nome, mode='r') # modo de leitura
4  texto = arquivo.read()
5  arquivo.close()
6  print(texto)
7
8  # Data science is a multi-disciplinary field that uses
9  # scientific methods, processes, algorithms and
10 # systems to extract knowledge and insights from structured
11 # and unstructured data. Data science is the same concept
12 # as data mining and big data: "use the most powerful
13 # hardware, the most powerful programming systems, and the
14 # most efficient algorithms to solve problems"
```

Escrevendo em arquivos textos

```
1  import pandas as pd
2  nome = 'ciencia-de-dados.txt'
3  arquivo = open(nome, mode='a') # modo escrita no final.
4  arquivo.write("\nEND-OF-FILE")
5  arquivo.close()
6
7  # Data science is a multi-disciplinary field that uses
8  # scientific methods, processes, algorithms and
9  # systems to extract knowledge and insights from structured
10 # and unstructured data. Data science is the same concept
11 # as data mining and big data: "use the most powerful
12 # hardware, the most powerful programming systems, and the
13 # most efficient algorithms to solve problems"
14 # # END-OF-FILE
```

Utilizando a cláusula **with**

```
1  import pandas as pd
2  nome = 'ciencia-de-dados.txt'
3  with open(nome, mode='r') as file:
4    print(file.read())
5  # Data science is a multi-disciplinary field that uses
6  # scientific methods, processes, algorithms and
7  # systems to extract knowledge and insights from structured
8  # and unstructured data. Data science is the same concept
9  # as data mining and big data: "use the most powerful
10 # hardware, the most powerful programming systems, and the
11 # most efficient algorithms to solve problems"
12 # # END-OF-FILE
```

Lendo o arquivo texto linha a linha

```
1  import pandas as pd
2  nome = 'ciencia-de-dados.txt'
3  with open(nome, mode='r') as arquivo:
4    print(arquivo.readline())
5    print(arquivo.readline())
6    print(arquivo.readline())
7
8  # Data science is a multi-disciplinary field that uses ...
9
10 # systems to extract knowledge and insights from structured
       ...
11
12 # is the same concept as data mining and big data: ...
```

Hora de colocar as mãos na massa

- Clone o repositorio do curso na sua máquina.
  - git clone https://github.com/gomesluiz/python-para-ciencia-de-dados.git
- Na pasta `python-para-ciencia-de-dados/notebooks`, abra o arquivo `aula-3-maos-na-massa-1.ipynb`
- Faça todos os exercícios neste notebook.

Estrutura de arquivos **flat**

python powered

titanic.csv

```
PassengerId,Survived,Pclass,Name,Gender,Age,SibSp,Parch,Ticket,Fa
re,Cabin,Embarked
1,0,3,"Braund, Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S
2,1,1,"Cumings, Mrs. John Bradley (Florence Briggs
Thayer)",female,38,1,0,PC 17599,71.2833,C85,C
3,1,3,"Heikkinen, Miss. Laina",female,26,0,0,STON/O2.
3101282,7.925,,S
```

row

	Name	Gender	Cabin	Survived
Braund, Mr. Owen Harris	male	NaN	0
Cumings, Mrs. John Bradley	female	C85	1
Heikkinen, Miss. Laina	female	NaN	1
Futrelle, Mrs. Jacques Heath	female	C123	1
Allen, Mr. William Henry	male	NaN	0

Cabeçalhos de **flat**

titanic.csv

```
PassengerId,Survived,Pclass,Name,Gender,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked
1,0,3,"Braund, Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S
2,1,1,"Cumings, Mrs. John Bradley (Florence Briggs Thayer)",female,38,1,0,PC
17599,71.2833,C85,C
3,1,3,"Heikkinen, Miss. Laina",female,26,0,0,STON/O2. 3101282,7.925,,S
4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",female,
35,1,0,113803,53.1,C123,S
5,0,3,"Allen, Mr. William Henry",male,35,0,0,373450,8.05,,S
6,0,3,"Moran, Mr. James",male,,0,0,330877,8.4583,,Q
7,0,1,"McCarthy, Mr. Timothy J",male,54,0,0,17463,51.8625,E46,S
8,0,3,"Palsson, Master. Gosta Leonard",male,2,3,1,349909,21.075,,S
9,1,3,"Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)",female,
27,0,2,347742,11.1333,,S
```

Tipos de arquivos flat

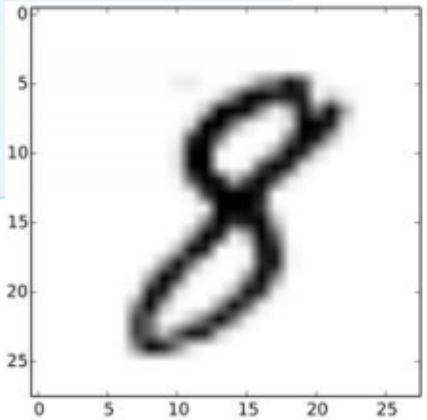
- .csv - Comma Separated Values
- .txt - Text file
- Tab-delimited file

Tab-Delimited File

MNIST.txt

pixel149	pixel150	pixel151	pixel152	pixel153
0	0	0	0	0
86	250	254	254	254
0	0	0	9	254
0	0	0	0	0
103	253	253	253	253
0	0	5	165	254
0	0	0	0	0
0	0	0	0	0
0	0	0	0	41
253	253	253	253	253

MNIST image →



Importando Arquivos com NumPy

- NumPy arrays: estrutura otimizada para dados numéricos
- Essencial para outros pacotes: e.g. scikit-learn

```
1  import numpy as np
2  arquivo = 'MNIST.TXT'
3  dados = np.loadtxt(arquivo, delimiter=',')
4  print(dados)
5  #[[7. 0. 0. ... 0. 0. 0.]
6  # [2. 0. 0. ... 0. 0. 0.]
7  # [1. 0. 0. ... 0. 0. 0.]
8  # ...
9  # [9. 0. 0. ... 0. 0. 0.]
10 #[5. 0. 0. ... 0. 0. 0.]
11 # [9. 0. 0. ... 0. 0. 0.]]
```

Hora de colocar as mãos na massa

- Na pasta `python-para-ciencia-de-dados/notebooks`, abra o arquivo `aula-3-maos-na-massa-2.ipynb`
- Faça todos os exercícios neste notebook.

Importação de arquivos com Pandas

- Estrutura bidimensionais com nomes de colunas
- Colunas com tipos diferentes de dados
- Manipulação, fatiamento, agrupamento e etc
- Estatísticas
- Series temporais

Manipulação de datafranes com Pandas

- Análise de dados exploratória
- Limpeza de dados
- Pré-processamento de dados
- Construção de modelos
- Visualização de dados

Importando arquivos csv com Pandas

```
1  import pandas as pd
2  arquivo = "winequality-red.csv"
3  dados = pd.read_csv(arquivo, sep=';') # arquivo separado por ;
4  print(dados.head())
5  #    fixed acidity  volatile acidity  citric acid
6  # 0            7.4              0.70         0.00
7  # 1            7.8              0.88         0.00
8  # 2            7.8              0.76         0.04
9  # 3           11.2              0.28         0.56
10 # 4            7.4              0.70         0.00
```

Hora de colocar as mãos na massa

- Na pasta `python-para-ciencia-de-dados/notebooks`, abra o arquivo `aula-3-maos-na-massa-3.ipynb`
- Faça todos os exercícios neste notebook.

Importando Arquivos do Excel com Pandas

```
1 import pandas as pd
2 arquivo = "condominio-balanca-mas-nao-cai.xlsx"
3 dados = pd.ExcelFile(arquivo)
4 print(dados.sheet_names)
5 # ['Jan19', 'Fev19', 'Mar19', 'Abr19']
6 df1 = dados.parse('Jan19')
7 df2 = dados.parse(0)
```

Hora de colocar as mãos na massa

- Na pasta `python-para-ciencia-de-dados/notebooks`, abra o arquivo `aula-3-maos-na-massa-4.ipynb`
- Faça todos os exercícios neste notebook.

Trabalhando com banco de dados relacionais (1)

- Baseados no modelo relacional, Edgar "Ted" Codd foi o primeiro a descrevê-lo
- Um banco de dados relacional é baseado entidades, denominadas tabelas, no relacionamento, associações, entre elas
- O contêiner fundamental em um banco de dados relacional é denominado de database ou schema
  - podem incluir estruturas de dados, os dados propriamente ditos e permissões de acesso

Trabalhando com banco de dados relacionais (2)

- Os dados são armazenados em tabelas e as tabelas são divididas em linhas e colunas. Por exemplo:

Table: comment

id	post_id	body
10	1	Python realmente...
11	2	Python facilita...
13	2	Concordo, ...

- Relacionamentos são estabelecidos entre tabelas para que a consistência dos dados seja mantida em qualquer situação e podem ser:

Trabalhando com banco de dados relacionais (3)

- 1:1, 1:N ou N:M

Table: comment

id	post_id	body
10	1	Python realmente...
11	2	Pandas facilita...
13	2	Concordo, ...

Table: post

id	title	body
1	A Ling. Python	Python é legal.
2	O Pandas	O Pandas...

Exemplo de banco de dados relacional

- Orders table

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName	ShipAddress
10248	VINET	5	7/4/1996 12:00:00 AM	8/1/1996 12:00:00 AM	7/16/1996 12:00:00 AM	3	32.38	Vins et alcools Chevalier	59 rue de l'Abbaye
10251	VICTE	3	7/8/1996 12:00:00 AM	8/5/1996 12:00:00 AM	7/15/1996 12:00:00 AM	1	41.34	Victuailles en stock	2, rue du Commerce
10254	CHOPS	5	7/11/1996 12:00:00 AM	8/8/1996 12:00:00 AM	7/23/1996 12:00:00 AM	2	22.98	Chop-suey Chinese	Hauptstr. 31

- Customers table

CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region	PostalCode	Country
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	None	12209	Germany
AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	None	WA1 1DP	UK
BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Forsterstr. 57	Mannheim	None	68306	Germany
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bouchers	Marseille	None	13008	France

- Employees table

EmployeeID	LastName	FirstName	Title	TitleOfCourtesy	BirthDate	HireDate	Address	City	Region
1	Davolio	Nancy	Sales Representative	Ms.	12/8/1948 12:00:00 AM	5/1/1992 12:00:00 AM	507 - 20th Ave. E.\r\nApt. 2A	Seattle	WA
2	Fuller	Andrew	Vice President, Sales	Dr.	2/19/1952 12:00:00 AM	8/14/1992 12:00:00 AM	908 W. Capital Way	Tacoma	WA
3	Leverling	Janet	Sales Representative	Ms.	8/30/1963 12:00:00 AM	4/1/1992 12:00:00 AM	722 Moss Bay Blvd.	Kirkland	WA

Estrutura da tabela **Orders**

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName	ShipAddress
10248	VINET	5	7/4/1996 12:00:00 AM	8/1/1996 12:00:00 AM	7/16/1996 12:00:00 AM	3	32.38	Vins et alcools Chevalier	59 rue de l'Abbaye
10251	VICTE	3	7/8/1996 12:00:00 AM	8/5/1996 12:00:00 AM	7/15/1996 12:00:00 AM	1	41.34	Victuailles en stock	2, rue du Commerce
10254	CHOPS	5	7/11/1996 12:00:00 AM	8/8/1996 12:00:00 AM	7/23/1996 12:00:00 AM	2	22.98	Chop-suey Chinese	Hauptstr. 31

Relacionamento entre as tabelas

- Orders table

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName	ShipAddress
10248	VINET	5	7/4/1996 12:00:00 AM	8/1/1996 12:00:00 AM	7/16/1996 12:00:00 AM	3	32.38	Vins et alcools Chevalier	59 rue de l'Abbaye
10251	VICTE	3	7/8/1996 12:00:00 AM	8/5/1996 12:00:00 AM	7/15/1996 12:00:00 AM	1	41.34	Victuailles en stock	2, rue du Commerce
10254	CHOPS	5	7/11/1996 12:00:00 AM	8/8/1996 12:00:00 AM	7/23/1996 12:00:00 AM	2	22.98	Chop-suey Chinese	Hauptstr. 31

- Customers table

CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region	PostalCode	Country
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	None	12209	Germany
AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	None	WA1 1DP	UK
BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Forsterstr. 57	Mannheim	None	68306	Germany
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bouchers	Marseille	None	13008	France

- Employees table

EmployeeID	LastName	FirstName	Title	TitleOfCourtesy	BirthDate	HireDate	Address	City	Region
1	Davolio	Nancy	Sales Representative	Ms.	12/8/1948 12:00:00 AM	5/1/1992 12:00:00 AM	507 - 20th Ave. E.\r\nApt. 2A	Seattle	WA
2	Fuller	Andrew	Vice President, Sales	Dr.	2/19/1952 12:00:00 AM	8/14/1992 12:00:00 AM	908 W. Capital Way	Tacoma	WA
3	Leverling	Janet	Sales Representative	Ms.	8/30/1963 12:00:00 AM	4/1/1992 12:00:00 AM	722 Moss Bay Blvd.	Kirkland	WA

Sistemas Gerenciadores de Banco de Dados

- PostgreSQL
- MySQL
- SQLite
- Oracle
- MS SQL Server
- DB2
- IBM Informix
- SQL = Structured Query Language

Manipulando Banco de Dados em Python

- Criando um vínculo com banco de dados com o pacote **SQLAlchemy** (`https://www.sqlalchemy.org/`)

```
1 from sqlalchemy import create_engine
2 engine = create_engine('sqlite:///Northwind.sqlite')
```

- Descobrindo os nomes das tabelas do banco de dados

```
1 from sqlalchemy import create_engine
2 engine  = create_engine('sqlite:///Northwind.sqlite')
3 tabelas = engine.table_names()
4 print(tabelas)
```

Etapas para Consulta no Banco de Dados (1)

- Importar pacotes e funções
- Criar o vínculo do banco de dados (database engine)
- Connectar ao banco de dados
- Consultar o banco de dados
- Salvar a consulta em um dataframe
- Fechar a conexão.

Etapas para Consulta no Banco de Dados (2)

```
1  import pandas as pd
2  from sqlalchemy import create_engine
3
4  engine = create_engine('sqlite:///Northwind.sqlite',
5    encoding='utf-16')
6
7  con = engine.connect()
8  rs = con.execute("SELECT * FROM Orders")
9
10 df = pd.DataFrame(rs.fetchall())
11 df.columns = rs.keys()
12
13 con.close()
14
15 print(df.head())
```

Utilizando a cláusula **with**

```
1  import pandas as pd
2  from sqlalchemy import create_engine
3
4  engine = create_engine('sqlite:///Northwind.sqlite')
5
6  with engine.connect() as con:
7    rs = con.execute("SELECT * FROM Orders")
8    df = pd.DataFrame(rs.fetchmany(size=5))
9    df.columns = rs.keys()
10
11 print(df.head())
```

Filtrando uma consulta com a cláusula **where**

```
1  import pandas as pd
2  from sqlalchemy import create_engine
3
4  engine = create_engine('sqlite:///Northwind.sqlite')
5
6  with engine.connect() as con:
7    rs = con.execute("SELECT * FROM Orders WHERE Freight >
       100.00")
8    df = pd.DataFrame(rs.fetchmany(size=5))
9    df.columns = rs.keys()
10
11 print(df.head())
```

Ordenando uma consulta com a cláusula **order by**

```
1  import pandas as pd
2  from sqlalchemy import create_engine
3
4  engine = create_engine('sqlite:///Northwind.sqlite')
5
6  with engine.connect() as con:
7    rs = con.execute("SELECT * FROM Orders ORDER BY OrderDate")
8    df = pd.DataFrame(rs.fetchmany(size=5))
9    df.columns = rs.keys()
10
11 print(df.head())
```

Relacionando mais de uma tabela

```
1  import pandas as pd
2  from sqlalchemy import create_engine
3
4  engine = create_engine('sqlite:///Northwind.sqlite')
5
6  with engine.connect() as con:
7    rs = con.execute(
8      "SELECT OrderID, CompanyName FROM  Orders" +
9      " INNER JOIN Customer ON Orders.CustomerID = Customer.
       CustomerID"
10   )
11   df = pd.DataFrame(rs.fetchmany(size=5))
12   df.columns = rs.keys()
13
14 print(df.head())
```

Módulos do SQLAlchemy

- Core (Foco no modelo relacional)
- ORM (Foco no modelo orientado a objetos

Conectando ao banco de dados

```
1 import pandas as pd
2 from sqlalchemy import create_engine
3
4 engine = create_engine('sqlite:///census_nyc.sqlite')
5 conn   = engine.connect()
```

- **engine**: uma interface comum entre o SQLAlchemy e o banco de dados
- **string de conexão**: todos os detalhes para localizar o banco de dados (e login, se necessário)

Utilizando o SQLAlchemy para consultas

- Fornece uma maneira "Pitônica" de construir consultas
- Esconde as diferenças entre diversos bancos de dados

```
1  import pandas as pd
2  from sqlalchemy import create_engine, Table, MetaData, select
3
4  engine = create_engine('sqlite:///Northwind.sqlite')
5  conn   = engine.connect()
6
7  metadata = MetaData()
8  pedidos  = Table('Orders', metadata, autoload=True,
9    autoload_with=engine)
10
11 commando   = select([pedidos])
12 resultados = conn.execute(commando).fetchall()
```

Extraindo campos dos resultados

```
1  import pandas as pd
2  from sqlalchemy import create_engine, Table, MetaData, select
3
4  engine = create_engine('sqlite:///Northwind.sqlite')
5  conn   = engine.connect()
6
7  metadata = MetaData()
8  pedidos  = Table('Orders', metadata, autoload=True,
9    autoload_with=engine)
10
11 #print(repr(pedidos))
12 comando    = select([pedidos])
13 resultados = conn.execute(resultados).fetchall()
14
15 for resultado in resultado[:5]:
16     print('Id:{} Data:{} Cidade:{}'.format(resultado.Id,
       resultado.OrderDate,
17       resultado.ShipCity))
```

Filtrando dados com **where** (1)

```
1  import pandas as pd
2  from sqlalchemy import create_engine, Table, MetaData, select
3
4  engine = create_engine('sqlite:///Northwind.sqlite')
5  conn   = engine.connect()
6
7  metadata = MetaData()
8  pedidos  = Table('Orders', metadata, autoload=True,
9    autoload_with=engine)
10
11 #print(repr(pedidos))
12 comando    = select([pedidos])
13 comando    = comando.where(pedidos.columns.Freight < 5)
14
15 resultados = conn.execute(comando).fetchall()
16
17 df = pd.DataFrame(resultados)
18 df.columns = resultados[0].keys()
19 print(df.head())
```

Filtrando dados com **where** (2)

```
#        Id CustomerId  EmployeeId   OrderDate      ...
# 0   10259       CENTC           4  2012-07-18     ...
# 1   10261       QUEDE           4  2012-07-19     ...
# 2   10264       FOLKO           6  2012-07-24     ...
# 3   10269       WHITC           5  2012-07-31     ...
# 4   10271       SPLIR           6  2012-08-01     ...
```