Python – Introdução ao Pandas

O pandas é uma biblioteca de código aberto, licenciada por BSD, que fornece estruturas de dados de alto desempenho e fáceis de usar e ferramentas de análise de dados para a linguagem de programação Python. Algumas coisas básicas sobre o Pandas:

- Introduz o conceito de Data Frame no Python, que são estruturas de dados alinhados de forma de tabela em linhas e colunas, divididos em dados , linhas e colunas.
- Possui várias funções para manipulação dos dados;
- É baseada em NumPy.

1. Leitura e carregamento de dados

Existem diferentes formatos de armazenamento de dados em arquivos. Não conseguiremos tratar todas as possibilidades, porém podemos destcar os seguintes:

- Dados delimitados em texto (csv);
- Microsoft Excel (xls, xlsx);
- Arquivos nativos do R (RData e rds)
- Formato fst (fst)
- SQLite (SQLITE)
- Texto não estruturado (txt).

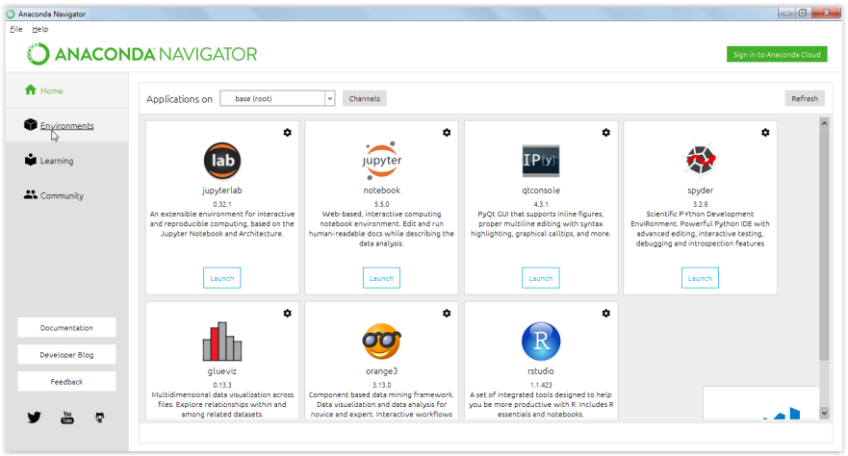
O Pandas possui funcoes para leitura de dados em diversos formatos, vamos comecar com o formato mais simples: csv. Utilizaremos como base de estudo um conjunto de dados contém uma lista de videogames com vendas superiores a 100.000 cópias. Disponível também em: https://www.kaggle.com/gregorut/videogamesales#vgsales.csv.

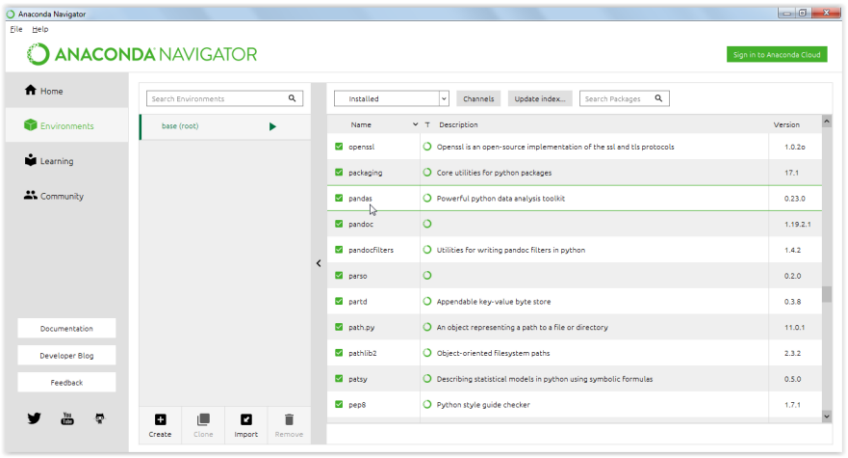
Os campos dessa base de dados incluem:

- Rank: classificação das vendas globais
- Name: o nome dos jogos
- Platform: plataforma do lançamento dos jogos (PC, PS4 etc.)
- Year: ano do lançamento do jogo
- Genre: gênero do jogo
- Publisher: editora do jogo
- NA_Sales: vendas na América do Norte (em milhões)
- EU_Sales: vendas na Europa (em milhões)
- JP_Sales: vendas no Japão (em milhões)
- Other_Sales: vendas no resto do mundo (em milhões)
- Global_Sales: total de vendas em todo o mundo.

2. Importando o Pandas

Antes de tudo, certifique-se de que o Pandas está instalado no Anaconda.





Depois importe a biblioteca pandas. Vamos iniciar os trabalhos com o uso da função read_csv(), que pode utilizar diversos parâmteros, cujos detalhes podem ser obtidos em:

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html

Básico:

- sep: Delimitador para usar.
- header: Número(s) de linha a ser usado como nome da coluna e início dos dados.
- names: Lista de nomes de colunas a serem usados. Se o arquivo não contiver nenhuma linha de cabeçalho, você deve passar explicitamente header=None. Duplicatas nesta lista não são permitidas.

Exemplo:

Dados contém uma lista de videogames com vendas superiores a 100.000 cópias. Disponível em:

https://www.kaggle.com/gregorut/videogamesales#vgsales.csv.

```
import numpy as np
import pandas as pd
games = pd.read_csv(
            "vgsales.csv",
            sep=',',
            header=0
        )
print(games.shape)
games.head()
```

Resultado:

```
(16598, 11)
```

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

```
sep: Delimitador para usar.
header: Número(s) de linha a ser usado como nome da coluna e início dos dados.
names: Lista de nomes de colunas a serem usados. Se o arquivo não contiver nenhuma linha de cabeçalho, você deve passar
explicitamente header=None. Duplicatas nesta lista não são permitidas.
```

```
import numpy as np
import pandas as pd
games = pd.read_csv(
          "vgsales.csv",
          sep=',',
          header=0
        )
print(games.shape)
games.head()
```

```
(16598, 11)
```

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

Outro exemplo:

Dados coletados do catálogo de dados abertos sibre crimes da cidade de Vancouver. Disponível em:

https://www.kaggle.com/agilesifaka/vancouver-crime-report/downloads/vancouver-crime-report.zip/1

```
import numpy as np
import pandas as pd
games = pd.read_csv("crime_records.csv",sep=',',header=0)
print(games.shape)
games.head()
```

Resultado:

```
(608815, 10)
```

	TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED_BLOCK	NEIGHBOURHOOD	X	Y
0	Theft from Vehicle	2003	6	28	13.0	30.0	8XX EXPO BLVD	Central Business District	491771.63	5458295.01
1	Theft from Vehicle	2003	11	17	16.0	0.0	56XX OAK ST	South Cambie	490682.32	5453536.96
2	Theft from Vehicle	2003	12	30	14.0	0.0	85XX STANLEY PARK DR	Stanley Park	489104.19	5460347.36
3	Theft of Vehicle	2003	1	15	14.0	45.0	6XX W 41ST AVE	Oakridge	491372.94	5453422.83
4	Theft from Vehicle	2003	12	28	16.0	45.0	85XX STANLEY PARK DR	Stanley Park	489104.19	5460347.36

3. Obtendo descrições e tipos dos dados

Precisamos sempre verificar se o carregamento dos dados foram feitos de forma correta, e se foram lidos no formato correto, para isso podemos fazer o seguinte:

```
print(games.describe(),"\n")
print(games.dtypes)
```

```
                YEAR          MONTH            DAY           HOUR  \
count  608815.000000  608815.000000  608815.000000  547782.000000
mean     2010.329796       6.466657      15.390511      13.723485
std         5.053476       3.403654       8.743798       6.751120
min      2003.000000       1.000000       1.000000       0.000000
25%      2006.000000       4.000000       8.000000       9.000000
50%      2010.000000       6.000000      15.000000      15.000000
75%      2015.000000       9.000000      23.000000      19.000000
max      2019.000000      12.000000      31.000000      23.000000

              MINUTE              X             Y
count  547782.000000  608815.000000  6.088150e+05
mean       17.092035  442884.965919  4.909681e+06
std        18.455757  147854.915444  1.638825e+06
min         0.000000       0.000000  0.000000e+00
25%         0.000000  490011.835000  5.453756e+06
50%        11.000000  491526.360000  5.456901e+06
75%        30.000000  493496.230000  5.458680e+06
max        59.000000  511303.000000  5.512579e+06

TYPE              object
YEAR               int64
MONTH              int64
DAY                int64
HOUR             float64
MINUTE           float64
HUNDRED_BLOCK     object
NEIGHBOURHOOD     object
X                float64
Y                float64
dtype: object
```