Felipe Galvão

Web Development / Desenvolvimento Web

Ciência de Dados com Python – Básico do Pandas – Leitura de DataFrames

Felipe Galvão – Python (Português) – fevereiro 18, 2016

English version of this post / Versão em inglês deste post

Introdução

Em um post anterior já mostramos como instalar a distribuição Anaconda para poder utilizar o Pandas para Análise e Ciência de Dados (clique aqui para ver). Agora, passado este passo, vamos introduzir um pouco o Pandas falando de sua principal estrutura, os DataFrames.

Os DataFrames são estruturas que comportam dados de forma tabular. Os DataFrames são compostos de linhas e colunas, sendo cada coluna um campo da tabela e cada linha um registro. Cada coluna possui dados de um mesmo tipo. Você pode imaginar uma tabela de Excel, mas nesse caso, cada coluna é limitada a um tipo de dado.

Criação de um DataFrame

Primeiro, uma forma bem simples de criar um DataFrame, através do bom e velho Dictionary do Python (vamos importar o pandas com o primeiro comando pois é a primeira vez que executamos código neste post):

```
import pandas as pd

df_data = {'pais': ['Brasil', 'Argentina','Argentina', 'Brasil', 'Chile','Chile'],
           'ano': [2005, 2006, 2005, 2006, 2007, 2008],
           'populacao': [170.1, 30.5, 32.2, 172.6, 40.8, 42.0]}
df = pd.DataFrame(df_data)
print(df)
```

```
    ano       pais  populacao
0  2005     Brasil      170.1
1  2006  Argentina       30.5

2  2005  Argentina       32.2
3  2006     Brasil      172.6
4  2007      Chile       40.8
5  2008      Chile       42.0
```

Outra forma, bem mais utilizada, é a importação de dados de um arquivo (nos mais variados formatos) ou banco de dados direto para um DataFrame.

Vamos utilizar o famoso exemplo do Titanic, do Kaggle, para exemplificar. Para quem não sabe o que é, o Kaggle é um site de competições de Ciência de Dados. Normalmente patrocinados por empresas, as competições geralmente envolvem previsões através de Datasets de treinamento e teste, mas também há competições envolvendo tarefas como reconhecimento de imagem e criação de análises interessantes com os Datasets providos pelo site. Introdução feita, vamos lá.

Cadastre-se no Kaggle (www.kaggle.com) e entre em "Competitions". Desça a tela e encontre a competição "Titanic: Machine Learning from Disaster". Clique nela e então você poderá ler sobre o que consiste esta competição. Esta á uma "competição" permanente no

Kaggle, que existe basicamente para que as pessoas possam treinar e aprender sobre Ciência de Dados e Machine Learning, aplicando novos conhecimentos adquiridos em um Dataset pronto para ser usado.

Nela, são providenciadas diversas informações sobre os passageiros em um Dataset de treino e um Dataset de teste, como idade, sexo, cabine, valor do tíquete pago, entre outros. Além disso, no Dataset de treino é fornecida a variável alvo, que diz se um passageiro sobreviveu (valor = 1) ou não (valor = 0) ao naufrágio. Já no Dataset de teste, são dadas as mesmas informações sobre outros passageiros, com exceção da variável alvo (a sobrevivência). Para os passageiros do Dataset de teste, você deverá prever se eles sobreviveram ou não, a partir das suas características, em comparação com as informações fornecidas no Dataset de treino. Na página da competição você também pode ver outros detalhes, como o que significa cada variável do Dataset, Leaderboard, alguns tutoriais envolvendo a competição.

Dada esta breve introdução, dentro da competição do Titanic, entre em Data e baixe os arquivos train e test. Estes são os Datasets de treino e teste, dois arquivos .csv. Salve-os em alguma pasta e crie o seu script Python na mesma pasta. O comando read_csv do Pandas lê o arquivo CSV e o armazena em um Dataframe. Depois iremos printar usando o comando head() dos Dataframes, que mostra as 5 primeiras linhas do mesmo, só para mostrar que o Dataset está realmente armazenado na variável. Vejamos (a partir de agora, consideraremos que o Pandas já está importado como pd, conforme o bloco de código anterior):

```
import pandas as pd

train_dataset = pd.read_csv('train.csv')

print(train_dataset.head())
```

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex  Age  SibSp  \
0                            Braund, Mr. Owen Harris    male   22      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female   38      1
2                             Heikkinen, Miss. Laina  female   26      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female   35      1
4                           Allen, Mr. William Henry    male   35      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

Lendo informações de um DataFrame

Para extrair os valores de uma coluna, existem duas formas bastante comuns. A primeira é chamar a coluna desejada como se o DataFrame fosse um Dictionary. A segunda é analoga a chamar um atributo de um objeto. Pela primeira forma também é possível ver o valor de mais de uma coluna de uma vez:

```
print(train_dataset['Age'])
```

```
0      22
1      38
2      26
3      35
4      35
...
886    27
887    19
888   NaN
889    26
890    32
```

```
print(train_dataset.Sex)
```

```
0        male
1      female
2      female
3      female
4        male
...
886      male
887    female
888    female
889      male
890      male
```

```
print(train_dataset[['Sex','Age','Fare']])
```

```
        Sex  Age      Fare
0      male   34    7.2500
1    female   10   71.2833
2    female   30    7.9250
3    female    5   53.1000
4      male   22    8.0500
...
886    male   59   13.0000
887  female   30   30.0000
888  female   19   23.4500
889    male   57   30.0000
890    male    7    7.7500
```

Para ver o conteúdo de linhas, usamos o campo ix, passando a ele a posição da linha. Caso seja passado uma lista de valores, todas as linhas são mostradas:

```
print(train_dataset.ix[3])
```

```
PassengerId                                     4
Survived                                        1
Pclass                                          1
Name           Futrelle, Mrs. Jacques Heath (Lily May Peel)
Sex                                        female
Age                                            35
SibSp                                           1
Parch                                           0
```

```
Ticket                                       113803
Fare                                           53.1
Cabin                                          C123
Embarked                                          S
```

```
print(train_dataset.ix[[0,10,50]])
    PassengerId  Survived  Pclass                            Name     Sex  \
0             1         0       3         Braund, Mr. Owen Harris    male
10           11         1       3  Sandstrom, Miss. Marguerite Rut  female
50           51         0       3       Panula, Master. Juha Niilo    male

    Age  SibSp  Parch     Ticket     Fare Cabin Embarked
0    22      1      0  A/5 21171   7.2500   NaN        S
10    4      1      1    PP 9549  16.7000    G6        S
50    7      4      1    3101295  39.6875   NaN        S
```

Como já vimos, o head() mostra as 5 primeiras linhas do DataFrame. Caso passemos um número inteiro para ele, ele retornará este mesmo número de linhas. Analogamente, o tail() mostra as 5 últimas linhas. Para obter uma lista com o nome de cada coluna, usamos columns(). Para a quantidade de colunas e linhas, podemos usar o shape(), que retorna um Tuple com ambos os valores, sendo o [0] o número de linhas e o [1] o número de colunas:

```
print(train_dataset.head(7))
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3
5            6         0       3
6            7         0       1

                                                Name     Sex  Age  SibSp  \
0                            Braund, Mr. Owen Harris    male   22      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female   38      1
2                             Heikkinen, Miss. Laina  female   26      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female   35      1
4                           Allen, Mr. William Henry    male   35      0
5                                   Moran, Mr. James    male  NaN      0
6                            McCarthy, Mr. Timothy J    male   54      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
5      0            330877   8.4583   NaN        Q
6      0             17463  51.8625   E46        S
```

```
print(train_dataset.tail())
```

```
print(train_dataset.head())
     PassengerId  Survived  Pclass                                          Name  \
886          887         0       2                         Montvila, Rev. Juozas
887          888         1       1                  Graham, Miss. Margaret Edith
888          889         0       3      Johnston, Miss. Catherine Helen "Carrie"
889          890         1       1                         Behr, Mr. Karl Howell
890          891         0       3                           Dooley, Mr. Patrick

        Sex  Age  SibSp  Parch      Ticket   Fare Cabin Embarked
886    male   27      0      0      211536  13.00   NaN        S
887  female   19      0      0      112053  30.00   B42        S
888  female  NaN      1      2  W./C. 6607  23.45   NaN        S
889    male   26      0      0      111369  30.00  C148        C
890    male   32      0      0      370376   7.75   NaN        Q

print(train_dataset.columns)
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')

print(train_dataset.shape)
(891, 12)
```

Uma função bastante interessante dos DataFrames é o describe(). O describe calcula estatísticas para cada coluna numérica do DataFrame, como contagem de valores, soma, média, mediana, etc. Ele é interessante quando se quer ter uma ideia dos dados que estão sendo trabalhados:

```
print(train_dataset.describe())
       PassengerId    Survived      Pclass         Age       SibSp  \
count   891.000000  891.000000  891.000000  714.000000  891.000000
mean    446.000000    0.383838    2.308642   29.699118    0.523008
std     257.353842    0.486592    0.836071   14.526497    1.102743
min       1.000000    0.000000    1.000000    0.420000    0.000000
25%     223.500000    0.000000    2.000000   20.125000    0.000000
50%     446.000000    0.000000    3.000000   28.000000    0.000000
75%     668.500000    1.000000    3.000000   38.000000    1.000000

max     891.000000    1.000000    3.000000   80.000000    8.000000

            Parch        Fare
count  891.000000  891.000000
mean     0.381594   32.204208
std      0.806057   49.693429
min      0.000000    0.000000
25%      0.000000    7.910400
50%      0.000000   14.454200
75%      0.000000   31.000000
max      6.000000  512.329200
```

Uma última funcionalidade importante na leitura de dados de um DataFrame é a possibilidade de filtra-lo. Filtraremos o DataFrame pelo valor da coluna “Sex”. Selecionaremos apenas as linhas do DataFrame onde o valor da coluna “Sex” seja igual a female:

```
print(train_dataset[train_dataset.Sex == "female"])
     PassengerId  Survived  Pclass  \
1              2         1       1
2              3         1       3
3              4         1       1
8              9         1       3
9             10         1       2
..           ...       ...     ...
880          881         1       2
882          883         0       3

885          886         0       3
887          888         1       1
888          889         0       3

                                                 Name     Sex  Age  SibSp  \
1    Cumings, Mrs. John Bradley (Florence Briggs Th...  female   38      1
2                                Heikkinen, Miss. Laina  female   26      0
3          Futrelle, Mrs. Jacques Heath (Lily May Peel)  female   35      1
8    Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)  female   27      0
9                  Nasser, Mrs. Nicholas (Adele Achem)  female   14      1
..                                                 ...     ...  ...    ...
880       Shelley, Mrs. William (Imanita Parrish Hall)  female   25      0
882                         Dahlberg, Miss. Gerda Ulrika  female   22      0
885                Rice, Mrs. William (Margaret Norton)  female   39      0
887                       Graham, Miss. Margaret Edith  female   19      0
888             Johnston, Miss. Catherine Helen "Carrie"  female  NaN      1

     Parch            Ticket      Fare Cabin Embarked
1        0          PC 17599   71.2833   C85        C
2        0  STON/O2. 3101282    7.9250   NaN        S
3        0            113803   53.1000  C123        S
8        2            347742   11.1333   NaN        S
9        0            237736   30.0708   NaN        C
..     ...               ...       ...   ...      ...
880      1            230433   26.0000   NaN        S
882      0              7552   10.5167   NaN        S
885      5            382652   29.1250   NaN        Q
887      0            112053   30.0000   B42        S
888      2        W./C. 6607   23.4500   NaN        S
```

Acredito que esse seja um bom começo na parte de leitura. No próximo post falarei um pouco de manipulação de dados, com criação de colunas, modificação de valores de coluna em DataFrames, entre outros assuntos.

Fiquem ligados! 🙂

Tags ANALISE DE DADOS BASIC PYTHON CIÊNCIA DE DADOS DATA ANALYSIS DATA SCIENCE DATAFRAME DATAFRAMES PANDAS PROGRAMACAO PYTHON PYTHON BASICO

Data Science with Python – Pandas Basics – Reading Dataframes Data Manipulation with Python (Pandas)