**05 - Trabalhando com dados**

É esperado que alguém que está iniciando sua carreira em Ciência de Dados consiga trabalhar com dados. Nesse cenário, o Python tem disponível uma biblioteca bastante importante, chamada **Pandas**, sobre a qual faremos um pequeno curso introdutório ao final deste treinamento.

Inicialmente gostaríamos de apresentar como ele funciona, abrindo um arquivo externo e colocando-o em uma estrutura de dados que é reconhecida pelo Pandas, além de analisarmos um pouco como esses dados funcionam. No futuro entenderemos mais profundamente como funcionam os tratamentos, faremos estatísticas descritivas e assim por diante.

No curso anterior nós trabalhamos com um *dataset* contendo características e valores de automóveis. A ideia é gerar um modelo de regressão voltado à precificação desses automóveis com base em tais características (um modelo de *machine learning*). Para isso, é necessário conhecer e tratar esses dados, algo que é bastante facilitado pelo Pandas.

No notebook Python do Colab, abriremos a aba na lateral esquerda, clicaremos em "Files" e então "Upload" para subirmos o arquivo db.csv (encontrado na pasta "data" dentre os arquivos baixados anteriormente). Como usaremos uma biblioteca que não vem previamente instalada, será necessário importá-la, algo que é feito usando o comando import.

Nesse caso, faremos import pandas as pd de modo a não só realizarmos a importação, mas também atribuirmos à biblioteca o apelido pd (facilitando a utilização dos seus comandos). Após executarmos com "Shift + Enter", prosseguiremos para a importação dos dados no nosso conjunto.

Criaremos uma variável dataset que receberá a chamada de pd.read\_csv() - o apelido da biblioteca, pd, seguido da função read\_csv(), que realiza a leitura de arquivos. Como parâmetro passaremos o arquivo que acabamos de subir, dv.csv.

dataset = pd.read\_csv('db.csv')COPIAR CÓDIGO

Ao executarmos receberemos alguns erros, pois existe um argumento faltando: sep, que define o separador utilizado no conjunto de dados. Nesse caso, o separador é ponto e vírgula (;). Poderemos então visualizar nosso conjunto a partir da variável dataset.

dataset = pd.read\_csv('db.csv', sep = ';')

datasetCOPIAR CÓDIGO

| **Nome** | **Motor** | **Ano** | **Quilometragem** | **Zero\_km** | **Acessórios** | **Valor** |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | Jetta Variant | Motor 4.0 Turbo | 2003 | 44410.0 | False | ['Rodas de liga', 'Travas elétricas', 'Piloto ... | 88078.64 |
| 1 | Passat | Motor Diesel | 1991 | 5712.0 | False | ['Central multimídia', 'Teto panorâmico', 'Fre... | 106161.94 |
| 2 | Crossfox | Motor Diesel V8 | 1990 | 37123.0 | False | ['Piloto automático', 'Controle de estabilidad... | 72832.16 |
| 3 | DS5 | Motor 2.4 Turbo | 2019 | NaN | True | ['Travas elétricas', '4 X 4', 'Vidros elétrico... | 124549.07 |
| 4 | Aston Martin DB4 | Motor 2.4 Turbo | 2006 | 25757.0 | False | ['Rodas de liga', '4 X 4', 'Central multimídia... | 92612.10 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 253 | Phantom 2013 | Motor V8 | 2014 | 27505.0 | False | ['Controle de estabilidade', 'Piloto automátic... | 51759.58 |
| 254 | Cadillac Ciel concept | Motor V8 | 1991 | 29981.0 | False | ['Bancos de couro', 'Painel digital', 'Sensor ... | 51667.06 |
| 255 | Classe GLK | Motor 5.0 V8 Bi-Turbo | 2002 | 52637.0 | False | ['Rodas de liga', 'Controle de tração', 'Câmbi... | 68934.03 |
| 256 | Aston Martin DB5 | Motor Diesel | 1996 | 7685.0 | False | ['Ar condicionado', '4 X 4', 'Câmbio automátic... | 122110.90 |
| 257 | Macan | Motor Diesel V6 | 1992 | 50188.0 | False | ['Central multimídia', 'Teto panorâmico', 'Vid... | 90381.47 |

Perceba que temos o mesmo conteúdo visualizado no arquivo CSV, que também pode ser aberto em um editor de planilhas como o Microsoft Excel. Agora temos um dataframe, a estrutura de dados disponibilizada pelo Pandas para trabalharmos com dados. Dentre as informações, temos o nome dos veículos, tipo de motor, ano, quilometragem, se o veículo é ou não zero quilômetros, os acessórios e o valor.

Repare que nosso retorno só nos mostrou dez linhas, mas é possível visualizar outras. Utilizando a função pd.set\_option() é possível definir alguns parâmetros do dataframe, como o número máximo de linhas retornadas, algo que configuraremos com display.max\_rows seguido do número determinado de linhas, neste exemplo 1000.

import pandas as pd

pd.set\_option('display.max\_rows', 1000)COPIAR CÓDIGO

Se executarmos dataset novamente, receberemos todas as 258 linhas - não foram mostradas 1000 pois nosso conjunto não possui esse total de informações. Também temos uma opção para exibir mais colunas, chamada display.max\_columns. Como gostaríamos de manter a configuração padrão, comentaremos ambas as linhas.

import pandas as pd

# pd.set\_option('display.max\_rows', 10000)

# pd.set\_option('display.max\_columns', 1000)COPIAR CÓDIGO

É possível obtermos os tipos de dados que temos no conjunto, algo que também fizemos no curso anterior, utilizando o comando dataset.dtypes.

dataset.dtypesCOPIAR CÓDIGO

Nome object Motor object Ano int64 Quilometragem float64 Zero\_km bool Acessórios object Valor float64 dtype: object

Aqui temos os tipos de cada coluna do dataframe. No Pandas, todo valor do tipo string é tratado como *object*. Além deste, temos o ano como tipo inteiro (int64), quilometragem como float64, Zero\_km como booleano (bool) e assim por diante. Note que, diferentemente dos arrays Numpy, nos dataframes do Pandas é possível trabalharmos com dados de tipos diferentes sem nenhum problema.

Vamos supor que desejamos calcular uma estatística descritiva com esse conjunto. Em nosso exemplo, usaremos as colunas "quilometragem" e "valor". Usaremos a variável dataset, que contém o nosso conjunto, e passaremos entre dois colchetes (pois estamos passando mais de uma informação) as duas variáveis que possuem os dados desejados - Quilometragem e Valor). Por fim, adicionaremos um ponto (.) e a função describe().

dataset[['Quilometragem', 'Valor']].describe()COPIAR CÓDIGO

O Pandas gerará, de forma simples e rápida, um conjunto de estatísticas descritivas dessas duas variáveis: a quantidade de valores, a média, o desvio padrão, o valor mínimo, os quartis da distribuição e o valor máximo.

|  | **Quilometragem** | **Valor** |
| --- | --- | --- |
| count | 197.000000 | 258.000000 |
| mean | 58278.421320 | 98960.513101 |
| std | 35836.733259 | 29811.932305 |
| min | 107.000000 | 50742.100000 |
| 25% | 27505.000000 | 70743.512500 |
| 50% | 55083.000000 | 97724.380000 |
| 75% | 90495.000000 | 124633.302500 |
| max | 119945.000000 | 149489.920000 |

Nos cursos de estatística da Alura nos aprofundamos no significado de cada uma dessas informações!

Com a função info() também podemos conseguir outras informações importantes. Por exemplo, acabamos de descobrir que temos menos ocorrências de Quilometragem do que de Valor.

dataset.info()COPIAR CÓDIGO

<class 'pandas.core.frame.DataFrame'> RangeIndex: 258 entries, 0 to 257 Data columns (total 7 columns): Nome 258 non-null object Motor 258 non-null object Ano 258 non-null int64 Quilometragem 197 non-null float64 Zero\_km 258 non-null bool Acessórios 258 non-null object Valor 258 non-null float64 dtypes: bool(1), float64(2), int64(1), object(3) memory usage: 12.5+ KB

Isso acontece pois, dentro do conjunto, temos alguns valores faltando (*missing*) ou nulos (*null*). Nos aprofundaremos mais no tratamento desses valores na última aula do curso. No próximo capítulo voltaremos a trabalhar com o Python, agora conhecendo as tuplas!