FAST TEST - MÓDULO 01

01 | Quando trabalhamos com o comando **pd.read_csv()** no Pandas, sabemos que isso nos possibilita a leitura dos “dados separados por vírgulas”, porém o padrão pode mudar, já que podemos ter arquivos .csv que são separados por ponto e vírgula (;). Qual relação abaixo se encaixa melhor para definir o parâmetro dentro do comando de leitura para que eu consiga ler os dados separados por “;”?

A) pd.read_csv(<arquivo.csv>, encoding=’utf-8)

B) pd.read_csv(<arquivo.csv>, sep=’;’)

C) pd.read_csv(<arquivo.csv>, sep=False)

D) pd.read_csv(<arquivo.csv>, encoding=’latin-1’)

E) pd.read_csv(<arquivo.csv>, sep=True)

A resposta correta é a opção B) `pd.read_csv(<arquivo.csv>, sep=';')`.

O parâmetro `sep` no comando `pd.read_csv()` permite especificar o caractere utilizado como separador entre os valores no arquivo CSV. Por padrão, o separador é a vírgula (`,`), mas quando o arquivo CSV utiliza um separador diferente, como ponto e vírgula (`;`), é necessário informar isso no parâmetro `sep`.

Portanto, a opção correta é `pd.read_csv(<arquivo.csv>, sep=';')`, onde você está indicando explicitamente que o separador utilizado no arquivo CSV é o ponto e vírgula.

---

02 | Qual seria uma maneira de inserir uma coluna nova na primeira posição em um Dataframe chamado “df”?

A) df.drop(0, “Nome da coluna”)

B) pd.options.display.float_format = False

C) df.insert(1, “Nome da coluna”)

D) df.insert(0, “Nome da coluna”)

E) df.drop(1, “Nome da coluna”)

A resposta correta é a opção D) `df.insert(0, "Nome da coluna")`.

Para inserir uma nova coluna no DataFrame na primeira posição, você pode usar o método `insert()` do Pandas. A sintaxe correta para isso é `df.insert(index, "Nome da coluna")`, em que "index" representa a posição na qual você deseja inserir a coluna.

No caso da opção D), `df.insert(0, "Nome da coluna")`, a coluna será inserida na primeira posição (índice 0), empurrando as colunas existentes para a direita.

As outras opções não são adequadas para inserir uma nova coluna na primeira posição:

A) `df.drop(0, "Nome da coluna")` remove a primeira linha do DataFrame ou a coluna especificada, não insere uma nova coluna.

B) `pd.options.display.float_format = False` define o formato de exibição dos números de ponto flutuante, não insere uma nova coluna.

C) `df.insert(1, "Nome da coluna")` insere a coluna na segunda posição (índice 1), não na primeira posição como solicitado.

E) `df.drop(1, "Nome da coluna")` remove a segunda linha do DataFrame ou a coluna especificada, não insere uma nova coluna.

---

03 | Quando usamos a função sample(n), o Dataframe é cortado em uma parte, sendo ‘n’ o número de linhas ou amostras que virão. De que forma esses dados aparecem para nós no Dataframe?

A) De forma a trazer os últimos ‘n’ elementos;

B) De forma totalmente aleatória;

C) De forma a selecionar ‘n’ colunas;

D) De forma pseudoaleatória;

E) De forma a trazer os primeiros ‘n’ elementos;

A resposta correta é a opção D) De forma pseudoaleatória.

Quando utilizamos a função `sample(n)` em um DataFrame, os dados são retornados de forma pseudoaleatória. Isso significa que a função utiliza um algoritmo para selecionar as linhas do DataFrame de maneira aproximadamente aleatória, mas o processo não é totalmente aleatório.

O método `sample(n)` retorna uma amostra de `n` linhas do DataFrame. Essas linhas são selecionadas de forma que cada uma tenha a mesma probabilidade de ser escolhida, mas a ordem em que são retornadas pode variar.

A opção A) "De forma a trazer os últimos 'n' elementos" não é correta, pois a função `sample(n)` não tem relação com a ordem dos elementos no DataFrame.

A opção B) "De forma totalmente aleatória" também não é correta, pois, como mencionado anteriormente, a seleção dos dados é feita de forma pseudoaleatória.

A opção C) "De forma a selecionar 'n' colunas" não está correta, pois a função `sample(n)` seleciona linhas, não colunas.

A opção E) "De forma a trazer os primeiros 'n' elementos" também não está correta, pois a função `sample(n)` não seleciona os primeiros elementos do DataFrame, mas sim uma amostra pseudoaleatória.

---

04 | Qual a função que modifica a legenda do eixo x da função plot() do Pandas?

A) set_ylabel()

B) name()

C) set_yname()

D) set_xlabel()

E) set_xname()

A resposta correta é a opção D) `set_xlabel()`.

A função `set_xlabel()` é usada para modificar a legenda do eixo x (eixo horizontal) em um gráfico gerado pelo método `plot()` do Pandas.

Ao chamar `set_xlabel("Texto da legenda")` em um gráfico, você estará definindo o texto que aparecerá como legenda do eixo x, fornecendo uma descrição apropriada para os dados representados nesse eixo.

As outras opções não são corretas para modificar a legenda do eixo x em um gráfico gerado pelo `plot()`:

A) `set_ylabel()` é usada para modificar a legenda do eixo y (eixo vertical), não do eixo x.

B) `name()` não é uma função relacionada à modificação da legenda do eixo x.

C) `set_yname()` não é uma função válida no Pandas para modificar a legenda do eixo x.

E) `set_xname()` também não é uma função válida no Pandas para modificar a legenda do eixo x.

---

5 | Qual das opções abaixo é uma maneira de acessar a última coluna de um Dataframe com seus respectivos valores?

A) df[df.columns[-1]]

B) df.columns[-1]

C) df[df.columns[0]]

D) df[df.columns[1]]

E) df.columns[0]

A resposta correta é a opção A) `df[df.columns[-1]]`.

Para acessar a última coluna de um DataFrame com seus respectivos valores, podemos utilizar `df[df.columns[-1]]`.

Explicando a opção A):

- `df.columns` retorna uma lista com os nomes das colunas do DataFrame.
- `1` é usado para acessar o último elemento da lista de colunas.
- `df[df.columns[-1]]` retorna a última coluna do DataFrame com seus respectivos valores.

As outras opções não são corretas para acessar a última coluna:

B) `df.columns[-1]` retorna apenas o nome da última coluna, mas não os valores.

C) `df[df.columns[0]]` retorna a primeira coluna do DataFrame com seus respectivos valores, não a última coluna.

D) `df[df.columns[1]]` retorna a segunda coluna do DataFrame com seus respectivos valores, não a última coluna.

E) `df.columns[0]` retorna apenas o nome da primeira coluna, mas não os valores.