**Data Visualization – Explorando com Seaborn**

Link para o drive desse curso:

<https://drive.google.com/drive/folders/19fh9DUEh_xbR6bDFguL3oTYZO9tVx4la>.

Biblioteca que usaremos:

- <http://pandas.pydata.org/pandas-docs/stable/>.

- <http://seaborn.pydata.org/introduction.html>.

1. **Aula 1 – Importando e Traduzindo Dados:**
   1. Aprendemos a importar a biblioteca pandas, os dados e a atribuir eles à uma variável.
   2. Traduzi ou não uma base de dados é sempre uma discussão importante a se ter. O ideal é conversar com a equipe e decidir se seria uma boa prática ou não a depender do objetivo e exibição desses dados.
      1. Para processos lúdicos e educativos, seria interessante traduzir, já pra publicação em artigos, por exemplo, não.
      2. Utilizando os dados.columns, conseguimos visualizar o index das colunas em forma de lista, ao invés de apenas renomear:

dados.columns



* + 1. Para renomear criamos um dict, e usamos o df.rename(columns=rename) como já sabemos fazer.
    2. Para renomear os campos de um DataFrame precisamos criar um dicionário, just like the columns, contendo o que queremos substituir para a substituição, aí usamos a função df.s.map(dict) e atribuímos essa series mapeada a ela mesma:

sim\_nao = {'No' : 'Não', 'Yes': 'Sim'}

gorjetas.sobremesa = gorjetas.sobremesa.map(sim\_nao)

gorjetas.head(1)



* + - 1. Esse mesmo processo se repete para os campos das outras variáveis:

dias = {'Sun': 'domingo', 'Sat': 'sabado', 'Thur': 'quinta', 'Fri': 'sexta'}

gorjetas.dia = gorjetas.dia.map(dias)

gorjetas.head(1)



refeição = {'Dinner': 'Jantar', 'Lunch': 'Almoço'}

gorjetas.refeicao = gorjetas.refeicao.map(refeicao)

gorjetas.head(1)



* 1. Para saber mais: Principais comandos
     1. Df.tail(): exibe as cinco últimas observações;
     2. Link para a documentação: <http://pandas.pydata.org/pandas-docs/stable/>.
  2. O que aprendemos:
     1. Importamos um arquivo chamado tips.csv e armazenamos numa variável chamada dados;
     2. Conhecemos os dados que vamos analisar;
     3. Por questões didáticas, decidimos traduzir nossa base de dados do inglês para o português;
     4. Armazenamos os dados traduzidos em uma nova variável chamada gorjetas.

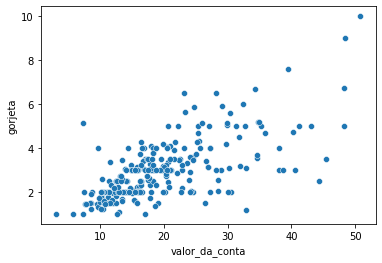
1. **Aula 2 – Comparando Distribuições de Maneira Explanatória:**
   1. A biblioteca feita para desenvolvimento e produção de gráficos é a matplotlib, mas nós utilizamos, hoje em dia, o pandas para análise, modificação e afins dos nossos dados, tendo essa biblioteca sido desenvolvida 10 anos depois da matplotlib.
      1. Nesse contexto a seaborn foi construída, sendo capaz de gerar gráficos bem mais sofisticados (usando a matplotlib por baixo dos panos) juntamente com o pandas e uma linguagem de bem mais alto nível do que a matplotlib poderia fazer.
      2. Importamos o seaborn como sns por convenção:

import seaborn as sns

* 1. Análise 1 - Valor da Conta e Gorjeta:
     1. Podemos gerar um gráfico scatterplot a partir das variáveis x = valor total da conta, com o y = valor da gorjeta, passando como a base de dados data = gorjetas. Dessa forma o gráfico será distribuído a partir do menor pro maior valor da conta no eixo x e da menor pra maior gorjeta no y:

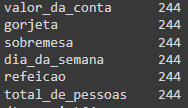
valor\_gorjeta = sns.scatterplot(x = 'valor\_da\_conta', y = 'gorjeta', data = gorjetas)

valor\_gorjeta



* + - 1. Podemos notar que parece ter uma progressão linear, onde conforme o valor da conta aumenta a gorjeta também aumenta.
    1. Para saber qual a contagem de observações não nulas usamos o count():

gorjetas.count()

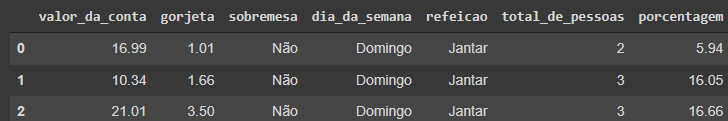


* 1. Podemos criar uma coluna com a porcentagem da gorjeta com relação ao valor total da conta:

gorjetas['porcentagem'] = (gorjetas.gorjeta / gorjetas.valor\_da\_conta) \* 100

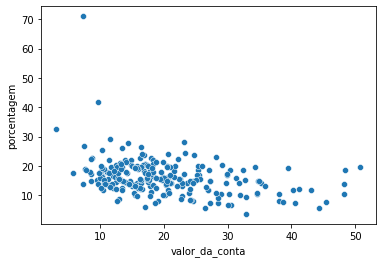
gorjetas.porcentagem = gorjetas.porcentagem.round(2)

gorjetas.head()



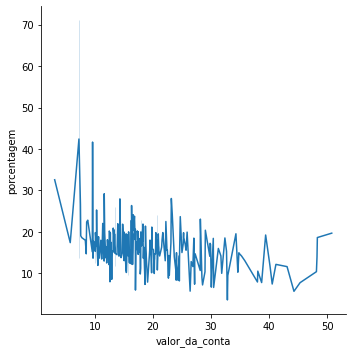
* + 1. E agora refazer o gráfico para ver se é proporcional a partir da porcentagem:

porcentagem\_conta = sns.scatterplot(x = 'valor\_da\_conta', y = 'porcentagem', data = gorjetas)



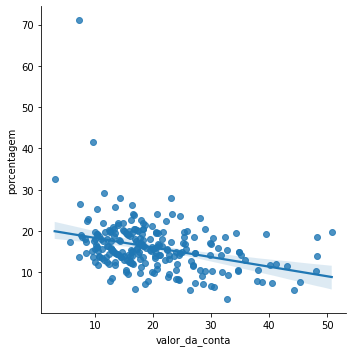
* + - 1. Podemos ver que, com relação ao outro gráfico, as contas com valores maiores deram mais gorjeta, mas, quando vemos esse gráfico, podemos concluir que não foram proporcionais, ou seja, visualmente o valor da conta não é proporcional ao valor da gorjeta.
  1. Podemos gerar o mesmo gráfico que acima, mas utilizando linhas ao invés de pontos a partir do sns.relplot(x = x, y = y, kind = ’line’, data = data). Se não passarmos o kind ele fará exatamente o mesmo gráfico que acima:

porcentagem\_conta\_linha = sns.relplot(x = 'valor\_da\_conta', y = 'porcentagem', kind = 'line', data = gorjetas)

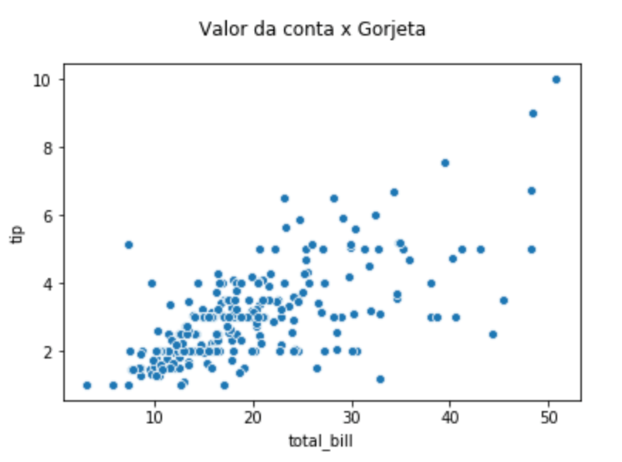


* + 1. E podemos ser ainda mais precisos na análise juntando ambos os gráficos com o sns.lmplot(x = x, y = y, data = data):

sns.lmplot(x = 'valor\_da\_conta', y = 'porcentagem', data= gorjetas)



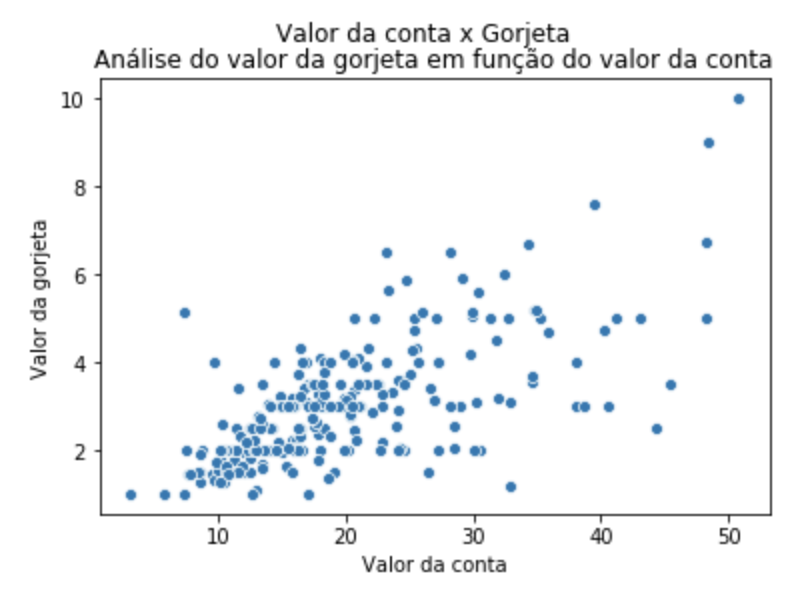
* + 1. Agora com esse gráfico podemos ter certeza que conforme aumenta o valor da conta, a porcentagem da gorjeta é menor, caindo o valor.
  1. Para saber mais:
     1. Podemos visualizar somente a imagem do nosso gráfico ou adicionar subtítulos, sem exibir o endereço da memória dele, com o get\_figure() e/ou o suptitle(‘titulo’), podendo ambos estarem juntos:
     2. primeiro\_plot.figure.suptitle('Valor da conta x Gorjeta'):



* + 1. Podemos também adicionar títulos com o set\_title(‘título’) e labels com o set(xlabel=’label’, ylabel=’label):

primeiro\_plot.set\_title('Análise do valor da gorjeta em função do valor da conta')

primeiro\_plot.set(xlabel='Valor da conta', ylabel='Valor da gorjeta')



* + 1. Podemos salvar o nosso gráfico com o img.savefig(‘nome.extensão’) e ela aparecerá no mesmo lugar onde fazemos upload de dados:

imagem = primeiro\_plot.get\_figure()

imagem.savefig('imagem.png')

