Relatório do Desafio Linx

Thiago Reis

10/01/2020

—

Analise de dados e Big Data

—

Estágio Linx

5

Output – Respostas das questões

```
          Questao 1

O faturamento total do mês foi de: R$ 19552328.32 reais

          Questao 2

O produto mais comprado do mês foi o de id: 626664333563363 com 74 vendas.

          Questao 3

Media total de vendas no mes por dia em RJ: 286.03
Media de vendas na sexta em RJ: 361.8
Media de vendas no sabado em RJ: 361.25
Media de vendas no domingo em RJ: 69.5
Media dos fins de semana em RJ: 264.18

          Questao 4

O numero de clientes que visitaram a pagina e compraram na loja fisica é: 592
O numero de clientes que visitaram a pagina e NAO compraram na loja fisica é: 5321
Ou seja, 10.01% visitaram o site e compraram na loja fisica.
Enquanto que 60% dos usuarios identificaveis (id's não nulos) visitaram o site e compraram online.

          Questao 5

Numero de clientes que acessaram o site, porem nao compraram na loja online: 2383
Supondo que 15.01% das pessoas que abandonaram o carrinho participem da campanha, e que todas gastem uma media de 468.06, com o
desconto de 20% fica uma media de 374.45 por compra.
A campanha na loja fisica terá um faturamento de: 133936.31
```

Questão 1- Qual foi o faturamento total no período?

```
# Codigo para questao 1


def soma_faturamento_offline():
    soma = 0
    for i in range(1,quantidade_maxima_offline+1):
        dataframe_i = offline_sales[offline_sales['quantity']==i] # D
        soma += sum(dataframe_i.price)*(i)                        # A
    return (soma)

def soma_faturamento_online():
    soma = 0
    for i in range(1,quantidade_maxima_online+1):
        dataframe_i = online_orders[online_orders['quantity']==i] # M
        soma += sum(dataframe_i.price)*(i)
    return (soma)

def faturamento_total(): # O faturamento total eu deduzo que seja a s
    return (soma_faturamento_online() + soma_faturamento_offline())
```

Para a questão 1 eu fiz duas funções, a lógica é percorrer o dataframe_i e somar a coluna 'price', para cada soma multiplicar pelo valor de i (que seria a quantidade vendida). A tabela abaixo representa o dataframe_i, quando i = 2: -

Tabela 1 dataframe_i

```
In [128]: offline_sales[offline_sales['quantity']==2]
```

Out[128]:

	date	state	store_id	sale_id	off_product_id	quantity	price	customer_id
79	2018-08-01	RS	3561636	303433383834633	623963303161316	2	138.0	64303065363239616431
116	2018-08-01	RJ	3630316	313237653263646	306666396162363	2	6390.0	62373833313031383163
125	2018-08-01	RS	3639343	323763393562366	333135626136306	2	169.0	30323162333538633032
133	2018-08-01	RS	3639343	386133336232336	393739626339643	2	119.0	30366664643862386332
182	2018-08-01	SP	3639373	386136343931633	393635616566393	2	298.0	None
...	...	...	...	...	...	...	...	...
28941	2018-08-31	RJ	3964313	633330663039643	353033396564613	2	238.0	62343236643264346264
28949	2018-08-31	MA	3966336	376635353463326	323531386239613	2	29.0	None
28964	2018-08-31	MA	3966336	653261353461343	653730303034323	2	98.0	None
29112	2018-08-31	RJ	6234643	303231636238383	373631326161383	2	598.0	63636665643236333834
29173	2018-08-31	RJ	6236393	393631386366323	643533626135633	2	198.0	64663738323561393836

531 rows × 8 columns

A tabela mostra apenas as vendas com ‘quantity’ = 2; A lógica é somar a coluna dos preços e multiplicar pela sua quantidade.

Nessa questão, minha maior dificudade foi aprender a usar a biblioteca Pandas de forma eficiente para organizar os dados, e reaprender Python (já tinha visto há 2 anos, porém só o básico)

Questão 2 – Qual o produto mais comprado onine?

```
# Codigo para questao 2

def produto_mais_comprado_online():
    lista=[]
    quantity_e_id = online_orders[['quantity','on_product_id']]
    for id_produto in online_orders.on_product_id:
        lista.append([sum(quantity_e_id[quantity_e_id['on_product_id']==str(id_produto)].quantity),id_produto])

    return lista


lista_mais_comprado=produto_mais_comprado_online()
```

Inicialmente, eu pensei em algo mais intuitivo, minha função simplesmente filtrava a tabela em 'on_product_id', e retornava a moda (com o método mode()), ou seja, retornava o produto que mais se repetiu na tabela. Porém eu não levei em consideração a quantidade do produto por pedido, e então eu refiz desta maneira acima. A tabela 'quantity_e_id' mostra apenas as colunas 'quantity' e 'on_product_id', o laço percorre todos os id's dos produtos, e os adiciona em uma lista, junto com sua quantidade vendida. Depois na hora de printar o resultado, eu chamo o método max(), que retorna o maior valor da lista, ou seja, a maior quantidade vendida do produto.

```
print("          Questao 2\n")
print("O produto mais comprado do mês foi o de id: " + str(max(lista_mais_comprado)[1]) + " com " + str(max(lista_mais_comprado)
```

Figura 1 Método max() na hora do print

Minhas maiores dificuldades foram: conseguir retornar a maior quantidade vendia, junto com o id, e usar a biblioteca Pandas (algo que eu nunca tinha visto antes). A tabela abaixo mostra os 10 itens mais vendidos:

```
In [155]: def produto_mais_comprado_online():
              lista = pd.DataFrame()
              quantity_e_id = online_orders[['quantity','on_product_id']]
              i = 0
              for id_produto in online_orders.on_product_id:
                  lista.set_value(id_produto,'quantity',sum(quantity_e_id[quantity_e_id['on_product_id']==str(id_produto)].quantity))

              return lista
          print(produto_mais_comprado_online().sort_values(by="quantity",ascending = False).head(10))
```

	quantity
626664333563363	74.0
376531636530353	51.0
656136316465643	50.0
396464373131666	47.0
656662626334303	45.0
343033316438393	45.0
393665363031353	45.0
316634393832653	44.0
633336383230346	41.0
613862646363323	40.0

Questão 3 – Cariocas gostam de comprar no fim de semana?

```
# Codigo para questao 3

def estado_fimDeSemana_vende(estado):

    data_frame_state_offline = offline_sales[offline_sales['state']==str(estado)]['date']   # Filtrar para datas de compras apen
    state_offline_frequencia = data_frame_state_offline.value_counts() # Dataframe das frequencias de compra no mes, desde o dia

    fim_de_semana = pd.DataFrame({'Sexta' : ["2018-08-03", "2018-08-10", "2018-08-17", "2018-08-24", "2018-08-31"],
                                  'Sabado' : ["2018-08-04","2018-08-11","2018-08-18","2018-08-25","0"],
                                  'Domingo' : ["2018-08-05","2018-08-12","2018-08-19","2018-08-26","0"]})
    media_sexta = round(state_offline_frequencia.loc[fim_de_semana["Sexta"]].mean(),2)
    media_sabado = round(state_offline_frequencia.loc[fim_de_semana["Sabado"]].mean(),2)
    media_domingo = round(state_offline_frequencia.loc[fim_de_semana["Domingo"]].mean(),2)
    print("Media total de vendas no mes por dia em " + str(estado) + ": %.2f" % round(state_offline_frequencia.mean(),2) + "\n" +
          "Media de vendas na sexta em " + str(estado) + ": " + str(media_sexta) + "\n" +
          "Media de vendas no sabado em " + str(estado) + ": " + str(media_sabado) + "\n"+
          "Media de vendas no domingo em " + str(estado) + ": " + str(media_domingo) + "\n"+
          "Media dos fins de semana em " + str(estado) + ": " + str(round((media_sexta + media_sabado + media_domingo)/3,2)) + "

print("          Questao 3\n")
estado_fimDeSemana_vende("RJ")
```

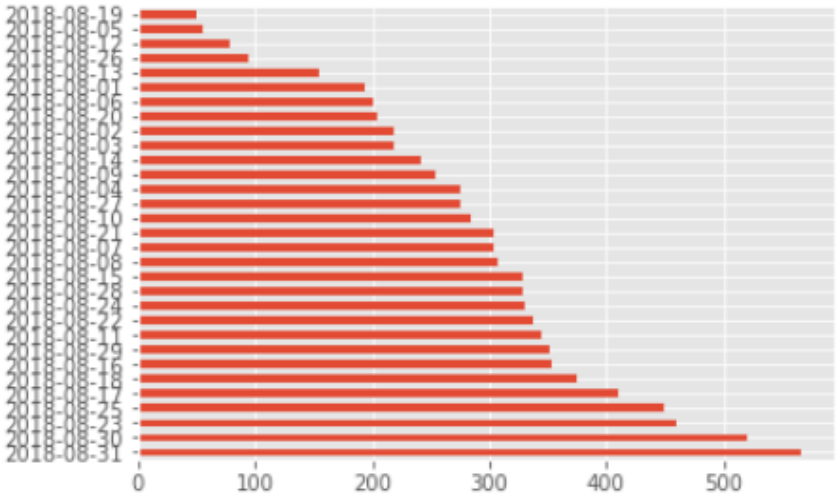
Para a questão 3, eu tentei fazer a função funcionar de forma genérica para qualquer estado informado, além disso fiz um dataframe específico para o mês de agosto de 2018, contendo todos os dias do final de semana. O dataframe 'state_offline_frequencia' retorna uma tabela com todos os dias do mês, informando quantas vendas foram realizadas em cada dia. Após isso, é calculado a média de vendas de cada dia do fim de semana, utilizando o método mean().

Como a média do fim de semana foi menor que a média por dia (diferença de 20 reais), eu diria que os cariocas não costumam comprar no fim de semana.

O gráfico abaixo mostra a media de vendas por dia no estado do Rio de Janeiro.

In [161]:
```
data_frame_state_offline = offline_sales[offline_sales['state']=="RJ"]['date']
state_offline_frequencia = data_frame_state_offline.value_counts()
plt.style.use('ggplot')
state_offline_frequencia.plot.barh()
```

Out[161]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x973ddb0>
```



Em questão de programação, não tive muita dificuldade com essa questão, minha dificuldade foi em tentar descobrir quais critérios seriam usados, se eram quantas compras foram feitas pelos cariocas, quantos produtos foram comprados, ou se eu teria que usar o dataset de compras online (apesar de não haver uma coluna 'state', acho que seria possível identificar de onde foi feita a compra pelo 'customer_id' de compras offline).

Questão 4 – É comum escolher online e terminar a compra na loja física?

```
# Codigo para questao 4

offline_sales_filtrado = offline_sales.dropna().customer_id.drop_duplicates()       #  }
online_pageviews_filtrado = online_pageviews.dropna().customer_id.drop_duplicates() #  }==== Id's sem valores nulos e repetidos
online_orders_filtrado =  online_orders.dropna().customer_id.drop_duplicates()      #  }

                          # Visitaram a pagina e:
optaram_fisica = 0        # Optou em comprar na loja fisica
nao_optaram_fisica = 0    # Nao optou em comprar na loja fisica

optaram_online = 0        # Optou em comprar online
nao_optaram_online = 0    # Nao optou em comprar online
```

Neste trecho, eu precisei filtrar todos os json utilizados, principalmente o 'online_pageviews' pois tinha muitos id's de usuarios nulos, que na minha função, não teriam utilidade, além de remover id's repetidos.

```
def soma_optar_loja_fisica_e_online():  # Altera o valor das variaveis acima, atribuindo a soma apenas dos clientes que visitaram
    global optaram_fisica
    global nao_optaram_fisica
    global optaram_online
    global nao_optaram_online
    global nao_optou_nenhuma

    for i in online_pageviews_filtrado:  # i assumira todos os valores customer_id
        if online_orders_filtrado.isin([i]).any().any():     # Retorna True se i (customer_id) pertence a offline_sales
            optaram_online+=1
        if not online_orders_filtrado.isin([i]).any().any():
            nao_optaram_online+=1
        if offline_sales_filtrado.isin([i]).any().any():
            optaram_fisica+=1
        if not offline_sales_filtrado.isin([i]).any().any():
            nao_optaram_fisica+=1
        if not offline_sales_filtrado.isin([i]).any().any() and not online_orders_filtrado.isin([i]).any().any():
            nao_optou_nenhuma+=1

soma_optar_loja_fisica_e_online()

print("          Questao 4\n")
print("O numero de clientes que visitaram a pagina e compraram na loja fisica é: " + str(optaram_fisica))
print("O numero de clientes que visitaram a pagina e NAO compraram na loja fisica é: " + str(nao_optaram_fisica))
percentual_optaram_fisica = optaram_fisica/(optaram_fisica+nao_optaram_fisica)*100
percentual_optaram_online = optaram_online/(optaram_online+nao_optaram_online)*100
print("Ou seja, %.2f" % round(percentual_optaram_fisica,2) + "% visitaram o site e compraram na loja fisica.")
print("Enquanto que %2.f" % round(percentual_optaram_online,2) + "% dos usuarios identificaveis (id's não nulos) visitaram o site
```

Para essa questão pensei o seguinte: a função percorrerá todos os 'customer_id' do 'online_pageviews_filtrado', para cada um que pertencer ao 'online_orders_filtrado', 'optaram_online' é incrimentado em 1, a mesma coisa para o 'offline_sales_filtrado'. Depois apenas printo os dados.

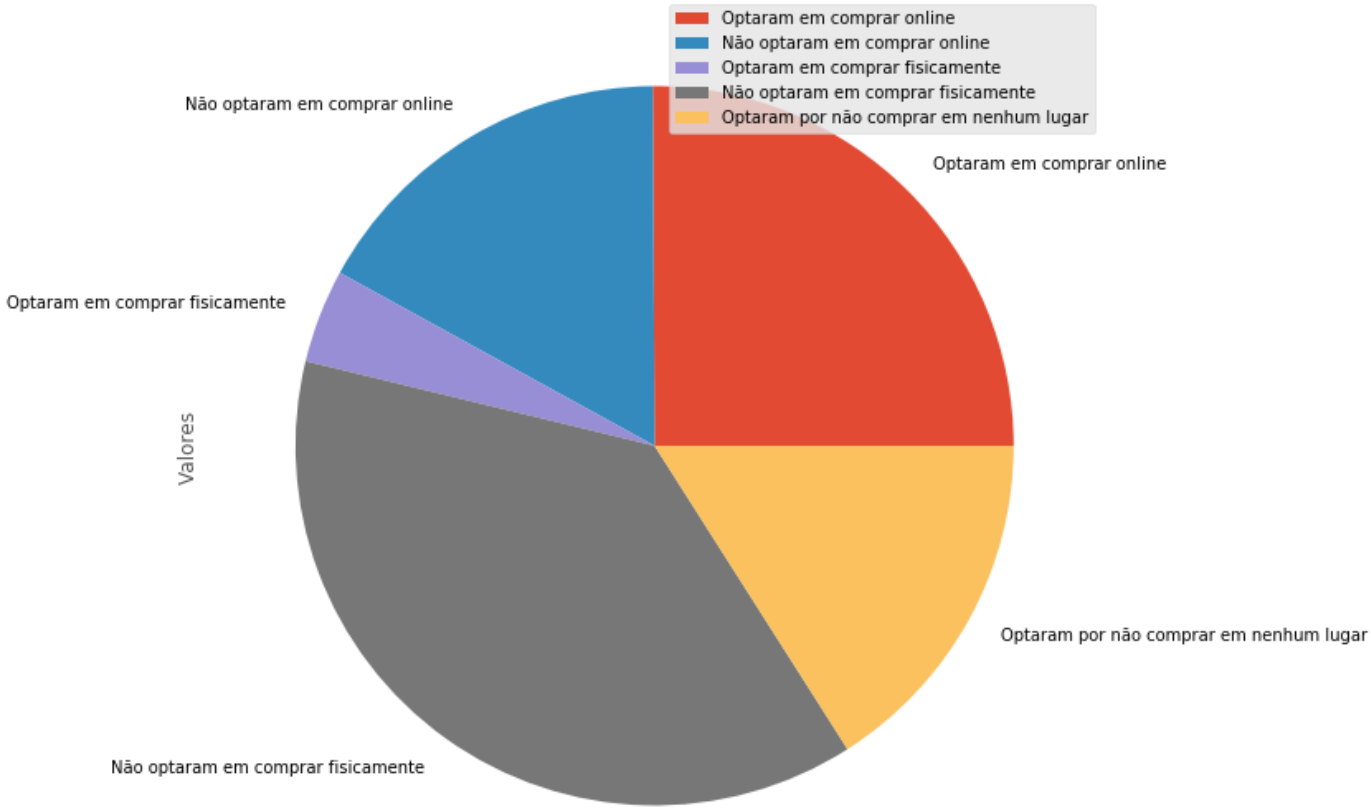
Com certeza minha maior dificuldade começou nessa questão, trabalhar com o json 'online_pageviews' foi muito dificil.

```
# Importando os arquivos json para o codigo

with open('offline_sales.json') as f:
    offline_sales = pd.DataFrame(json.loads(line) for line in f)

with open('online_orders.json') as f:
    online_orders = pd.DataFrame(json.loads(line) for line in f)
```

Foi assim que eu importei os outros json para o código, então para o 'online_pageviews' eu fiz a mesma coisa, porém o programa não compilava (erro de memória). Eu comecei a achar que o problema era do computador, pois eu usei um notebook pouco potente. Porém quando testei em outro computador mais potente, acontecia o mesmo. Então eu tinha que achar uma forma de lidar com esse arquivo grande. Eu fiz o seguinte:

```
customer_id_array = []
with open('online_pageviews.json') as f:
    for line in f:
        customer_id_array.append(json.loads(line)['customer_id'])
online_pageviews = pd.DataFrame(customer_id_array,columns = ['customer_id'])
```

Eu criei o dataframe apenas com os valores da coluna 'customer_id', pois as outras colunas não me pareciam úteis para resolver os problemas dados. Isso economizou muita mémoria, e consegui trabalhar com ele.



O gráfico acima mostra a proporção de escolha dos clientes que visitaram a página.

Questão 5 - O time de marketing desta rede quer fazer uma campanha oferecendo um cupom de 20% nas compras de loja física para quem visitou o site e abandonou um carrinho com produtos. Estime o resultado dessa campanha.

A questão 5, na minha opnião, a mais interpretativa, e que mais necessitava de suposições.

```
# Codigo para questao 5

def visitaram_mas_nao_compraram(): # Clientes que acessaram o site, porem nao compraram na loja online
    return nao_optaram_online



print("          Questao 5\n")
print("Numero de clientes que acessaram o site, porem nao compraram na loja online: "+str(visitaram_mas_nao_compraram()))

media_precos_loja_fisica = round(offline_sales['price'].mean(),2)
percentual_estimado = round(percentual_optaram_online+5,2)

print("Supondo que " + str(percentual_estimado) + "% das pessoas que abandonaram o carrinho participem da campanha, e que todas
numero_de_pessoas = round(nao_optaram_online*percentual_estimado/100,2)
print("A campanha na loja fisica terá um faturamento de: "+str(round(numero_de_pessoas*media_precos_loja_fisica*0.8,2)))
```

Nessa questão, não tive que criar algoritmos, usei apenas os dados das outras questões. Minha estimação da campanha teve a seguinte lógica: como a campanha visava as pessoas que visitaram o site e abandonaram o carrinho, o percentual que usei foi das pessoas que acessaram o site e compraram na loja física, ou no meu código, 'percentual_optaram_fisica". Usei esse percentual porque ele diria aproximadamente quantas pessoas iriam ver o site e comprar na loja física. Como havia o desconto de 20%, aumentei em 5% o numero estimado de participantes. O 5% foi arbitrário da minha parte, pois sinceramente não entendo de marketing, mas acredito que não seja um absurdo considerar isso. Para o faturamento, o numero de pessoas é o percentual_estimado x não_optaram_online, e multipliquei pela média dos preços de itens comprados.

Bônus. O que mais de interessante tem nestes dados?


```
{
   "date": "2018-08-01",
   "visitor_id": "34303165316239643163326l3",
   "deviceType": "mobile",
   "order_id": "356664366366353",
   "on_product_id": "313562333039323",
   "quantity": 1,
   "price": 629.0,
   "customer_id": "63393337303931353431"
}
```


Como o tipo de dispotivo do usuário é fornecido, seria possível saber a proporção de acessos por dispositivo. Isso seria útil para saber se é rentável aprimorar página para tais dispositivos, produzir aplicativos e etc.

Com os dados recebidos é possível:

- Prever quais produtos serão comprados por cada usuário.
- Prever qual será o faturamento do mês seguinte.
- Estimar outras campanhas, como por exemplo: oferecer ofertas em estados que não vendem muito, produtos que não vendem muito, usuários que não compram há um tempo, etc.
- Necessário melhorar os servidores? (baseado no número de acessos)
- Quais dias o site tem mais acesso?
- Quais lojas filiais vendem mais? Quais vendem menos?
- Observar a demanda em cada estado
- Observar a oferta em cada estado

Considerações finais e referências

Só de fazer esse projeto acredito que já foi MUITO importante para mim, aprendi mais que em um semestre inteiro da faculdade, além de ter sido o primeiro contato com uma área importante como a ánalise de dados, algo que não verei no curso. Consegui ver que quando eu quero algo eu consigo aprender muito. Estou satisfeito com o resultado, consegui aprender muito em poucos dias, talvez algumas respostas estejam erradas, ou não usei os parametros certos, ou interpretei errado, mas mesmo assim estou satisfeito. Obrigado.

Referencias usadas:
<introdução ao pandas>
https://medium.com/data-hackers/uma-introdu%C3%A7%C3%A3o-simples-ao-pandas-1e15eea37fa1
<documentaçao pandas>
https://pandas.pydata.org/pandas-docs/version/0.21/index.html
Muitas dúvidas esclarecidas no StackOverflow 😊😊
https://stackoverflow.com/