Análises Estatísticas na compra e venda de veículos usados

Jonatas Halliday Sant Anna do Nascimento

- Análise exploratória de Dados

O presente relatório tem como principal foco a visualização e explicação da análise exploratória de dados do processo seletivo Lighthouse Programa de Formação em Dados. Inicialmente o primeiro passo foi ver quais colunas do dataset tinham valores nulos.

Para tal, utilizei o seguinte script para visualizar a porcentagem de dados faltantes no dataset inteiro:

```
[ ] total_cells = np.product(cars_train.shape) # faz a multiplicacao de cada feature usando a dimensao do df
    total_missing_values = cars_train.isnull().sum().sum() # conta os dados ausentes
    #percent of missing data
    percent = (total_missing_values/total_cells) * 100
    print(percent)

    17.40397885156935
```

Isso nos mostra um valor de aproximadamente 17% do dataset original tem valores faltantes.

Em seguida foi feito uma análise parecida mas analisando a porcentagem de dados faltantes por coluna:

```
missing_percentage = cars_train.isnull().mean() * 100
print(missing_percentage)
```

```
id                          0.000000
num_fotos                   0.000000
marca                       0.000000
modelo                      0.000000
versao                      0.000000
ano_de_fabricacao           0.000000
ano_modelo                  0.000000
hodometro                   0.000000
cambio                      0.000000
num_portas                  0.000000
tipo                        0.000000
blindado                    0.000000
cor                         0.000000
tipo_vendedor               0.000000
cidade_vendedor             0.000000
estado_vendedor             0.000000
anunciante                  0.000000
entrega_delivery            0.000000
troca                       0.000000
elegivel_revisao            0.000000
dono_aceita_troca          25.899135
veiculo_único_dono         64.768118
revisoes_concessionaria    68.996755
ipva_pago                  33.548540
veiculo_licenciado         46.234451
garantia_de_fábrica        85.245403
revisoes_dentro_agenda     80.022985
veiculo_alienado          100.000000
preco                       0.000000
dtype: float64
```

Em seguida foi feito o tratamento desses dados faltantes da seguinte forma:

Na coluna 'dono_aceita_troca' preenchi os valores NaN com a string "Não aceita troca". Essa escolha se deu pelo fato de **todos** os valores que não era preenchido com 'Aceita troca' serem NaN e portanto assumi que os valores NaN era de donos que não aceitam trocas.

O mesmo raciocínio se deu para a coluna 'veiculo_único_dono' e a coluna 'revisoes_concessionaria' na qual os valores NaN foram preenchidos com a string 'Mais que um dono' e 'Nem toda a revisão foi feita em concessionária' respectivamente. Para a coluna 'ipva_pago', 'veiculo_licenciado', 'garantia_de_fábrica' e 'revisoes_dentro_agenda' foram preenchidas usando a mesma lógica. Para a coluna 'veiculo_alienado' optei por remover a coluna, visto que todas as linhas tinham apenas valores NaN, gerando apenas ruído inútil ao dataset.

Em seguida foi necessário fazer uma limpeza na coluna 'estado_vendedor' visto que havia dado redundante na mesma célula,

com o estado escrito por extenso e em seguido da sigla do estado. Fiz essa limpeza como mostrado abaixo e optei por colocar a coluna apenas com a sigla do estado em questão:

```
cars_train['estado_vendedor'] = cars_train['estado_vendedor'].astype('str')
cars_train['estado_vendedor']
```

```
0            São Paulo (SP)
1         Minas Gerais (MG)
2            São Paulo (SP)
3            São Paulo (SP)
4       Rio de Janeiro (RJ)
              ...
29579          Goiás (GO)
29580         Paraná (PR)
29581          Bahia (BA)
29582        São Paulo (SP)
29583        São Paulo (SP)
Name: estado_vendedor, Length: 29584, dtype: object
```

```
[ ] cars_train['estado_vendedor'] = cars_train['estado_vendedor'].str.split('(').str[1]
```

```
[ ] cars_train['estado_vendedor'] = cars_train['estado_vendedor'].str.replace(')','')
```

```
[ ] cars_train['estado_vendedor']
```

```
0        SP
1        MG
2        SP
3        SP
4        RJ
        ..
29579    GO
29580    PR
```

- Análise estatística dos dados

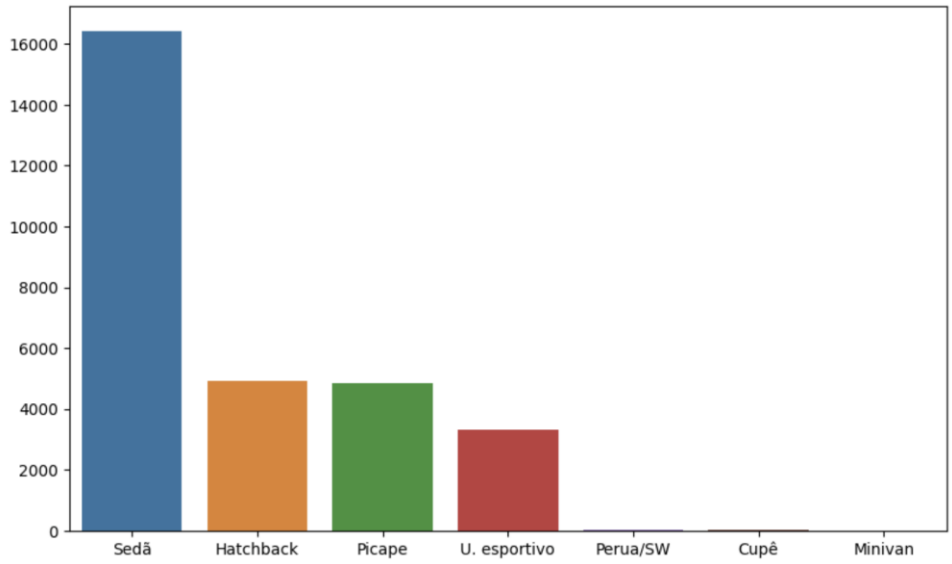
Inicialmente optei por mostrar um gráfico de barras mostrando o tipo do carro em busca de responder a pergunta (a) do desafio:

a. Qual o melhor estado cadastrado na base de dados para se vender um carro de marca popular e por quê?

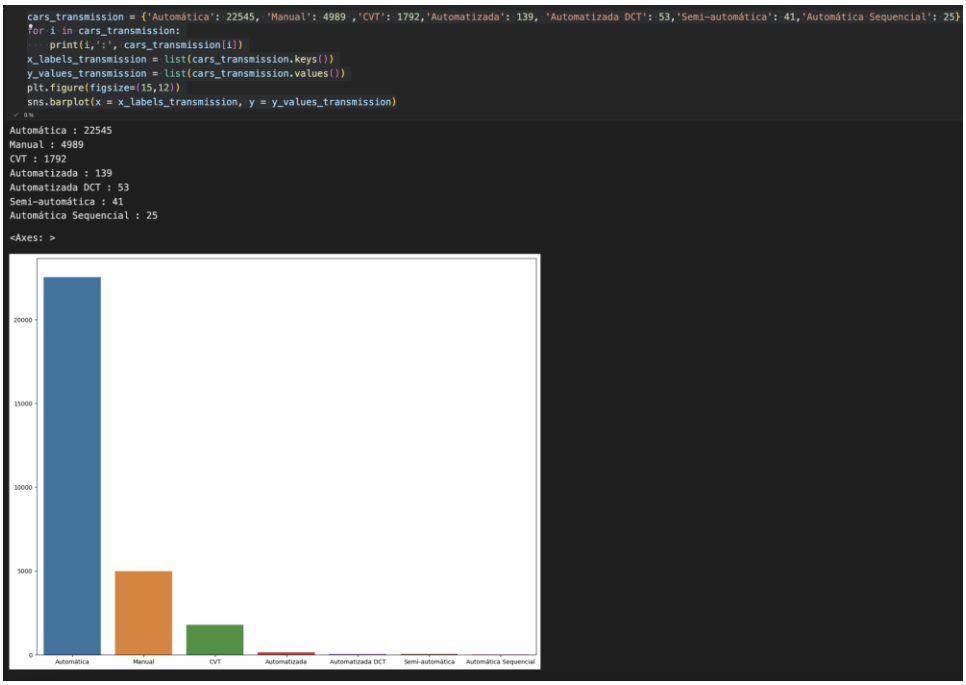
Abaixo, o script utilizado para gerar o gráfico e o gráfico em questão:

```
cars_labels = {'Sedã':16429, 'Hatchback':4924 ,'Picape':4849,'U. esportivo':3322, 'Perua/SW':
for i in cars_labels:
    print(i,':', cars_labels[i])
x_labels = list(cars_labels.keys())
y_values = list(cars_labels.values())
plt.figure(figsize=(10,6))
sns.barplot(x = x_labels, y = y_values)
plt.show()
plt.savefig("tipocarro.png")
```

```
Sedã : 16429
Hatchback : 4924
Picape : 4849
U. esportivo : 3322
Perua/SW : 27
Cupê : 26
Minivan : 7
```



Para responder a (a) foi necessário definir o que é um carro popular, e optei por descobrir a preferência do câmbio no presente dataset como primeira investigação. Para verificar, optei por visualizar o gráfico de forma semelhante a coluna 'cambio'.



Observa-se uma grande preferência por carros automáticos.

Calculando o percentual, observamos que cerca de 76% do dataset é composto de carro automático, conforme o trecho do código abaixo.

Para analisar as marcas populares, optei por fazer um value_counts() da coluna marca e usei esse link para confirmar que realmente são as marcas mais populares atualmente no Brasil. Obtive as seguintes marcas:
VOLKSWAGEN,CHEVROLET,TOYOTA,HYUNDAI,FIAT,PEUGEOT,FORD,RENAULT, HONDA,CITROËN,NISSAN e JEEP

Condizentes com a pesquisa feita. Optei também por fazer um recorte da seguinte forma: Peguei apenas carros cujo preço são menores que a

média da coluna 'preco' e que pertença a alguma das marcas citadas acima. Em seguida fiz um agrupamento pela coluna estado_vendedor e fiz um count para contar a marca mais vendida por estado, como mostrado abaixo:

```
count_brands_state = popular_cars.groupby(['estado_vendedor','marca']).size().reset_index(name = 'count')
brand_state = count_brands_state.groupby('estado_vendedor')['count'].idxmax()
brand_sells_most = count_brands_state.loc[brand_state]
print(brand_sells_most)
brand_sells_most = pd.DataFrame(brand_sells_most)
```

✓ 0.0s

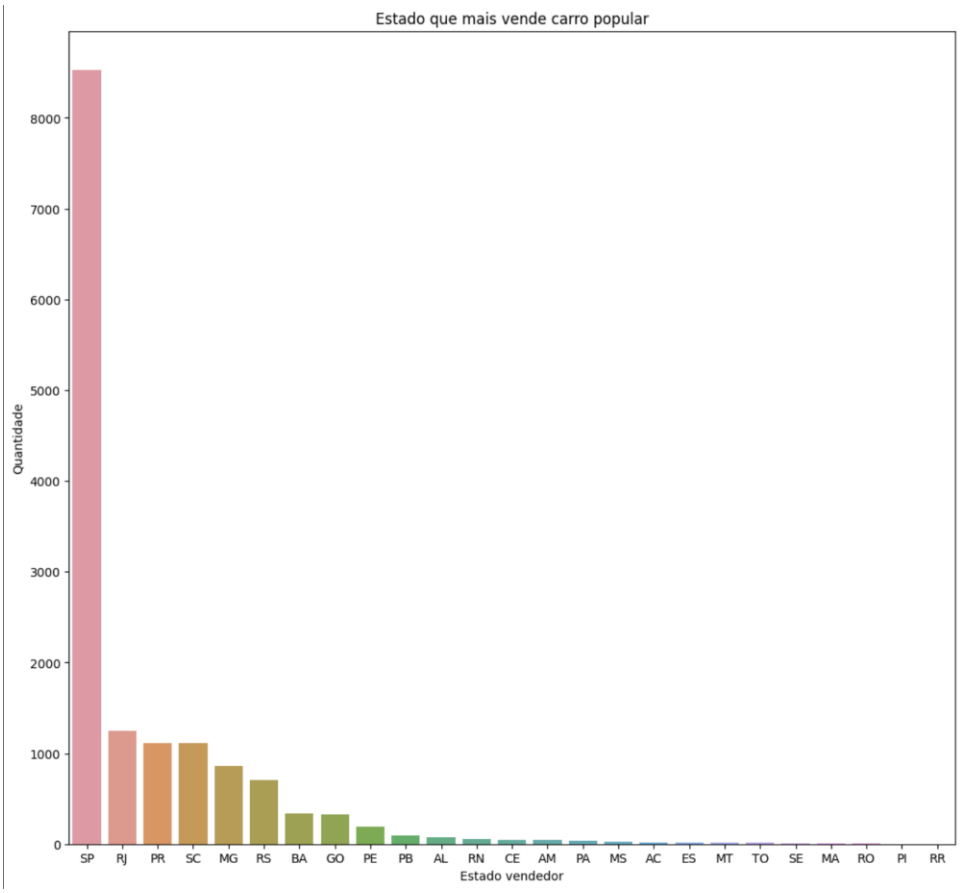
	estado_vendedor	marca	count
1	AC	CITROËN	4
17	AL	VOLKSWAGEN	23
19	AM	FIAT	16
26	BA	CHEVROLET	75
38	CE	CHEVROLET	12
47	ES	CHEVROLET	6
64	GO	VOLKSWAGEN	82
66	MA	PEUGEOT	4
78	MG	VOLKSWAGEN	175
80	MS	FIAT	9
90	MT	PEUGEOT	8
92	PA	CHEVROLET	11
98	PB	CHEVROLET	36
118	PE	VOLKSWAGEN	37
119	PI	CHEVROLET	1

Optei por fazer um plot com gráfico em barros utilizando o script:

```
#popular_cars_state_df.reset_index(inplace = True)
plt.figure(figsize=(13,12))
sns.barplot(x = popular_cars_state_df['estado_vendedor'], y = popular_cars_state_df['count'])
plt.title("Estado que mais vende carro popular")
plt.xlabel("Estado vendedor")
plt.ylabel("Quantidade")
plt.show()
```

✓ 0.1s

E obtive o seguinte gráfico:

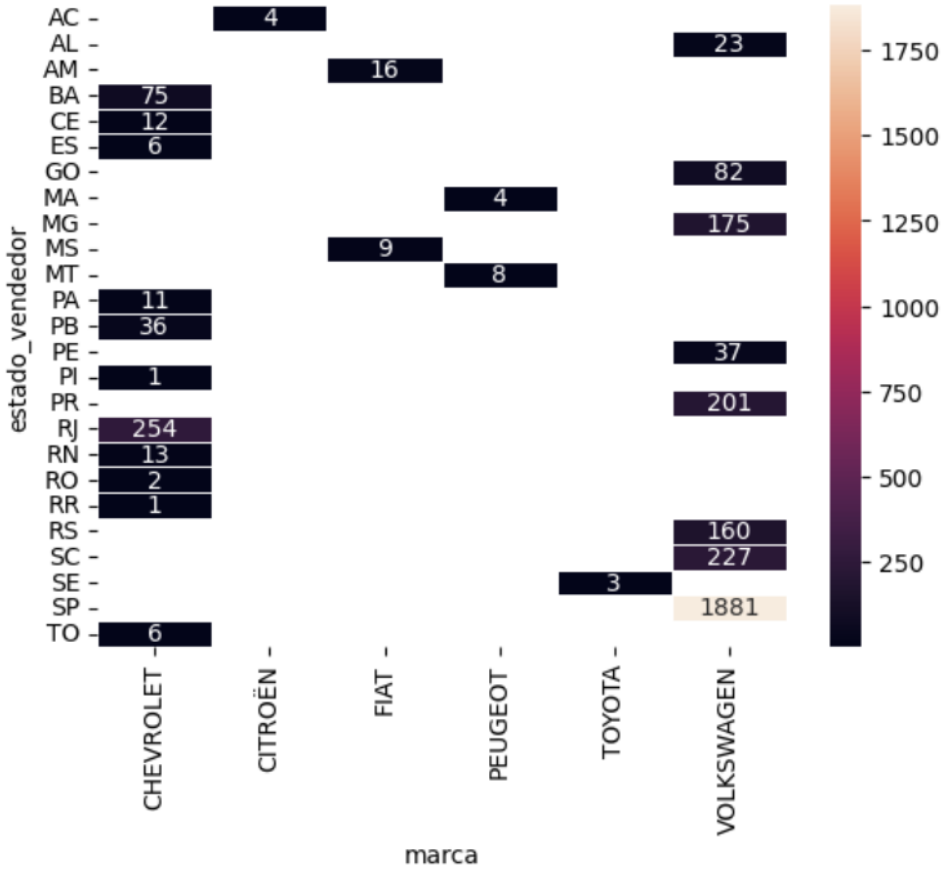


Dessa forma, vemos que o melhor estado cadastrado para vender um carro popular é SP. Optei por utilizar um heatmap com o estado e as marcas que mais vendem por estado e o fiz da seguinte forma:

```
pivot_df = brand_sells_most.pivot(index = 'estado_vendedor',columns = 'marca',values = 'count')
```
0.0s

```
sns.heatmap(data = pivot_df, annot = True,fmt = ".0f" ,linewidths= 0.5)
```
0.2s

Obtendo o seguinte mapa:



Para a pergunta (b) Qual o melhor estado para se comprar uma picape com transmissão automática e por quê?

Olhando agora, pela perspectiva de quem compra o carro, é necessário procurar pelos menores preços.

```
#1 filtra entre todo o nosso universo: no caso somente as picapes

picapes  = cars_train[cars_train['tipo'] == 'Picape']

picapes_automatic = picapes[picapes['cambio'] == 'Automática']

picapes_automatic = picapes_automatic[['marca','modelo','estado_vendedor', 'tipo', 'cambio','preco']].sort_values(by = 'preco',ascending = True)

picapes_automatic
```

	marca	modelo	estado_vendedor	tipo	cambio	preco
7696	VOLKSWAGEN	NEW BEETLE	SP	Picape	Automática	1.595320e+04
28497	TOYOTA	ETIOS	SP	Picape	Automática	2.566465e+04
24028	HYUNDAI	VELOSTER	RJ	Picape	Automática	2.672972e+04
4658	DODGE	DAKOTA	SP	Picape	Automática	2.932715e+04
690	VOLKSWAGEN	GOL	PR	Picape	Automática	3.347473e+04
...	...	...	...	...	...	...
5522	PORSCHE	718	SP	Picape	Automática	9.447675e+05
6846	BMW	Z4	SP	Picape	Automática	9.668446e+05
21827	PORSCHE	718	SP	Picape	Automática	1.028431e+06
2271	RAM	2500	SP	Picape	Automática	1.154360e+06
9175	PORSCHE	911	SC	Picape	Automática	1.359813e+06

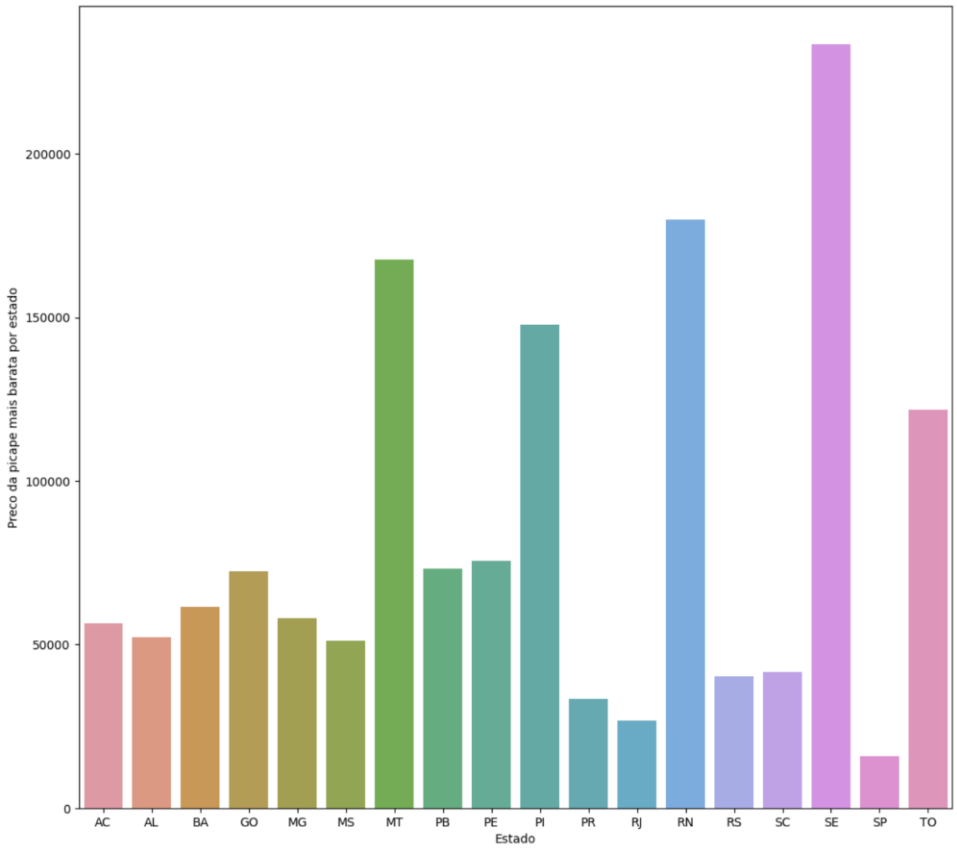
3300 rows × 6 columns

Usando o trecho abaixo apenas filtrei por picapes na coluna 'tipo' e automático na coluna 'cambio'. E utilizei o seguinte script para tal:

```
plt.figure(figsize=(13,12))
sns.barplot(x = cars_low['estado_vendedor'], y = cars_low['preco'])
#plt.title("Estado que mais vende carro popular")
plt.xlabel("Estado")
plt.ylabel("Preco da picape mais barata por estado")
plt.show()
```

✓ 0.1s

Obtive o seguinte gráfico:



Com isso vemos que o Estado mais barato para se comprar uma picape com transmissão automática é o estado de SP com o valor de 15953.204811 reais da marca VW, seguido do RJ com 26729.716235 reais da marca HYUNDAI.

Para a pergunta (c) Qual o melhor estado para se comprar carros que ainda estejam dentro da garantia de fábrica e por quê?

Para responder a essa pergunta apenas filtrei por carros que ainda estejam na garantia:

```
cars_in_warranty = cars_train[cars_train['garantia_de_fábrica'] == 'Garantia de fábrica']
cars_in_warranty
```

	id	num_fotos	marca	modelo	versao	ano_de_fabricacao	ano_modelo	hodometro	cambio	num_portas
4	338980975753200343894519909855598027197	8.0	SSANGYONG	KORANDO	2.0 GLS 4X4 16V TURBO DIESEL 4P AUTOMÁTICO	2013	2015	71491.0	Automática	4
8	271933559722390902682872823440667919 59	8.0	VOLKSWAGEN	UP	1.0 TSI HIGH UP 12V FLEX 4P MANUAL	2017	2018	39987.0	Manual	4
17	22623637632558238619093107555320587591	8.0	VOLKSWAGEN	POLO	1.0 200 TSI COMFORTLINE AUTOMÁTICO	2019	2019	35346.0	Automática	4
19	11829522559733270521825482734325810 2397	8.0	CHEVROLET	ONIX	1.0 TURBO FLEX PLUS PREMIER AUTOMÁTICO	2021	2022	7277.0	Manual	4
21	16168924699415058432833470215835318 9075	8.0	PORSCHE	CAYENNE	3.0 V6 GASOLINA AWD TIPTRONIC S	2020	2020	8220.0	Automática	4
...	...	...	...	...	...	...	...	...	...	...
29564	71538310533068859372352294256851 59881	8.0	VOLKSWAGEN	GOL	1.6 16V MSI TOTALFLEX 4P AUTOMÁTICO	2021	2021	21166.0	Automática	4

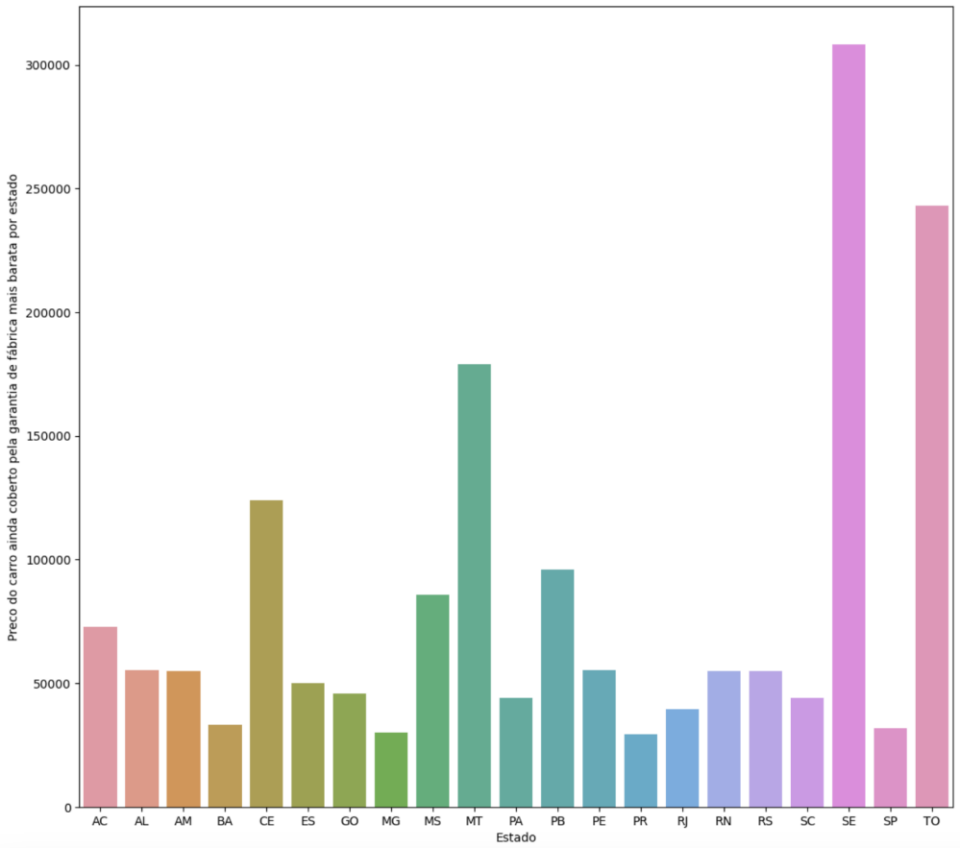
E procurei pelo índice de menor preço. Organizei conforme o trecho de código abaixo para filtrar por marca, modelo, estado do vendedor e o preço.

```
warranty_cars_idx = cars_in_warranty.groupby('estado_vendedor')['preco'].idxmin()
warranty_low = cars_in_warranty.loc[warranty_cars_idx,['marca','modelo','estado_vendedor','preco']]
warranty_low
```

	marca	modelo	estado_vendedor	preco
3421	HYUNDAI	HB20	AC	72811.745577
20493	PEUGEOT	208	AL	55178.097786
14413	TOYOTA	YARIS	AM	54908.191330
12966	VOLKSWAGEN	POLO	BA	33100.914663
3981	PEUGEOT	2008	CE	123939.878795
21534	HYUNDAI	ELANTRA	ES	49919.575982
27939	VOLKSWAGEN	GOL	GO	45812.306967
1930	MITSUBISHI	LANCER	MG	29906.894268
21675	FIAT	STRADA	MS	85730.862915
19842	BMW	Z4	MT	178934.006888
25494	CHEVROLET	MONTANA	PA	43906.629995
11000	VOLKSWAGEN	VIRTUS	PB	95762.746630
25980	CHERY	ARRIZO 5	PE	55243.838003
6101	PEUGEOT	307	PR	29328.116594
5426	AUDI	A6	RJ	39556.398656
12613	VOLKSWAGEN	VIRTUS	RN	54848.887128
20071	NISSAN	KICKS	RS	54742.833352
5427	VOLKSWAGEN	GOL	SC	44004.769748
1413	BMW	X5	SE	308356.595507
7492	PEUGEOT	307	SP	31763.159542
10351	FORD	RANGER	TO	243002.217032

Utilizando o script abaixo, preferi fazer o plot para visualizar melhor

```
plt.figure(figsize=(13,12))
sns.barplot(x = warranty_low['estado_vendedor'], y = warranty_low['preco'])
#plt.title("Estado que mais vende carro popular")
plt.xlabel("Estado")
plt.ylabel("Preco do carro ainda coberto pela garantia de fábrica mais barata por estado")
plt.show()
```



Munido do recorte feito e do gráfico, conseguimos observar que:

Logo, vemos que o estado com o preco mais barato de um carro que ainda é coberto pela garantia de fábrica é o PR com o valor de 29328.116594 reais com o modelo PEUGEOT 307, seguido de MG com o valor de 29906.894268 reais com o modelo MITSUBISHI LANCER.