Trabalho Final de RDI

Felipe Teodoro, Jamille Rocha, Pedro Aragão, Manuella Borges e Henrique Valle

2024-12-03

Utilizando Scrapy para raspar dados da OLX

Neste projeto, utilizei o framework **Scrapy** para realizar a raspagem de dados de anúncios de carros na plataforma OLX. O Scrapy é uma ferramenta poderosa para extração de dados estruturados de websites de forma automatizada e eficiente, capaz de lidar com várias páginas ao mesmo tempo, respeitando limitações de velocidade impostas pelos sites (através de mecanismos como o **AutoThrottle**).

OBS: Uma das vantagens do scrapy é que ele automatiza esses processos e caso uma das requisições não de certo, ele continua

O **OlxCarros** foi configurada para realizar requisições em páginas da OLX, especificamente na seção de **Carros e eletrodomésticos** . As seguintes etapas foram implementadas:

1. **Configuração da aranha**: Definimos um `USER_AGENT` customizado para imitar um navegador legítimo e evitar bloqueios pelo site. Ativamos também o `AUTOTHROTTLE`, que ajusta dinamicamente a velocidade das requisições para evitar sobrecarregar o servidor.
2. **Raspagem de várias páginas**: O método `start_requests` gera requisições para as primeiras 100 páginas do site OLX, e cada página é processada em paralelo, otimizando o tempo de execução.
3. **Extração de dados**: A aranha usa o XPath para acessar a estrutura JSON da página OLX e extrair as informações desejadas, como:
    - **Título** = nome do carro
    - **Preço** do carro
    - **Localização**
4. **Armazenamento dos dados**: Os dados coletados são salvos em um arquivo `JSON`, permitindo uma análise futura e visualização de todos os anúncios extraídos.

Este projeto demonstra como o Scrapy pode ser utilizado para automatizar a coleta de dados de páginas web de forma escalável e eficiente. É uma ferramenta ideal para grandes volumes de dados, com funcionalidades integradas que garantem robustez e confiabilidade na raspagem.

Instalando as bibliotecas necessárias

```
!pip install scrapy requests
```

```
# Caso queira rodar o arquivo coloque eval = TRUE na chunck
import scrapy
from scrapy.crawler import CrawlerProcess
import json
import requests

class OlxCarros(scrapy.Spider):
    name = 'olx '
    custom_settings = {
        'USER_AGENT':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTM
        # serve para ajustar automaticamente  a velocidade de requisição baseado
        #no tempo de resposta do servidor
        # Minimizando o risco de bloqueio e evitando sobrecarregar o site
        'AUTO_THROTTLE_ENABLE':True
    }

    def start_requests(self):
        for page in range(1, 101):
            #Yield é tipo um return
            yield scrapy.Request(f'https://www.olx.com.br/autos-e-pecas/carros-vans-e-ut
            #Fazer mais de um ao mesmo tempo
          #yield scrapy.Request(f'https://www.olx.com.br/eletro/estado-df?o={page}')

    def parse(self, response, **kwargs):
        #Estrutura html do site olx
        html = json.loads(response.xpath('//script[@id="__NEXT_DATA__"]/text()').get())
        carros = html.get('props').get('pageProps').get('ads')
        for carro in carros:
            yield {
                'title': carro.get('title'),
                'price': carro.get('price'),
                'locations': carro.get('location')
            }

#Essa parte adicionei com o GPT só para garantir caso alguém queira rodar o scrapy
#de maneira mais fácil, porém no final do arquivo tem um comentário de como fazer
#para rodar pelo terminal
```

```
# Configurar o processo Scrapy para rodar
process = CrawlerProcess(settings={
    'FEEDS': {
        'olx_carros.json': {
            'format': 'json',
        },
    },
})

# Rodar a aranha OLX
process.crawl(OlxCarros)
process.start()

#Uma das vantagens do scrapy é que ele automatiza esses processos e caso uma das requi
#não de certo, ele continua

#para rodar no terminal:scrapy runspider .(Acessar o local do seu arquivo, nesse caso
#o meu era \olx_eletro.py )-O olx_eletro.json (Escolher como salvar o arquivo)
```

Análise dos dados obtidos do Scrapy

Importando dataset via drive

```
import pandas as pd
# Acessando o dataset com os dados obtidos do webscrapping
dataset = "https://drive.google.com/uc?export=download&id=1rlpBN9aod4NSPccQ60yowXCVmeg57

# Lendo arquivo
df = pd.read_json(dataset)
```

Realizando limpeza

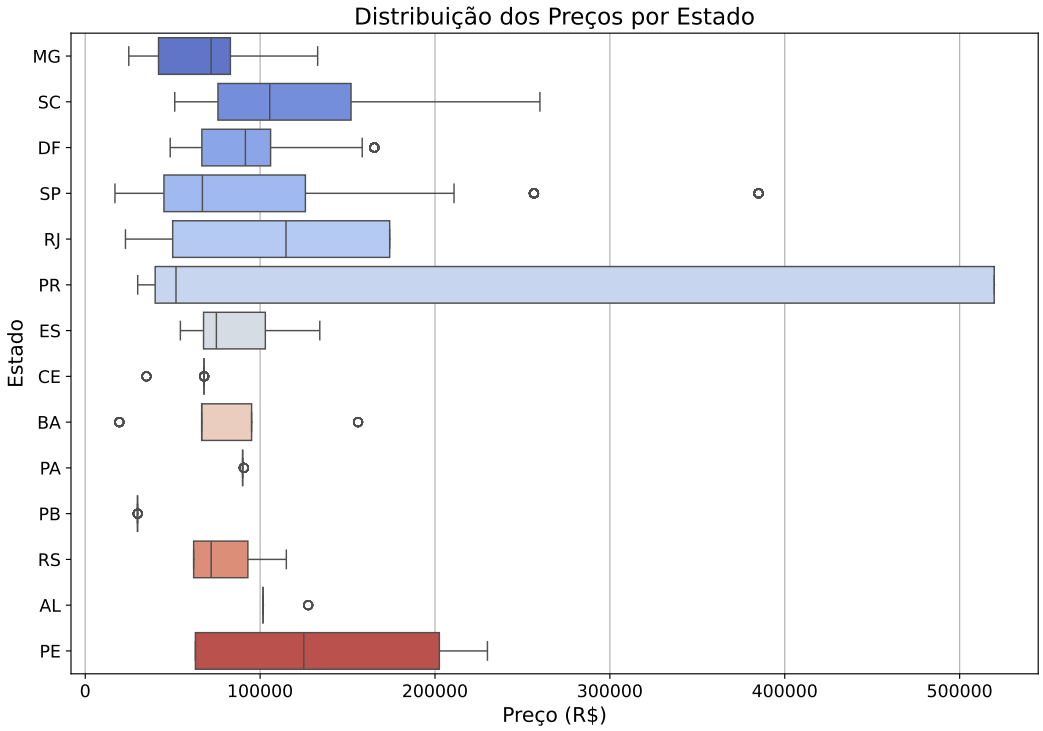
```
import pandas as pd
import numpy as np

# Função para limpar e converter preços
def clean_price(price):
    try:
        # Remove símbolos como 'R$', espaços e converte para float
        return float(price.replace('R$', '').replace('.', '').replace(',', '').strip())
```

```
    except:
        return np.nan

# Função para dividir localização em cidade e estado
def split_location(location):
    try:
        city, state = location.split(' - ')
        return city.strip(), state.strip()
    except:
        return np.nan, np.nan

# Limpando os dados
df.dropna(subset=['title', 'price', 'locations'], inplace=True)

# Aplicando a limpeza nos preços
df['price'] = df['price'].apply(clean_price)

# Removendo linhas com preços inválidos (NaN)
df.dropna(subset=['price'], inplace=True)

# Aplicando a separação de localização em cidade e estado
df[['city', 'state']] = df['locations'].apply(lambda x: pd.Series(split_location(x)))

# Removendo linhas com localização inválida
df.dropna(subset=['city', 'state'], inplace=True)

# Exibindo as primeiras linhas para verificar a limpeza
print(df.head())
```

```
##                                                title     price  ...      city state
## 0     Fiat Cronos 2024 1.0 firefly flex drive manual   70890.0  ...     Betim    MG
## 1  Volkswagen T-cross 2023 1.0 200 tsi total flex...  102019.0  ...  São José    SC
## 2  Hyundai Hb20 2024 1.0 tgdi flex platinum autom...  111190.0  ...  Criciúma    SC
## 3      Hyundai Hb20 2024 1.0 12v flex comfort manual   70790.0  ...  Brasília    DF
## 4                          Peças de morave 2011 diesel   60000.0  ...  Ipatinga    MG
##
## [5 rows x 5 columns]
```

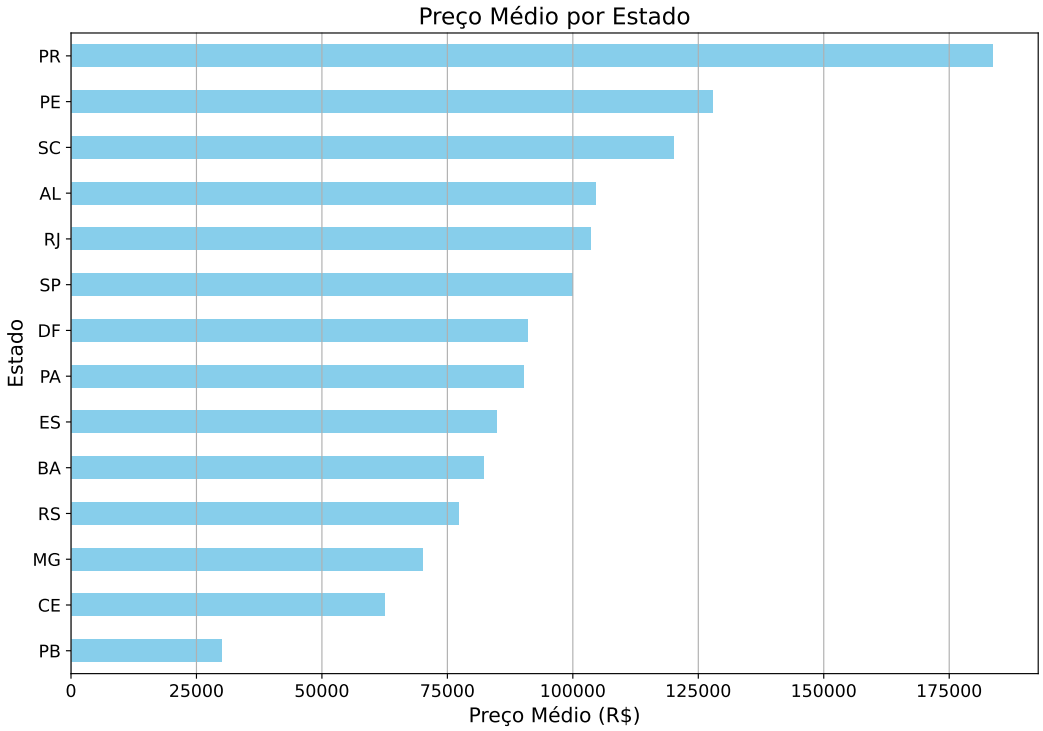
Realizando análises descritivas

Distribuição dos Preços por Estado



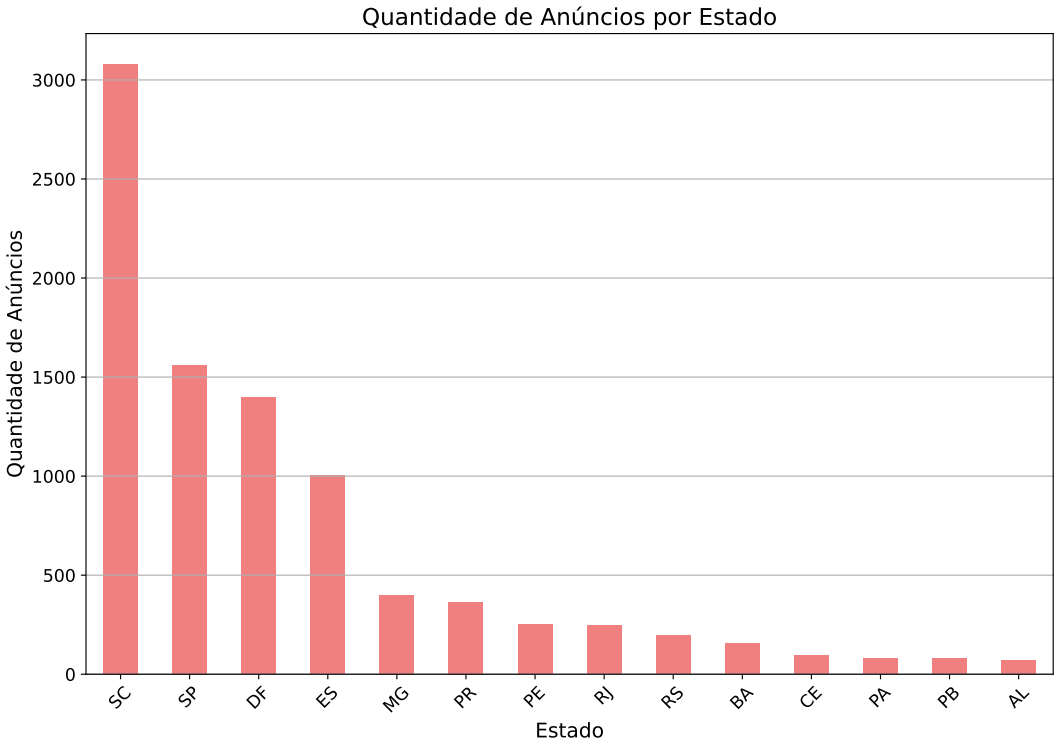
- A variabilidade nos preços é evidente, com estados como PR apresentando valores muito altos e grande dispersão (outliers significativos). Isso pode indicar a presença de veículos de luxo em alguns estados.
- Alguns estados, como PB e AL, têm uma distribuição mais estreita, sugerindo um mercado menos diversificado ou com menos veículos de alto valor.
- A presença de outliers (carros muito caros) em diversos estados pode estar distorcendo o preço médio.

Preço média por Estado



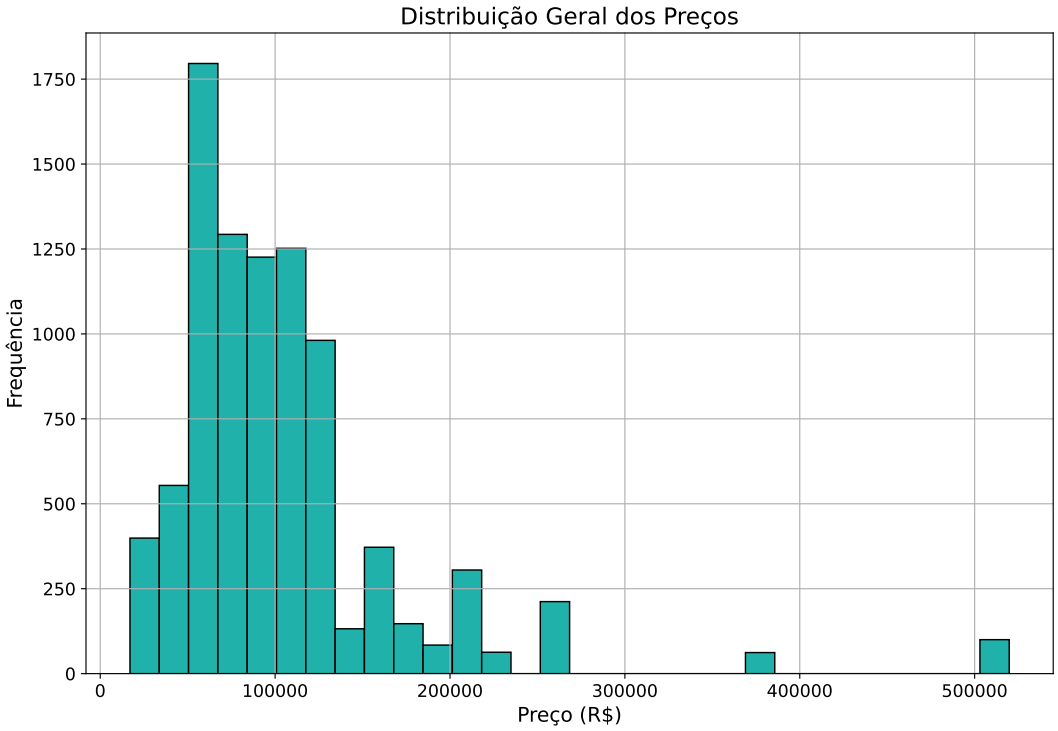
- PR lidera com o maior preço médio, provavelmente influenciado pela presença de veículos de luxo (como visto na análise de outliers).
- Estados do Nordeste (como PB e CE) apresentam preços médios mais baixos, o que pode ser reflexo de menor poder aquisitivo local ou predominância de veículos populares.
- Estados como SC e SP, apesar de terem mercados grandes, possuem preços médios mais moderados, indicando uma oferta diversificada.

Quantidade de anúncio por Estado



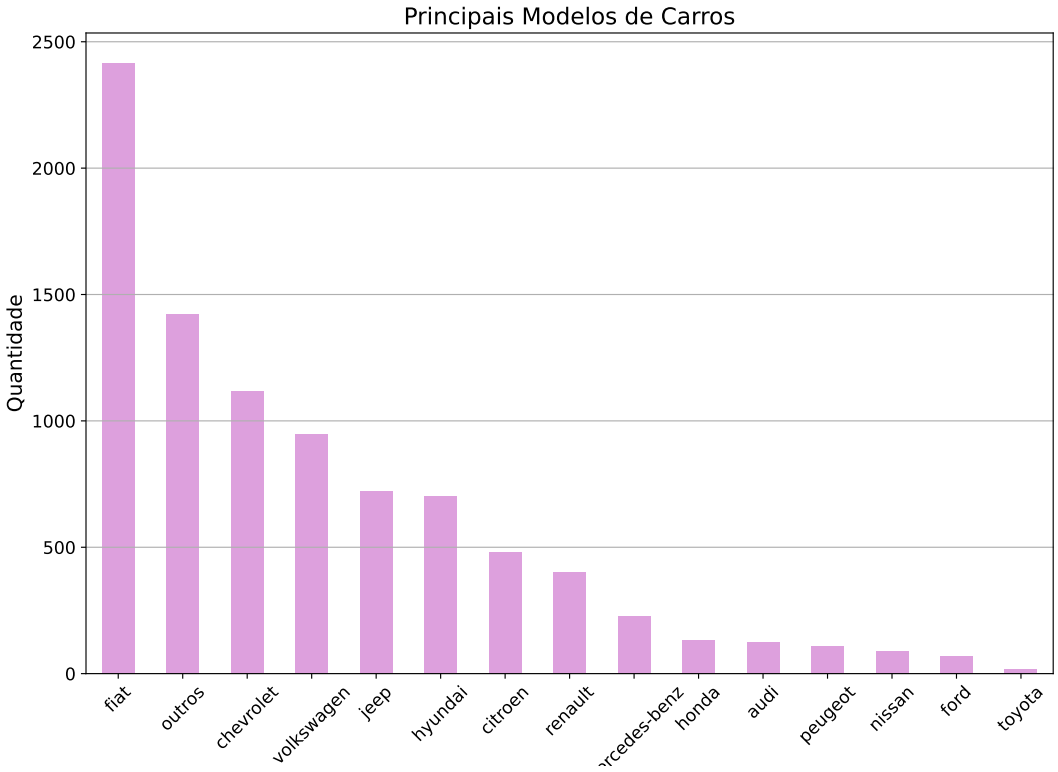
- SC, SP, e DF concentram a maioria dos anúncios, sugerindo que são mercados mais ativos e com maior oferta.
- Estados como AL, PB, e PA têm pouca representatividade, indicando mercados menores ou menor uso da plataforma na região.
- Essa concentração pode enviesar análises gerais, pois estados com poucos anúncios podem não representar o mercado local de maneira precisa.

Distribuição geral dos preços



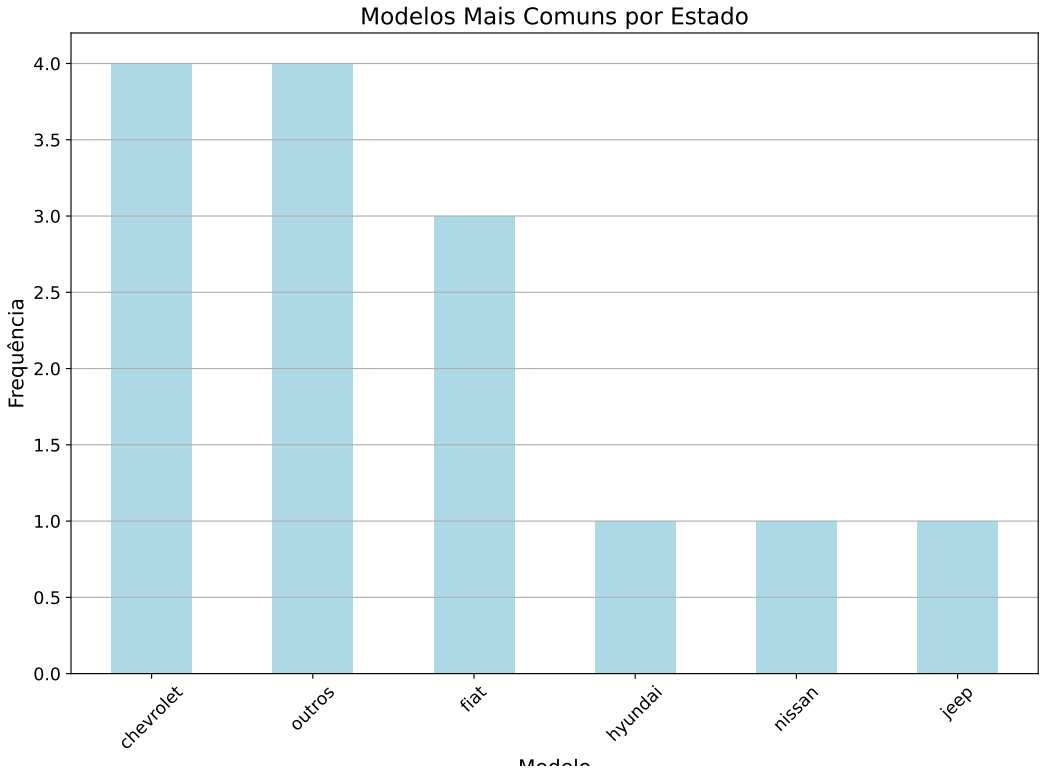
- A maior concentração de veículos está na faixa de preço entre R$ 50.000 e R$ 120.000, indicando que a maioria dos anúncios é de veículos populares ou seminovos.
- Há uma cauda longa na distribuição, com preços muito altos (acima de R$ 300.000) representando uma pequena fração do mercado. Essa concentração é consistente com um mercado que prioriza carros de custo médio, mas com uma pequena participação de veículos premium.

Principais Modelos de Carro



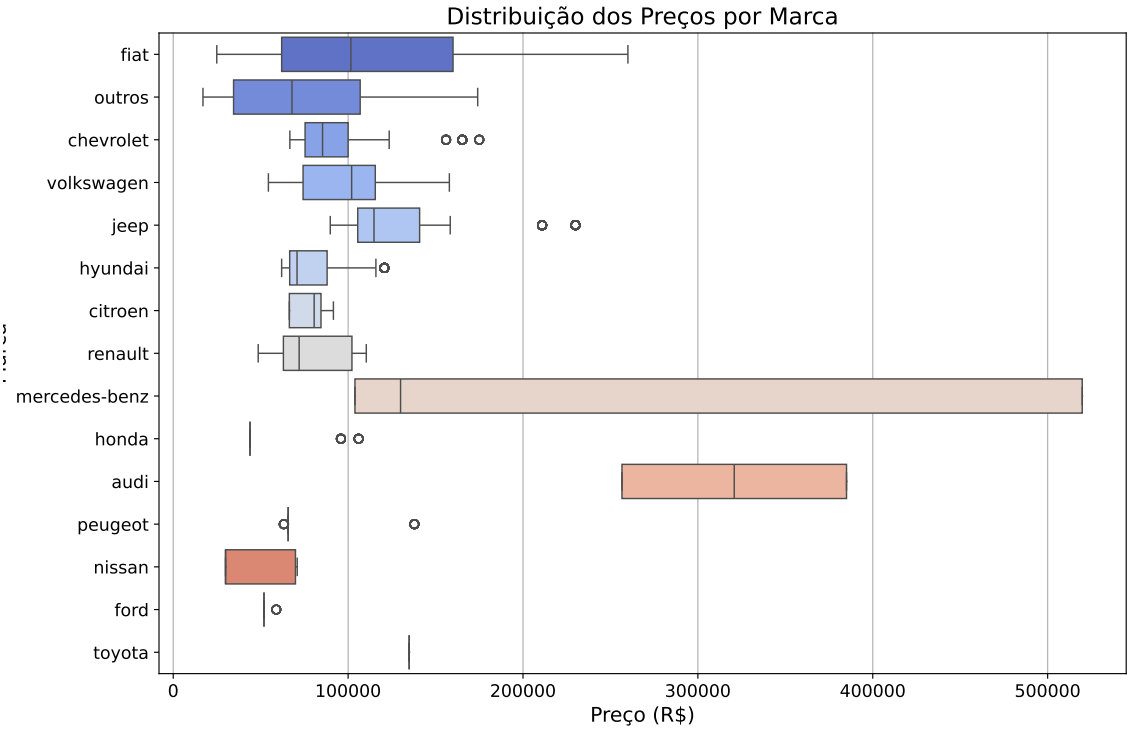
- Fiat e Chevrolet dominam em muitos estados, refletindo sua forte presença no mercado brasileiro.
- Alguns estados, como RJ e RS, apresentam o valor “outros”, indicando que os modelos mais comuns podem estar diluídos ou que não foi possível identificar um padrão claro.
- A predominância de modelos populares em alguns estados pode estar alinhada com o perfil econômico da região.

Modelo Mais Comum por Estado



- A Fiat lidera com ampla margem, mostrando sua forte conexão com o consumidor de veículos mais acessíveis.
- Marcas premium e de menor volume, como Mercedes-Benz e Audi, mantêm nichos bem definidos, mas representam uma pequena fatia do mercado.
- O agrupamento “outros” é significativo e merece atenção em análises futuras, já que pode conter marcas emergentes ou de mercado específico.

Distribuição de preço por marca



- A Mercedes-Benz domina a faixa de preços mais elevados, destacando-se pela ampla variação de valores.
- Marcas populares, como Fiat e Renault, têm os veículos mais acessíveis, com menor dispersão de preços.
- Outliers indicam a presença de modelos premium ou edições especiais em marcas comumente acessíveis.
- A dispersão dos dados reflete a diversidade de portfólios entre marcas, especialmente para aquelas com presença no segmento SUV.

Carros mais caros da OLX

```
## Top 10 Modelos com Preços Médios Mais Altos:

## title
## mercedes-benz gle400d 4m co 2021                                519777.0
## audi q5 spb. s-line black 2.0 tfsi qua.s-tron                   384990.0
## fiat titano ranch turbodiesel at. 2025                          259990.0
## audi q5 2022 2.0 45 tfsi gasolina prestige quattro s tronic     256590.0
```

```
## jeep commander overland 4x4 diesel 2023                          229990.0
## fiat toro ranch turbodiesel 4x4 at9 2025                         216990.0
## jeep compass 2022 1.3 t270 turbo flex limited at6                210890.0
## fiat toro volcano turbodiesel 4wd at9                            199990.0
## fiat scudo cargo 1.5 td 4p 2025                                  182990.0
## chevrolet s10 2023 s10 pick-up ls 2.8 tdi 4x4 cd dies. mec.      174990.0
## Name: price, dtype: float64
```

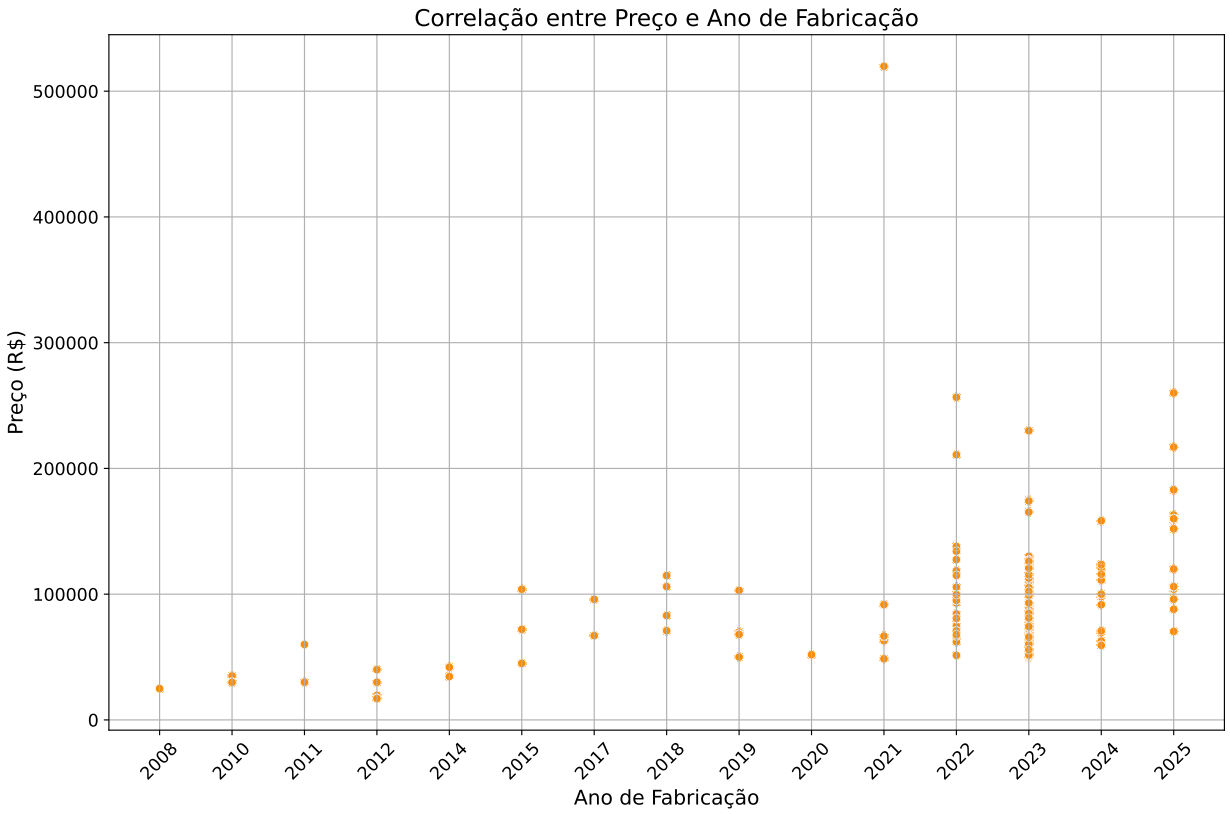
- Veículos premium, como Mercedes-Benz GLE400 e Audi Q5, dominam o topo da lista, reforçando a ideia de que a cauda longa na distribuição geral dos preços é composta principalmente por carros de luxo.
- A lista é dominada por SUVs e caminhonetes, refletindo tendências do mercado por veículos maiores e mais robustos. Esses modelos provavelmente são responsáveis por outliers e distorções nos preços médios de alguns estados.

Mais limpeza

Dessa vez realizei uma limpeza para extrair somente o ano do carro e ver a sua relação com o preço

```python
def extract_year(title):
    match = re.search(r'\b(19|20)\d{2}\b', title)
    if match:
        return match.group(0)
    return None

# Aplicar a função para criar a coluna 'year'
df['year'] = df['title'].apply(extract_year)
df = df.sort_values(by='year')
```

Correlaçõa entre Ano e Preço



- Existe uma relação direta entre a fabricação recente e preços mais elevados, com maior dispersão nos anos mais novos devido à variedade de modelos e categorias.
- Veículos fabricados antes de 2015 têm preços consistentemente baixos, indicando depreciação acentuada. -Outliers em anos recentes indicam modelos premium ou de luxo.
- A análise reforça a importância do ano de fabricação como um fator-chave no valor de mercado dos veículos.

Realizando análise sobre o Data Hackers 2023

Extraindo e visualizando os dados

```
!pip install pandas numpy re seaborn matplotlib
```

```
import pandas as pd

dados = 'https://drive.google.com/uc?export=download&id=1S50IWhhX2n76P8954wQj_1mHe5LOjje
data = pd.read_csv(dados)
data.head()
```

```
##                         ('P0', 'id')  ...  ('P8_d_12 ', 'Treinando e aplicando LLM's p
## 0  001b2d1qtli8t9z7oqgdhj001b2d4i0g  ...
## 1  0026aa3fwd78u0026asg7456tfkjg2cs  ...
## 2  00r21rb9pusd1b0v7ew00r21rw3dy69w  ...
## 3  00urm3jf2cek12w6ygue00urm3jzd17j  ...
## 4  00v0az4g792svil00vn6y1kfm9hq8vy9  ...
##
## [5 rows x 399 columns]
```

Selecionar colunas interessantes de se analisar

```
# Selecionar Colunas Específicas
colunas_selecionadas = [
    "('P1_a ', 'Idade')",
    "('P1_c ', 'Cor/raca/etnia')",
    "('P1_m ', 'Área de Formação')",
    "('P2_h ', 'Faixa salarial')",
    "('P1_e_2 ', 'Experiencia prejudicada devido a minha Cor Raça Etnia')",
    "('P1_b ', 'Genero')",
    "('P1_f_1', 'Quantidade de oportunidades de emprego/vagas recebidas')",
    "('P1_f_3', 'Aprovação em processos seletivos/entrevistas')",
    "('P2_s ', 'Qual a forma de trabalho ideal para você?')",
    "('P3_b ', 'Quais desses papéis/cargos fazem parte do time (ou chapter) de dados da
    "('P3_d ', 'Quais são os 3 maiores desafios que você tem como gestor no atual moment
    "('P4_b ', 'Quais das fontes de dados listadas você já analisou ou processou no trab
    "('P4_d ', 'Quais das linguagens listadas abaixo você utiliza no trabalho?')"
]

# Filtrar o dataframe original para manter apenas as colunas especificadas
df_filtrado = data[colunas_selecionadas]
```

Renomear as colunas para facilitar o processo de análise

```
# Renomear as colunas selecionadas com nomes mais simples
colunas_renomeadas = {
    "('P1_a ', 'Idade')": 'Idade',
    "('P1_c ', 'Cor/raca/etnia')": 'Cor_raca_etnia',
    "('P1_m ', 'Área de Formação')": 'Area_de_Formacao',
    "('P2_h ', 'Faixa salarial')": 'Faixa_salarial',
    "('P1_e_2 ', 'Experiencia prejudicada devido a minha Cor Raça Etnia')": 'Experiencia
    "('P1_b ', 'Genero')": 'Genero',
```

```
    "('P1_f_1', 'Quantidade de oportunidades de emprego/vagas recebidas')": 'Qtd_oportun
    "('P1_f_3', 'Aprovação em processos seletivos/entrevistas')": 'Aprovacao_processos_s
    "('P2_s ', 'Qual a forma de trabalho ideal para você?')": 'Forma_trabalho_ideal',
    "('P3_b ', 'Quais desses papéis/cargos fazem parte do time (ou chapter) de dados da
    "('P3_d ', 'Quais são os 3 maiores desafios que você tem como gestor no atual moment
    "('P4_b ', 'Quais das fontes de dados listadas você já analisou ou processou no trab
    "('P4_d ', 'Quais das linguagens listadas abaixo você utiliza no trabalho?')": 'Ling
}

# Aplicar a renomeação no DataFrame filtrado
df_filtrado = df_filtrado.rename(columns=colunas_renomeadas)

# Exibir as primeiras linhas do DataFrame renomeado
df_filtrado.head()
```

```
##    Idade  ...   Linguagens_utilizadas
## 0     31  ...              Python, SQL
## 1     30  ...              SQL, Python
## 2     37  ...  SQL, Python, SAS/Stata
## 3     22  ...                     NaN
## 4     34  ...              SQL, Python
##
## [5 rows x 13 columns]
```

Transformação de dados

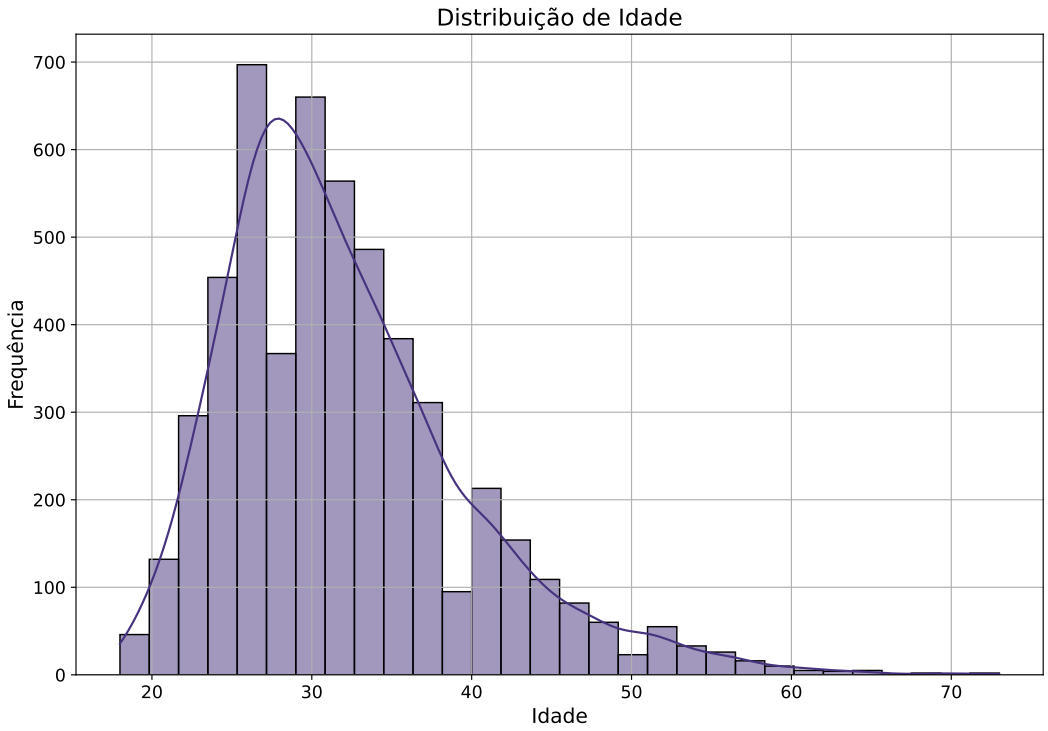
Arrumar a faixa salária que está como intervalo

```
import numpy as np
import re

# Função para extrair a média salarial de cada faixa
def extrair_media_salarial(faixa):
    if pd.isna(faixa):
        return np.nan
    valores = re.findall(r'\d+', faixa)
    if len(valores) == 4:
        min_valor = int(valores[0] + valores[1])
        max_valor = int(valores[2] + valores[3])
        media = (min_valor + max_valor) / 2
        return media
    return np.nan
```

```
# Aplicar a função na coluna de faixa salarial
df_filtrado['Media_salarial'] = df_filtrado['Faixa_salarial'].apply(extrair_media_salari
```

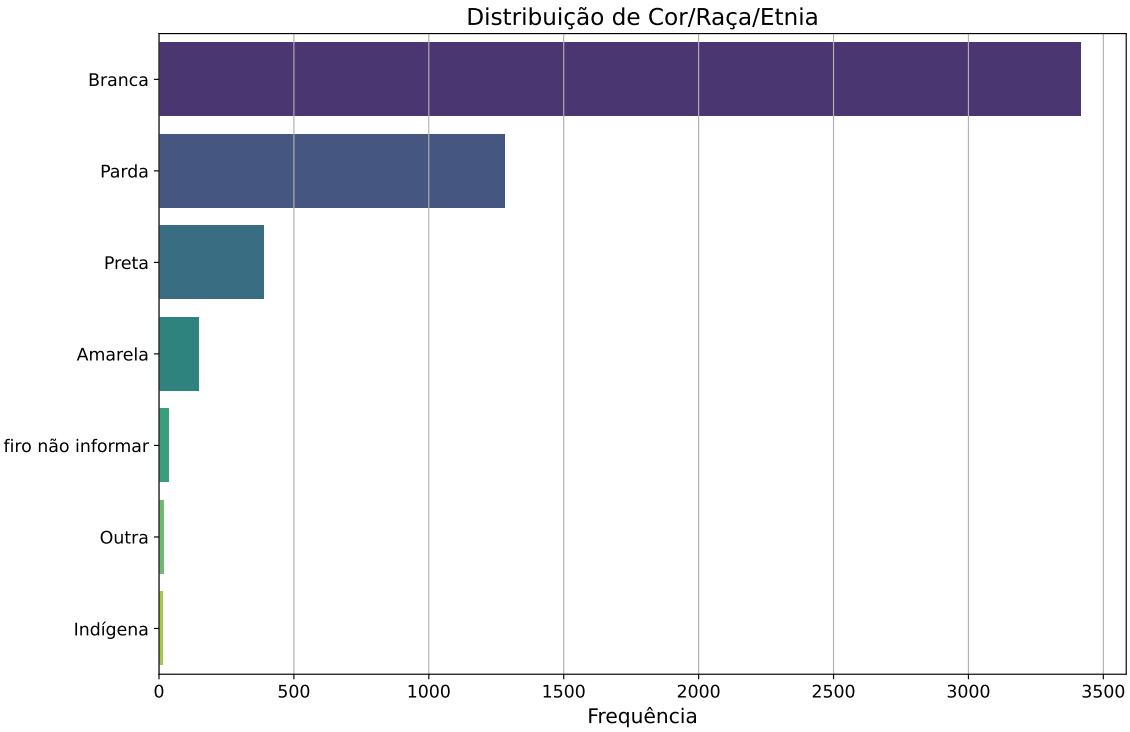
Análise exploratória de dados

Distribuição das idades



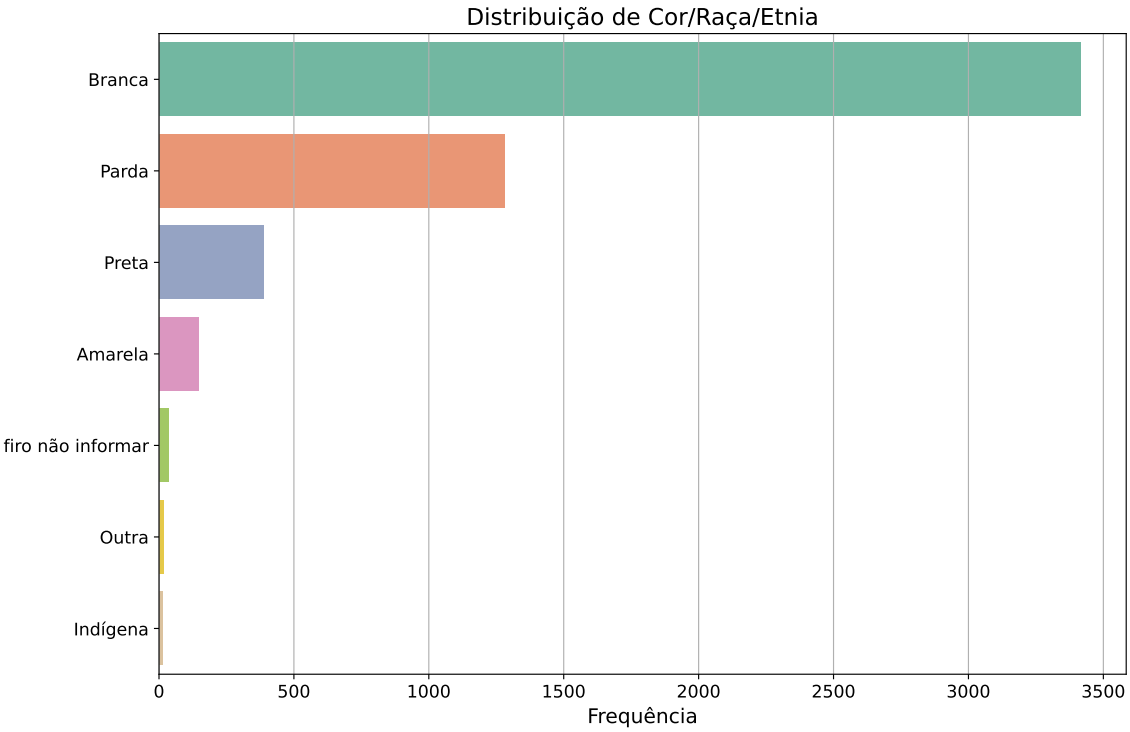
- A maior concentração de indivíduos está entre 25 e 35 anos, com um pico em torno dos 30 anos. Isso sugere que a maior parte das pessoas na área de dados está no início ou no meio de suas carreiras profissionais.
- A partir dos 35 anos, há uma diminuição clara na frequência. Isso pode indicar que a área de dados atrai predominantemente profissionais mais jovens ou que há menos profissionais permanecendo nessa área em idades mais avançadas.

Distribuição das Etnia



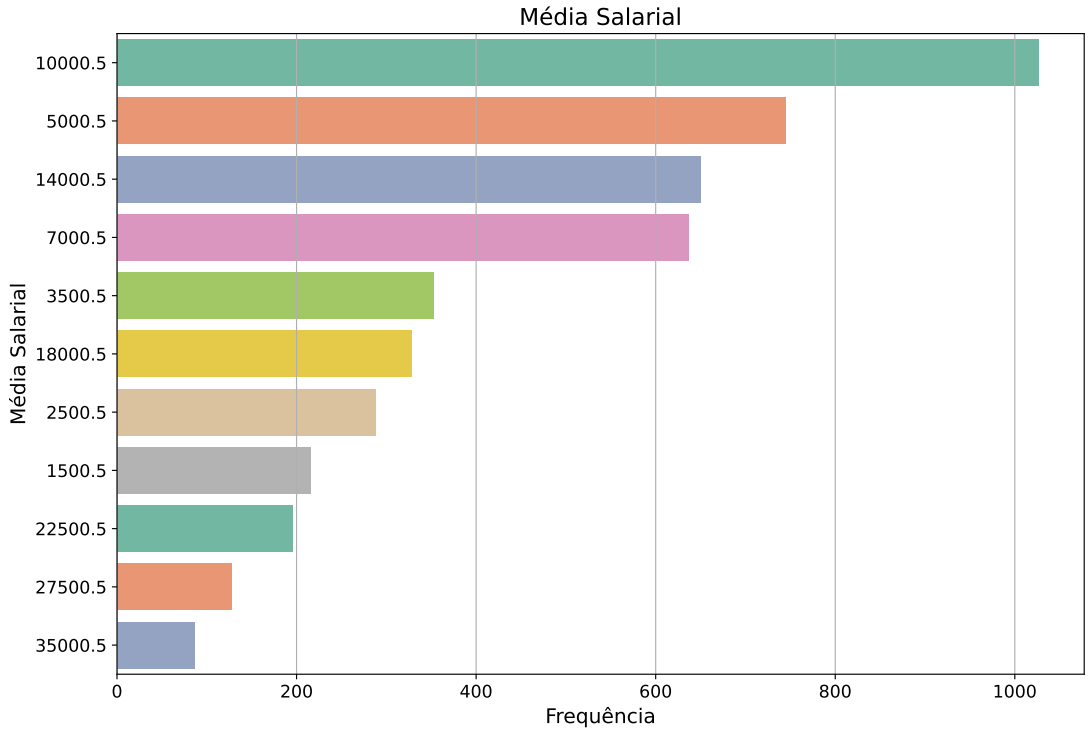
- A desigualdade na representatividade de cor/raça/etnia pode indicar barreiras sistêmicas na área de dados. A predominância de indivíduos brancos e pardos sugere que grupos historicamente marginalizados, como pessoas pretas e indígenas, podem enfrentar dificuldades para ingressar ou se destacar nesse setor.
- Um número moderado de pessoas optou por não declarar sua cor/raça/etnia, o que pode indicar uma hesitação ou desconforto em fornecer essa informação.
- Em um setor que depende de diferentes perspectivas para resolver problemas complexos, a falta de diversidade pode limitar a inovação e perpetuar viés nos produtos ou análises criados.

Distribuição da área de formação



- A ampla representatividade de cursos como Computação e Engenharia reforça a ideia de que a área de dados é dominada por profissionais com formação técnica ou matemática.
- A presença de profissionais de Marketing e Ciências Biológicas demonstra que o setor também está aberto a profissionais com formações diversas, desde que possuam ou adquiram habilidades relacionadas à análise de dados.

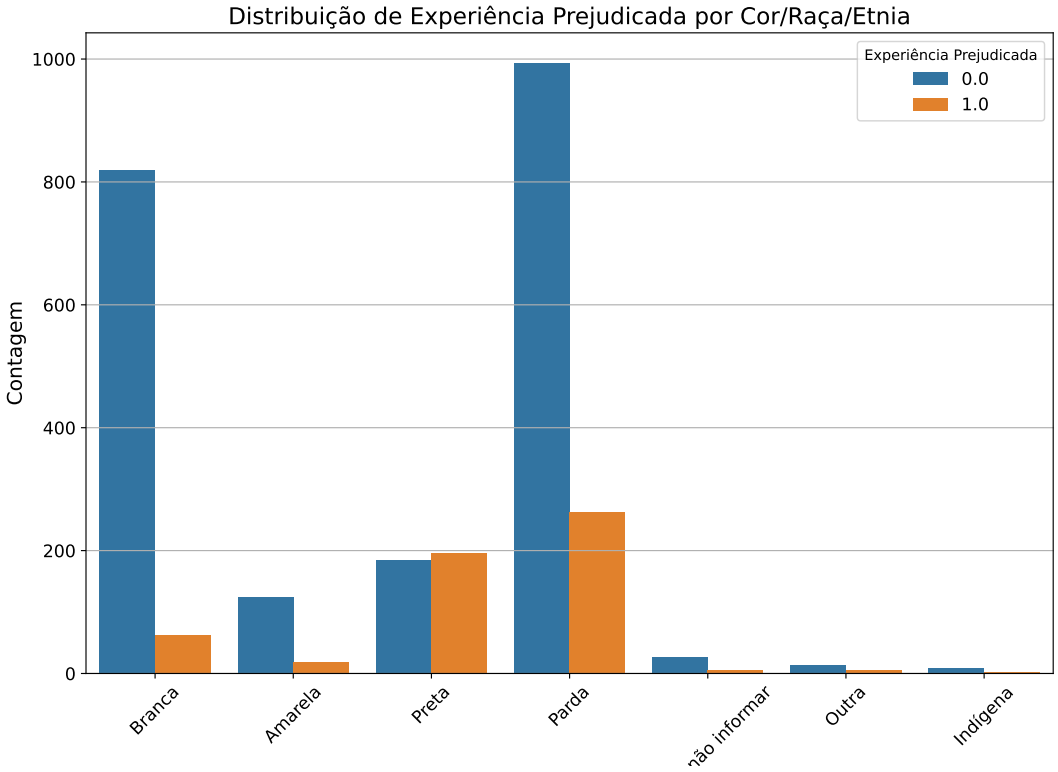
Distribuição do Salário



- A faixa salarial mais representada está em torno de RS 10.000,50, seguida pelas faixas de RS 5.000,50 e R$ 14.000,50. Isso sugere que a maioria dos profissionais no setor de dados possui remuneração intermediária a alta.
- As faixas menores (como RS 3.500,50 e R$ 1.500,50) também têm menor representatividade, indicando que a área de dados tende a remunerar acima da média desde o início da carreira

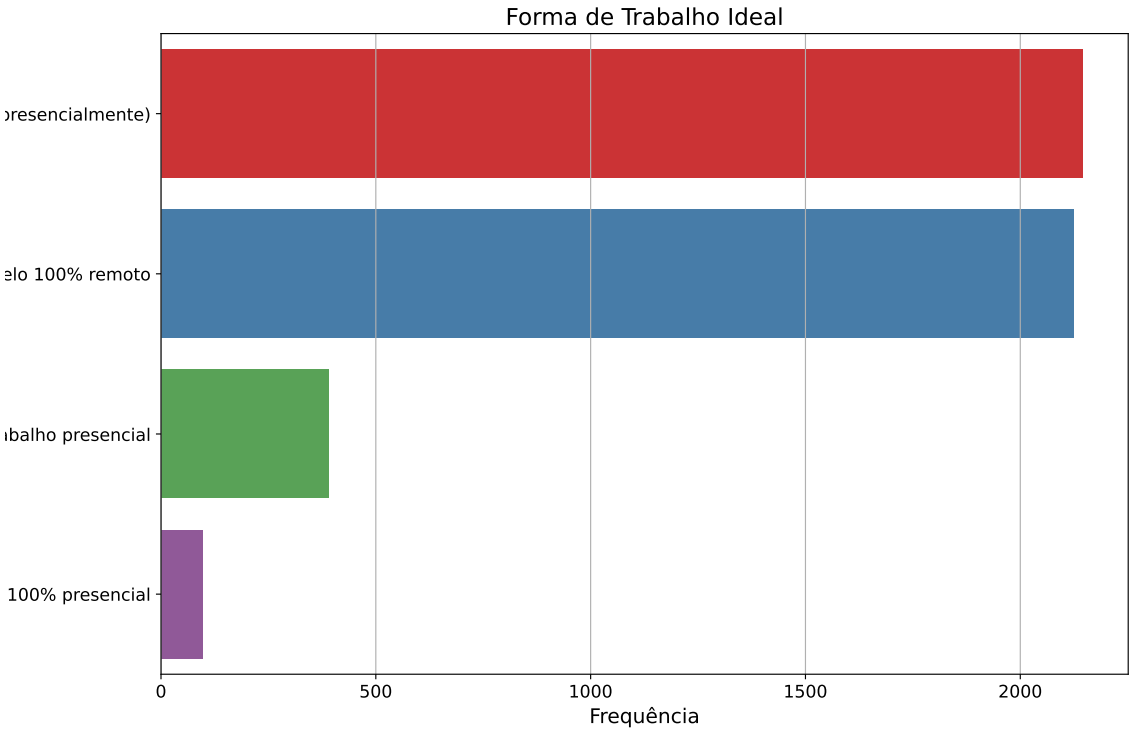
Experiência prejudicada

Já que há pessoas que se sentem prejudicadas devido a sua cor/raça/etnia, decidi ver qual a proporção de cada um e entender mais a fundo o que está acontecendo no mercado



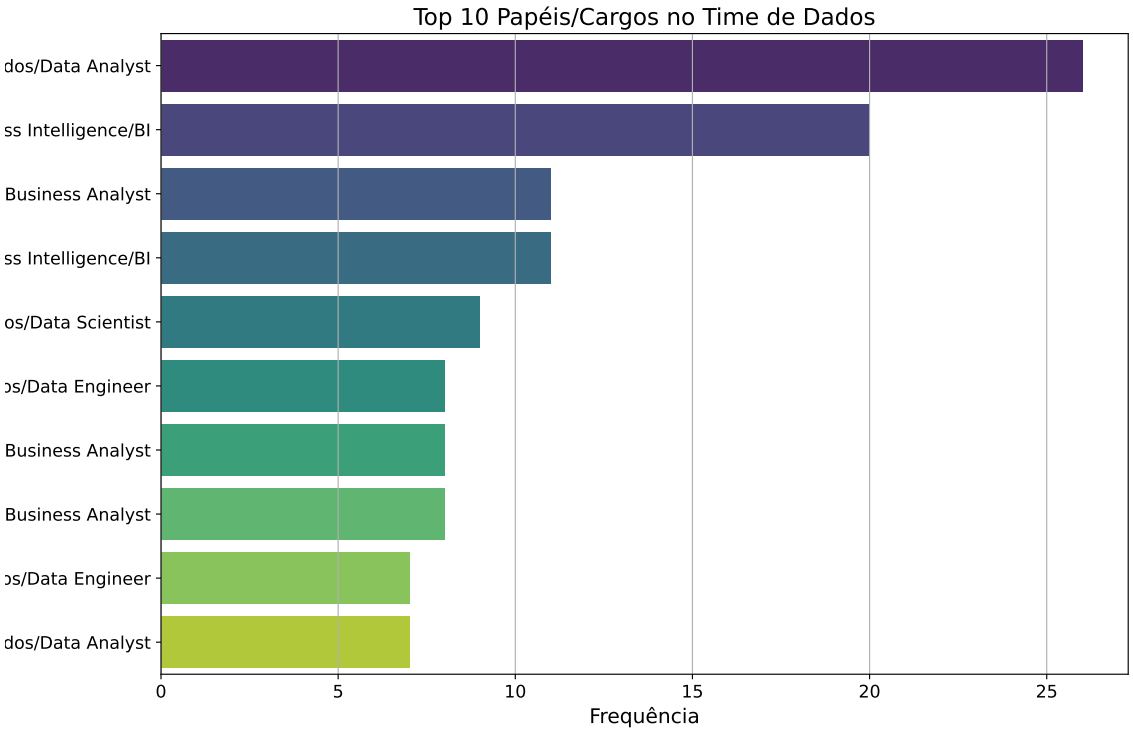
- O gráfico evidencia que indivíduos negros e pardos relatam mais experiências prejudicadas do que os brancos, o que reflete desigualdades no mercado de trabalho em dados.
- Essas diferenças podem ser resultado de fatores como:Discriminação racial direta, Falta de oportunidades iguais e Barreiras educacionais ou econômicas prévias que impactam o acesso ao setor de dados.

Modelo de trabalho



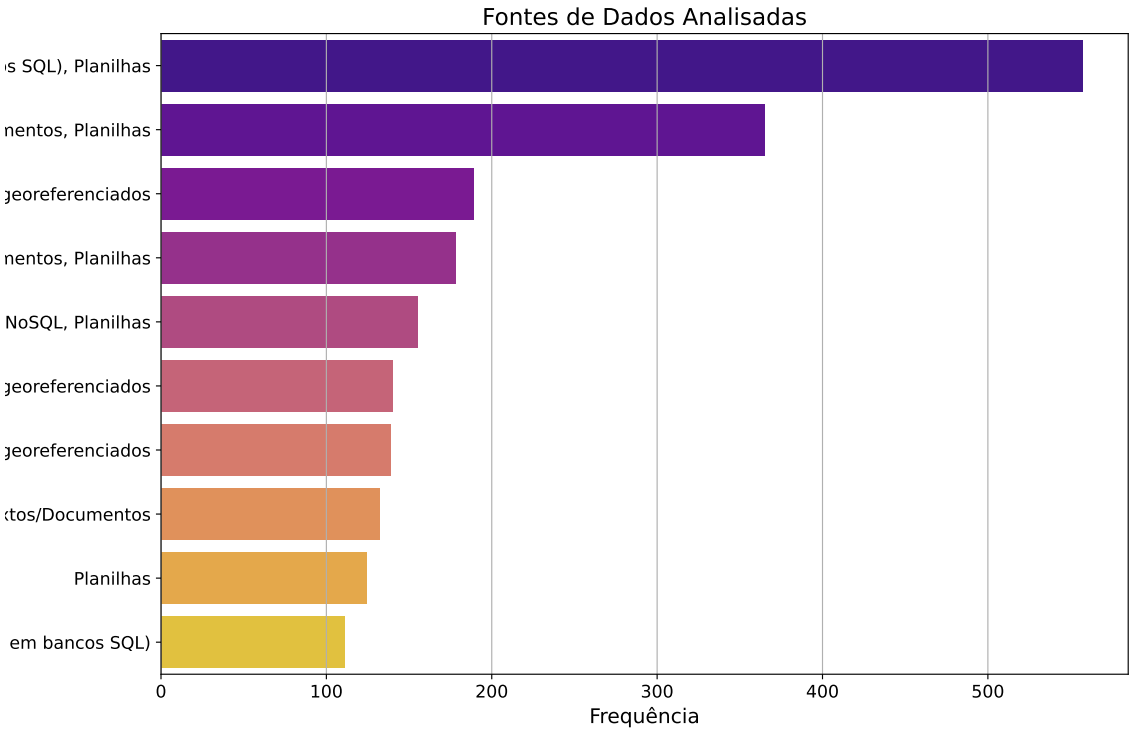
- A maior parte dos profissionais prefere um modelo híbrido flexível. Essa preferência reflete a busca por autonomia e equilíbrio entre vida pessoal e trabalho.
- A pandemia popularizou modelos de trabalho remoto e híbrido, e isso parece ter alterado permanentemente as preferências, especialmente em setores que não dependem de presença física.
- O setor de dados, sendo altamente tecnológico, se adapta bem ao trabalho remoto, o que facilita a predominância de preferências por modelos flexíveis.
- As preferências indicam que os profissionais associam maior flexibilidade a uma melhor qualidade de vida e produtividade.

Principais cargos na área de dados



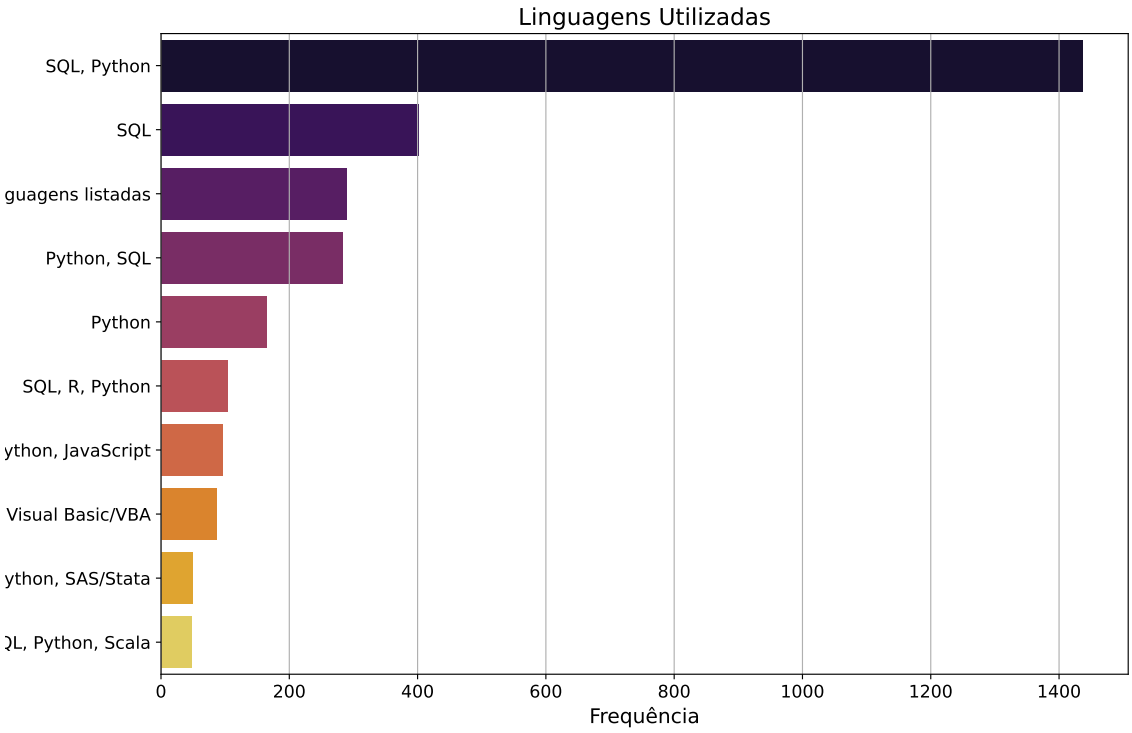
- A maior frequência de analistas de dados e BI sugere que muitas equipes de dados estão mais focadas na análise e entrega de insights do que na construção de infraestrutura avançada (como é o caso de engenheiros de dados) ou na pesquisa de ponta (cientistas de dados).
- Os papéis combinados podem refletir uma demanda por profissionais "generalistas" que podem executar diversas tarefas. Isso pode ser vantajoso para pequenas equipes, mas pode levar a sobrecarga e falta de especialização.
- Embora cargos como cientista de dados e engenheiro de dados sejam menos frequentes, sua presença sugere que há uma evolução no mercado em direção a papéis mais técnicos e especializados.

Principais fontes de dados analisadas na área de dados



- A preferência por bancos SQL e planilhas mostra que, mesmo com o crescimento de tecnologias modernas, as ferramentas clássicas continuam sendo indispensáveis, talvez devido à familiaridade e facilidade de uso.
- A presença de bancos NoSQL e dados não estruturados (textos, documentos) mostra que o mercado está começando a adotar tecnologias modernas para atender a novas demandas, como Big Data e análise avançada.
- O uso de múltiplas fontes indica a necessidade de ferramentas e profissionais capazes de integrar e transformar dados de diversas origens.

Principais linguagens utilizadas na área de dados



- Esses resultados reforçam que SQL e Python são os pilares do trabalho com dados. A combinação dessas ferramentas cobre uma ampla gama de tarefas, desde manipulação de dados até aprendizado de máquina.
- A presença de linguagens como R e JavaScript reflete a diversidade de demandas no setor: R para análise estatística avançada. JavaScript para integração com desenvolvimento front-end ou visualizações interativas.
- O grupo que não utiliza linguagens pode apontar para uma dependência maior de ferramentas gráficas ou funções menos técnicas. Isso pode ser uma oportunidade para capacitação em linguagens.

Conclusão

O projeto demonstrou a eficiência do uso do **Scrapy** para raspagem de dados em larga escala. Foi possível estruturar informações relevantes de anúncios da OLX e armazená-las para análise posterior. Além disso, foram destacados aspectos éticos e desafios técnicos da raspagem de dados, como a importância de respeitar os termos de uso das plataformas e ajustar a velocidade das requisições para evitar sobrecarga. A reflexão sobre as tecnologias utilizadas reforça a relevância de ferramentas clássicas e modernas para o trabalho com dados.

Referências

1. **Fontes de Dados**: Plataforma OLX (dados raspados da seção de carros) e dataset publicado pelo DataHackers retirado do kaggle .
2. **Ferramentas**:
    - Framework Scrapy para automação de raspagem.
    - Python para análise de dados e visualizações.
3. **Bibliotecas Utilizadas**:
    - `scrapy`: Para construção de aranhas e coleta de dados.
    - `requests`: Para manipulação de requisições HTTP.
    - `json`: Para manipulação dos dados raspados.
    - Bibliotecas adicionais para visualizações e análises: `matplotlib`, `seaborn`.