

report

April 17, 2025

```
[ ]: import os
import subprocess
import sys
import getpass
from pathlib import Path

# --- CONFIGURAÇÃO ---
raw_source = "./"
# Se o path começar com "/media/~/", substitui "~" pelo seu usuário em /media
username = getpass.getuser()
if raw_source.startswith("/media/~/"):
    corrected = raw_source.replace("/media/~/", f"/media/{username}/")
else:
    corrected = raw_source
# Agora expande "~" se ainda existir
SOURCE_DIR = Path(os.path.expanduser(corrected))
OUTPUT_DIR = SOURCE_DIR / "pdfs"
PYTHON_EXE = sys.executable
# -----

# Verifica pasta de origem
if not SOURCE_DIR.exists():
    raise FileNotFoundError(f"Pasta de origem não encontrada: {SOURCE_DIR}")

# Cria apenas a pasta pdfs dentro de SOURCE_DIR, sem recriar o resto
OUTPUT_DIR.mkdir(exist_ok=True)

# Converte cada .ipynb em PDF
for nb_path in SOURCE_DIR.glob("*.ipynb"):
    pdf_name = nb_path.stem + ".pdf"
    cmd = [
        PYTHON_EXE, "-m", "jupyter", "nbconvert",
        str(nb_path),
        "--to", "pdf",
        "--output", pdf_name,
        "--output-dir", str(OUTPUT_DIR),
        "--debug"
```

```

]
print(f" Convertendo {nb_path.name} → {pdf_name}")
try:
    out = subprocess.run(cmd, check=True, capture_output=True, text=True)
    print(out.stdout)
except subprocess.CalledProcessError as e:
    print(f" Erro ao converter {nb_path.name}: \n{e.stderr}")
print(" Conversão de todos os notebooks finalizada.")

```

Convertendo 1 - Data Loading.ipynb → 1 - Data Loading.pdf

Convertendo 2 - Data Analysis.ipynb → 2 - Data Analysis.pdf

Convertendo 0 - Geral.ipynb → 0 - Geral.pdf

Erro ao converter 0 - Geral.ipynb:

```

[NbConvertApp] Searching ['/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment - Otimização da taxa de retenção/.aha_venv/etc/jupyter',
'/home/tschmitz/.jupyter', '/usr/local/etc/jupyter', '/etc/jupyter'] for config
files

```

```

[NbConvertApp] Looking for jupyter_config in /etc/jupyter

```

```

[NbConvertApp] Looking for jupyter_config in /usr/local/etc/jupyter

```

```

[NbConvertApp] Looking for jupyter_config in /home/tschmitz/.jupyter

```

```

[NbConvertApp] Looking for jupyter_config in /media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment - Otimização da taxa de
retenção/.aha_venv/etc/jupyter

```

```

[NbConvertApp] Looking for jupyter_nbconvert_config in /etc/jupyter

```

```

[NbConvertApp] Looking for jupyter_nbconvert_config in /usr/local/etc/jupyter

```

```

[NbConvertApp] Looking for jupyter_nbconvert_config in /home/tschmitz/.jupyter

```

```

[NbConvertApp] Looking for jupyter_nbconvert_config in /media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment - Otimização da taxa de
retenção/.aha_venv/etc/jupyter

```

```

[NbConvertApp] Looping through config variables with prefix "JUPYTER_NBCONVERT"

```

```

[NbConvertApp] Converting notebook 0 - Geral.ipynb to pdf

```

```

[NbConvertApp] Notebook name is '0 - Geral'

```

```

[NbConvertApp] Template paths:

```

```

    /media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex

```

```

    /media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/base

```

```

    /media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/share/jupyter

```

```

    /media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/share/jupyter/nbconvert/templates

```

```

    /media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de

```

```

retenção/.aha_venv/share/jupyter/nbconvert/templates/compatibility
/home/tschmitz/.local/share/jupyter
/home/tschmitz/.local/share/jupyter/nbconvert/templates
/home/tschmitz/.local/share/jupyter/nbconvert/templates/compatibility
/usr/local/share/jupyter
/usr/local/share/jupyter/nbconvert/templates
/usr/local/share/jupyter/nbconvert/templates/compatibility
/usr/share/jupyter
/usr/share/jupyter/nbconvert/templates
/usr/share/jupyter/nbconvert/templates/compatibility
/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/share/jupyter/nbconvert/templates
[NbConvertApp] Applying preprocessor: TagRemovePreprocessor
[NbConvertApp] Applying preprocessor: RegexRemovePreprocessor
[NbConvertApp] Applying preprocessor: SVG2PDFPreprocessor
[NbConvertApp] Applying preprocessor: LatexPreprocessor
[NbConvertApp] Applying preprocessor: HighlightMagicsPreprocessor
[NbConvertApp] Applying preprocessor: ExtractOutputPreprocessor
[NbConvertApp] Applying preprocessor: ExtractAttachmentsPreprocessor
[NbConvertApp] Attempting to load template index.tex.j2
[NbConvertApp]      template_paths: /media/tschmitz/hope-keeper/2025_github/px-
data-science/7 - AHA! Moment - Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex:/media/tschmitz/hope-
keeper/2025_github/px-data-science/7 - AHA! Moment - Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/base:/media/tschmitz/hope-
keeper/2025_github/px-data-science/7 - AHA! Moment - Otimização da taxa de
retenção/.aha_venv/share/jupyter:/media/tschmitz/hope-keeper/2025_github/px-
data-science/7 - AHA! Moment - Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates:/media/tschmitz/hope-
keeper/2025_github/px-data-science/7 - AHA! Moment - Otimização da taxa de reten
ção/.aha_venv/share/jupyter/nbconvert/templates/compatibility:/home/tschmitz/.lo
cal/share/jupyter:/home/tschmitz/.local/share/jupyter/nbconvert/templates:/home/
tschmitz/.local/share/jupyter/nbconvert/templates/compatibility:/usr/local/share
/jupyter:/usr/local/share/jupyter/nbconvert/templates:/usr/local/share/jupyter/n
bconvert/templates/compatibility:/usr/share/jupyter:/usr/share/jupyter/nbconvert
/templates:/usr/share/jupyter/nbconvert/templates/compatibility:/media/tschmitz/
hope-keeper/2025_github/px-data-science/7 - AHA! Moment - Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates
[NbConvertApp] ERROR | Error while converting '0 - Geral.ipynb'
Traceback (most recent call last):
  File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/nbconvertapp.py", line 487, in export_single_notebook
    output, resources = self.exporter.from_filename(
  File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/exporters/templateexporter.py", line 390, in from_filename
    return super().from_filename(filename, resources, **kw) #

```

```

type:ignore[return-value]
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/exporters/exporter.py", line 201, in from_filename
    return self.from_file(f, resources=resources, **kw)
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/exporters/templateexporter.py", line 396, in from_file
    return super().from_file(file_stream, resources, **kw) #
type:ignore[return-value]
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/exporters/exporter.py", line 220, in from_file
    return self.from_notebook_node(
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/exporters/pdf.py", line 184, in from_notebook_node
    latex, resources = super().from_notebook_node(nb, resources=resources, **kw)
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/exporters/latex.py", line 92, in from_notebook_node
    return super().from_notebook_node(nb, resources, **kw)
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/exporters/templateexporter.py", line 429, in
from_notebook_node
    output = self.template.render(nb=nb_copy, resources=resources)
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/jinja2/environment.py", line 1295, in render
    self.environment.handle_exception()
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/jinja2/environment.py", line 942, in handle_exception
    raise rewrite_traceback_stack(source=source)
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex/index.tex.j2", line
8, in top-level template code
    ((* extends cell_style *))
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/share/jupyter/nbconvert/templates/lat
ex/style_jupyter.tex.j2", line 176, in top-level template code
    \prompt{(((prompt)))}{(((prompt_color)))}{(((execution_count)))}{(((extra_sp
ace)))}
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex/base.tex.j2", line 7,

```

```

in top-level template code
  ((* extends 'document_contents.tex.j2' -*))
  File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/share/jupyter/nbconvert/templates/lat
ex/document_contents.tex.j2", line 51, in top-level template code
    ((* block figure scoped -*))
    File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex/display_priority.j2",
line 5, in top-level template code
        ((* extends 'null.j2' -*))
        File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex/null.j2", line 30, in
top-level template code
            ((* block body -*))
            File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex/base.tex.j2", line
241, in block 'body'
                ((( super() )))
                File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex/null.j2", line 32, in
block 'body'
                    ((* block any_cell scoped -*))
                    File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de
retenção/.aha_venv/share/jupyter/nbconvert/templates/latex/null.j2", line 85, in
block 'any_cell'
                        ((* block markdowncell scoped-*)) ((* endblock markdowncell -*))
                        File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/share/jupyter/nbconvert/templates/lat
ex/document_contents.tex.j2", line 68, in block 'markdowncell'
                            ((( cell.source | citation2latex | strip_files_prefix |
convert_pandoc('markdown+tex_math_double_backslash', 'json',extra_args=[]) |
resolve_references | convert_explicitly_relative_paths |
convert_pandoc('json','latex'))))
                            File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/filters/pandoc.py", line 36, in convert_pandoc
                                return pandoc(source, from_format, to_format, extra_args=extra_args)
                            File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/utils/pandoc.py", line 50, in pandoc
                                check_pandoc_version()
                            File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-

```

```
packages/nbconvert/utils/pandoc.py", line 98, in check_pandoc_version
    v = get_pandoc_version()
File "/media/tschmitz/hope-keeper/2025_github/px-data-science/7 - AHA! Moment
- Otimização da taxa de retenção/.aha_venv/lib/python3.10/site-
packages/nbconvert/utils/pandoc.py", line 75, in get_pandoc_version
    raise PandocMissing()
nbconvert.utils.pandoc.PandocMissing: Pandoc wasn't found.
Please check that pandoc is installed:
https://pandoc.org/installing.html
[NbConvertApp] Exiting application: jupyter-nbconvert
```

Convertendo report.ipynb → report.pdf

```
[ ]: # Analisando os documentos escreva uma apresentação detalhada do estudo, em
      ↪alto nível.
# Escrever 15 parágrafo
# Não incluir nada de python
# Explicar em alto nível
# Um leigo deve entender do que se trata

# Analisando os documentos descreva a metodologia e os resultados que foram
      ↪obtidos no estudo.
# Escrever 15 parágrafo
# Não incluir nada de python
# Explicar em alto nível
# Um leigo deve entender do que se trata

# Analisando os documentos descreva a análise e discussão e os resultados que
      ↪foram obtidos no estudo.
# Escrever 15 parágrafo
# Não incluir nada de python
# Explicar em alto nível
# Um leigo deve entender do que se trata

# Analisando os documentos descreva as conclusões do estudo
# Escrever 15 parágrafo
# Não incluir nada de python
# Explicar em alto nível
```

```
# Um leigo deve entender do que se trata

# Agora com base nos texto gerado nas etapas anteriores quero um relatório
↳ completo.
# Este relatório deve comunicar os achados para os tomadores de decisão
# Não incluir nada de python
# Explicar em alto nível
# Um leigo deve entender do que se trata
# Estrutura do relatório: Apresentação (4 paragrafos), metodologia e
↳ resultados(8 paragrafos), análise e discussão(4 paragrafos), conclusão(2
↳ paragrafos)
```

```
[ ]: # Analisando os documentos extraia os codigos que geram visualizações.
# Adapte estes codigos para salvar as imagens em PNG na pasta "figuras"
# Nomear estas Figuras em ordem, figura_1.png, figura_2.png, figura_3.png, et
# Retornar um codigo unico que gere e salve todas as imangens em png na pasta
↳ correta
```

```
[ ]: # Analisando o relatório gerado na etapa anterior quero que gere um código
↳ python para gerar um fluxograma
# Incluir neste fluxograma as etapas do estudo
# Incluir os principais resultados
# Incluir os principais achados
# Incluir as principais conclusões
# O fluxograma deve ser salvo na pasta "figuras" como um arquivo png
# O nome do arquivo deve ser "fluxograma.png"
# O código deve ser único e gerar o fluxograma completo
# O código deve ser executável e gerar o fluxograma completo
```

```
# Código de referencia: from graphviz import Digraph# Fluxograma acadêmico
↳ estilizado flow = Digraph('Academic_Flowchart', format='png')# Atributos do
↳ grafo (layout) flow.attr('graph', rankdir='TB', # Direção
↳ top-to-bottom para estilo acadêmico fontsize='12',
↳ fontname='Times New Roman', bgcolor='white', margin='0.
↳ 2')# Atributos dos nós (estilo acadêmico) flow.attr('node',
↳ shape='rectangle', style='rounded,filled',
↳ fillcolor='#F7F9FB', # Tom suave color='#2E4053', # Borda
↳ escura fontname='Times New Roman', fontsize='12',
↳ margin='0.2')# Atributos das arestas (linhas) flow.attr('edge',
↳ color='#2E4053', arrowhead='vee', penwidth='1.5')#
↳ Definição das etapas etapas = [ ('A', 'Início'), ('B', '1. Importação
↳ de Dados\n(pandas)'), ('C', '2. Pré-processamento\n(numpy, unicode)'),
↳ ('D', '3. Estatística Descritiva\n(statsmodels)'), ('E', '4. Modelagem
↳ Preditiva\n(scikit-learn)'), ('F', '5. Visualização de
↳ Dados\n(matplotlib, seaborn)'), ('G', '6. Exportação de
↳ Relatórios\n(openpyxl, tabulate)'), ('H', '7. Deploy &
↳ BQ\n(functions-framework, \ngoogles-cloud-bigquery)'), ('I', 'Conclusão')]#
↳ Cria os nós for key, label in etapas: flow.node(key, label)# Conecta as
↳ etapas em sequência for (src, _), (dst, _) in zip(etapas, etapas[1:]):
↳ flow.edge(src, dst)# Exibe o fluxograma flow
```

```
[ ]: # Agora com base no relatório completo quero que gere um documento HTML para
↳ ele.
# Não usar Python nesta etapa.
# Gerar em HTML puramente.
# Usar as normas da ABNT
# Neste documento incluir o texto adaptado do relatório.
# Incluir também as figuras geradas na etapa anterior. com titulos e descrição.
# Incluir e citar no texto as figuras a partir dos PNGs gerados na etapa
↳ anterior
# Também incluir o fluxograma gerado na etapa anterior.
# Deixar responsivo
```

```
[ ]: from src.libs.lib import *
import pandas as pd
import matplotlib.pyplot as plt
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
df_original = pd.read_csv("src/data/
↳ tabela_ocorrencias_dbpx_com_a_pontuação_academia.csv")
```

```
[20]: import os
from graphviz import Digraph

# Cria a pasta 'figuras' caso não exista
os.makedirs("figuras", exist_ok=True)
```



```

# Fluxograma do estudo
flow = Digraph('Fluxograma_Estudo', format='png')

# Atributos do grafo
flow.attr(
    'graph',
    rankdir='TB',
    fontsize='12',
    fontname='Times New Roman',
    bgcolor='white',
    margin='0.2'
)

# Atributos dos nós
flow.attr(
    'node',
    shape='rectangle',
    style='rounded, filled',
    fillcolor='#F7F9FB',
    color='#2E4053',
    fontname='Times New Roman',
    fontsize='12',
    margin='0.2'
)

# Atributos das arestas
flow.attr(
    'edge',
    color='#2E4053',
    arrowhead='vee',
    penwidth='1.5'
)

# Definição dos nós e rótulos
nodes = [
    ('A', 'Início'),
    ('B', '1. Carregamento de Dados'),
    ('C', '2. Tratamento Inicial'),
    ('D', '3. Atribuição de Pontuação'),
    ('E', '4. Regras para Desistências'),
    ('F', '5. Cálculo Cumulativo e Gatilhos'),
    ('G', '6. Geração de Eventos de Ação'),
    ('H', '7. Análise de Perfil de Eventos'),
    ('I', 'Principais Resultados:\n- Top motoristas para atualização\n- Top_\n- motoristas para comitê'),

```

```

    ('J', 'Principais Achados:\n- Ocorrências frequentes vs graves\n- Eficácia_\n- dos gatilhos'),
    ('K', 'Conclusões:\n- Validação do processo\n- Recomendações de adoção'),
    ('L', 'Fim')
]

# Cria os nós
for key, label in nodes:
    flow.node(key, label)

# Conecta as etapas em sequência
for (src, _), (dst, _) in zip(nodes, nodes[1:]):
    flow.edge(src, dst)

# Gera e salva o arquivo PNG
flow.render(filename='figuras/fluxograma', cleanup=True)

```

[20]: 'figuras/fluxograma.png'

```

[ ]: import os
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.colors import Normalize
from matplotlib import cm

# Cria a pasta 'figuras' caso não exista
os.makedirs("figuras", exist_ok=True)

# Supondo que df_original já esteja carregado e preparado conforme no script_\n- original:
# df_original = pd.read_csv("src/data/\n- tabela_ocorrencias_dbpx_com_a_pontuação_academia.csv")
# e depois de todos os merges e ajustes de pontuação.

# 1. Contagem de ocorrências por descrição
df = df_original.copy()
counts = df['description'].value_counts()
norm = Normalize(vmin=counts.min(), vmax=counts.max())
cmap = cm.get_cmap('Spectral')
colors = cmap(norm(counts.values))

fig, ax = plt.subplots(figsize=(12, 8))
ax.bar(counts.index, counts.values, color=colors)
ax.set_yscale('log')
ax.set_xlabel('Descrição')
ax.set_ylabel('Contagem de Ocorrências (escala log)')
ax.set_title('Contagem de Ocorrências por Descrição')

```

```

plt.xticks(rotation=45, ha='right')
sm = cm.ScalarMappable(cmap=cmap, norm=norm)
sm.set_array([])
fig.colorbar(sm, ax=ax).set_label('Número de Ocorrências')
fig.savefig('figuras/figura_1.png')
plt.close(fig)

# 2. Total de pontos por descrição
df = df_original.copy()
df['Pontuação'] = pd.to_numeric(df['Pontuação'], errors='coerce').fillna(0)
sums = df.groupby('description')['Pontuação'].sum().sort_values(ascending=False)
norm = Normalize(vmin=sums.min(), vmax=sums.max())
colors = cmap(norm(sums.values))

fig, ax = plt.subplots(figsize=(12, 8))
ax.bar(sums.index, sums.values, color=colors)
ax.set_yscale('log')
ax.set_xlabel('Descrição')
ax.set_ylabel('Total de Pontos (escala log)')
ax.set_title('Total de Pontos por Descrição')
plt.xticks(rotation=45, ha='right')
sm = cm.ScalarMappable(cmap=cmap, norm=norm)
sm.set_array([])
fig.colorbar(sm, ax=ax).set_label('Total de Pontos')
fig.savefig('figuras/figura_2.png')
plt.close(fig)

# 3. Top 50 motoristas por envios para atualização
df = df_original.copy()
updates = df[df['description'] == 'Motorista enviado para atualização']
counts = updates['driver_id'].value_counts().head(50)
norm = Normalize(vmin=counts.min(), vmax=counts.max())
colors = cmap(norm(counts.values))

fig, ax = plt.subplots(figsize=(15, 10))
ax.bar(counts.index.astype(str), counts.values, color=colors)
ax.set_xlabel('driver_id')
ax.set_ylabel('Contagem de Envios para Atualização')
ax.set_title('Top 50 Motoristas por Envios para Atualização')
plt.xticks(rotation=90)
sm = cm.ScalarMappable(cmap=cmap, norm=norm)
sm.set_array([])
fig.colorbar(sm, ax=ax).set_label('Número de Envios')
fig.savefig('figuras/figura_3.png')
plt.close(fig)

# 4. Top 50 motoristas por total de pontos

```

```

df = df_original.copy()
df['Pontuação'] = pd.to_numeric(df['Pontuação'], errors='coerce').fillna(0)
points_by_driver = df.groupby('driver_id')['Pontuação'].sum()
    ↪sort_values(ascending=False).head(50)
norm = Normalize(vmin=points_by_driver.min(), vmax=points_by_driver.max())
colors = cmap(norm(points_by_driver.values))

fig, ax = plt.subplots(figsize=(15, 10))
ax.bar(points_by_driver.index.astype(str), points_by_driver.values,
    ↪color=colors)
ax.set_xlabel('driver_id')
ax.set_ylabel('Total de Pontos')
ax.set_title('Top 50 Motoristas por Total de Pontos')
plt.xticks(rotation=90)
sm = cm.ScalarMappable(cmap=cmap, norm=norm)
sm.set_array([])
fig.colorbar(sm, ax=ax).set_label('Total de Pontos')
fig.savefig('figuras/figura_4.png')
plt.close(fig)

# 5. Top 50 motoristas por envios para comitê
df = df_original.copy()
comite = df[df['description'] == 'Motorista enviado para comitê']
counts = comite['driver_id'].value_counts().head(50)
norm = Normalize(vmin=counts.min(), vmax=counts.max())
colors = cmap(norm(counts.values))

fig, ax = plt.subplots(figsize=(15, 10))
ax.bar(counts.index.astype(str), counts.values, color=colors)
ax.set_xlabel('driver_id')
ax.set_ylabel('Contagem de Envios para Comitê')
ax.set_title('Top 50 Motoristas por Envios para Comitê')
plt.xticks(rotation=90)
sm = cm.ScalarMappable(cmap=cmap, norm=norm)
sm.set_array([])
fig.colorbar(sm, ax=ax).set_label('Número de Envios')
fig.savefig('figuras/figura_5.png')
plt.close(fig)

```