aula17

ME315-2S2025

brotto

2025-10-07 12:01:51.129643

Sumário

§ Funny

Exemplos de matemática em LaTeX

$$Q(x) = x_1^2 + 3x_1x_2 + 1x_2^2 = \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} 2 & 1.5 \\ 1.5 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = x'Ax$$

§ Python

```
## Inicia sessão python
   venv_python = "temp.venv"
   if(!virtualenv_exists(venv_python)){ virtualenv_create(venv_python) }

   py_install("pandas", envname = venv_python)
```

```
## Using virtual environment "temp.venv" ...
```

```
   py_install("polars", envname = venv_python)
```

```
## Using virtual environment "temp.venv" ...
```

```
   py_install("pyarrow", envname = venv_python)
```

```
## Using virtual environment "temp.venv" ...
```

```
   py_install("matplotlib", envname = venv_python)
```

```
## Using virtual environment "temp.venv" ...
```

```
   use_virtualenv(venv_python)
```

```
import pandas as pd
import polars as pl
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt

# Definir todos os nomes das colunas
novos_nomes = ["age","workclass","fnlwgt","education","education-num","marital-status","occupation","rel

# Ler e renomear de uma vez
df = pl.read_csv(
    "renda_adulta.csv.gz",
    has_header=False,
    new_columns=novos_nomes
)

print("Dados carregados e colunas renomeadas!")
```

```
## Dados carregados e colunas renomeadas!
```

```
# Tipos de dados de cada coluna
print("\nTipos detalhados por coluna:")
```

```
##
## Tipos detalhados por coluna:
```

```
for coluna, tipo in df.schema.items():
    print(f"Coluna '{coluna}': {tipo}")
```

```
## Coluna 'age': Int64
## Coluna 'workclass': String
## Coluna 'fnlwgt': Int64
## Coluna 'education': String
## Coluna 'education-num': Int64
## Coluna 'marital-status': String
## Coluna 'occupation': String
## Coluna 'relationship': String
## Coluna 'race': String
## Coluna 'sex': String
## Coluna 'capital-gain': Int64
## Coluna 'capital-loss': Int64
## Coluna 'hours-per-week': Int64
## Coluna 'native-country': String
## Coluna 'income': String
```

```
# Dimensões
print("Dimensões do DataFrame:")
```

```
## Dimensões do DataFrame:
```

```
print(f"Shape: {df.shape}")
```

```
## Shape: (32561, 15)
```

```
print(f"Número de linhas: {df.height}")
```

```
## Número de linhas: 32561
```

```
print(f"Número de colunas: {df.width}")
```

```
## Número de colunas: 15
```

```
print(f"Total de elementos: {df.height * df.width}")
```

```
## Total de elementos: 488415
```

```
# Contagem por categoria de renda
contagem_renda = df.group_by('income').agg(
    pl.len().alias('quantidade')
).sort('quantidade', descending=True)

print("1. Distribuição por faixa salarial:")
```

```
## 1. Distribuição por faixa salarial:
```

```
print(contagem_renda)
```

```
## shape: (2, 2)
##
##   income   quantidade
##   ---      ---
##   str      u32
##
```

```
##   <=50K    24720
##   >50K     7841
##
```

```
# Formato alternativo mais explicativo
acima_50k = df.filter(pl.col('income') == '>50K').height
abaixo_50k = df.filter(pl.col('income') == '<=50K').height

print(f"\nPessoas com renda >50K: {acima_50k}")
```

```
##
## Pessoas com renda >50K: 7841
```

```
print(f"Pessoas com renda <=50K: {abaixo_50k}")
```

```
## Pessoas com renda <=50K: 24720
```

```
print(f"Total: {acima_50k + abaixo_50k}")
```

```
## Total: 32561
```

```
# Criar o objeto renda_lango no formato longo
renda_longo = df.unpivot(
    index=['age', 'workclass', 'fnlwgt', 'education', 'education-num',
             'marital-status', 'occupation', 'relationship', 'race', 'sex',
             'hours-per-week', 'native-country', 'income'],
    on=['capital-gain', 'capital-loss'],
    variable_name='tipo',
    value_name='Valor'
)

print("\n2. Dataset em formato longo:")
```

```
##
## 2. Dataset em formato longo:
```

```
print(renda_longo.head())
```

```
## shape: (5, 15)
##
##   age   workclass         fnlwgt   education   …   native-country   income   tipo          Valor
##   ---   ---               ---      ---             ---              ---      ---           ---
##   i64   str               i64      str             str              str      str           i64
##
##   39    State-gov         77516    Bachelors   …   United-States    <=50K    capital-
gain   2174
##   50    Self-emp-not-inc  83311    Bachelors   …   United-States    <=50K    capital-
gain   0
##   38    Private           215646   HS-grad     …   United-States    <=50K    capital-
gain   0
##   53    Private           234721   11th        …   United-States    <=50K    capital-
gain   0
##   28    Private           338409   Bachelors   …   Cuba             <=50K    capital-
gain   0
##
```

```
print(f"Dimensões do renda_longo: {renda_longo.shape}")
```

```
## Dimensões do renda_longo: (65122, 15)
```

```
# Médias de horas por semana por classe salarial
medias_horas = df.group_by('income').agg(
    pl.mean('hours-per-week').alias('media_horas_semana')
).sort('media_horas_semana', descending=True)

print("\n3. Médias de horas trabalhadas por classe salarial:")
```

```
##
## 3. Médias de horas trabalhadas por classe salarial:
```

```
print(medias_horas)
```

```
## shape: (2, 2)
##
##  income   media_horas_semana
##  ---      ---
##  str      f64
##
##  >50K     45.473026
##  <=50K    38.84021
##
```

```
# Contagem de pessoas por ocupação
pessoas_por_profissao = df.group_by('occupation').agg(
    pl.len().alias('numero_pessoas')
).sort('numero_pessoas', descending=True)

print("\n4. Número de pessoas por profissão:")
```

```
##
## 4. Número de pessoas por profissão:
```

```
print(pessoas_por_profissao)
```

```
## shape: (15, 2)
##
##  occupation        numero_pessoas
##  ---               ---
##  str               u32
##
##  Prof-specialty    4140
##  Craft-repair      4099
##  Exec-managerial   4066
##  Adm-clerical      3770
##  Sales             3650
##  …                 …
##  Farming-fishing   994
##  Tech-support      928
##  Protective-serv   649
##  Priv-house-serv   149
##  Armed-Forces      9
##
```

```
# Top 10 profissões
print("\nTop 10 profissões com mais pessoas:")
```

```
##
## Top 10 profissões com mais pessoas:
```

```
print(pessoas_por_profissao.head(10))
```

```
## shape: (10, 2)
##
##   occupation          numero_pessoas
##   ---                 ---
##   str                 u32
##
##   Prof-specialty      4140
##   Craft-repair        4099
##   Exec-managerial     4066
##   Adm-clerical        3770
##   Sales               3650
##   Other-service       3295
##   Machine-op-inspct   2002
##   ?                   1843
##   Transport-moving    1597
##   Handlers-cleaners   1370
##
```

```
# Preparar dados para o gráfico
dados_grafico = df.group_by('income').agg(
    pl.mean('hours-per-week').alias('media_horas')
).sort('media_horas')

print("\n5. Dados para o gráfico de barras:")
```

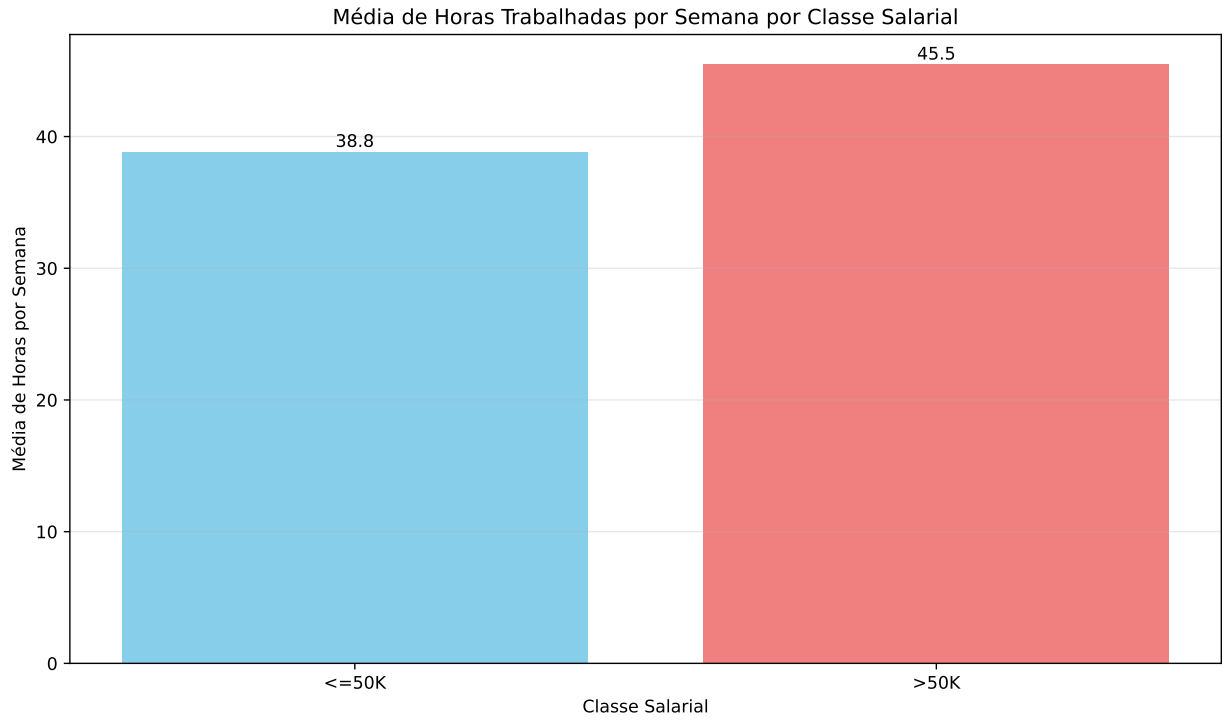
```
##
## 5. Dados para o gráfico de barras:
```

```
print(dados_grafico)
```

```
## shape: (2, 2)
##
##   income   media_horas
##   ---      ---
##   str      f64
##
##   <=50K    38.84021
##   >50K     45.473026
##
```

```
# Se quiser visualizar diretamente, podemos usar matplotlib
try:
    import matplotlib.pyplot as plt

    # Converter para pandas apenas para plotagem
    dados_plot = dados_grafico.to_pandas()

    plt.figure(figsize=(10, 6))
    plt.bar(dados_plot['income'], dados_plot['media_horas'], color=['skyblue', 'lightcoral'])
    plt.title('Média de Horas Trabalhadas por Semana por Classe Salarial')
    plt.xlabel('Classe Salarial')
    plt.ylabel('Média de Horas por Semana')
```

```
    plt.grid(axis='y', alpha=0.3)

    # Adicionar valores nas barras
    for i, v in enumerate(dados_plot['media_horas']):
        plt.text(i, v + 0.1, f'{v:.1f}', ha='center', va='bottom')

    plt.tight_layout()
    plt.show()

except ImportError:
    print("Matplotlib não instalado. Instale com: pip install matplotlib")
```




```
# Análise de discriminação salarial por gênero
discriminacao_genero = df.group_by(['sex', 'income']).agg(
    pl.len().alias('quantidade'),
    pl.mean('hours-per-week').alias('media_horas_trabalhadas'),
    pl.mean('age').alias('media_idade'),
    pl.mean('education-num').alias('media_educacao')
).sort(['sex', 'income'])

print("\nDESAFIO: Análise de discriminação por gênero:")
```

```
##
## DESAFIO: Análise de discriminação por gênero:
```

```
print(discriminacao_genero)
```

```
## shape: (4, 6)
##
##   sex      income   quantidade   media_horas_trabalhadas   media_idade   media_educacao
##   ---      ---      ---          ---                       ---           ---
```

```
##   str      str      u32         f64                     f64           f64
##
##   Female   <=50K    9592        35.916701               36.210801     9.820475
##   Female   >50K     1179        40.426633               42.12553      11.787108
##   Male     <=50K    15128       40.693879               37.147012     9.452142
##   Male     >50K     6662        46.366106               44.625788     11.580606
##
```

```
# Proporções por gênero
proporcoes_genero = df.group_by('sex').agg(
    (pl.col('income') == '>50K').mean().alias('proporcao_acima_50k'),
    pl.len().alias('total')
)

print("\nProporção de pessoas com renda >50K por gênero:")
```

```
##
## Proporção de pessoas com renda >50K por gênero:
```

```
print(proporcoes_genero)
```

```
## shape: (2, 3)
##
##   sex      proporcao_acima_50k   total
##   ---      ---                   ---
##   str      f64                   u32
##
##   Male     0.305737              21790
##   Female   0.109461              10771
##
```

```
# Análise mais detalhada: renda por gênero e profissão
analise_detalhada = df.filter(pl.col('occupation').is_not_null()).group_by(['sex', 'occupation']).agg(
    (pl.col('income') == '>50K').mean().alias('proporcao_alta_renda'),
    pl.len().alias('amostra'),
    pl.mean('hours-per-week').alias('media_horas')
).filter(pl.col('amostra') > 50)  # Filtrar profissões com amostra significativa

print("\nAnálise detalhada por gênero e profissão (amostra > 50):")
```

```
##
## Análise detalhada por gênero e profissão (amostra > 50):
```

```
print(analise_detalhada.sort('proporcao_alta_renda', descending=True).head(10))
```

```
## shape: (10, 5)
##
##   sex      occupation         proporcao_alta_renda   amostra   media_horas
##   ---      ---                ---                    ---       ---
##   str      str                f64                    u32       f64
##
##   Male     Exec-managerial    0.580667               2907      46.371173
##   Male     Prof-specialty     0.561524               2625      44.096762
##   Male     Tech-support       0.410345               580       40.713793
##   Male     Sales              0.374948               2387      44.223712
##   Male     Protective-serv    0.350785               573       43.446771
##   Female   Prof-specialty     0.254125               1515      39.423762
```

```
##   Female   Exec-managerial    0.241588              1159      41.517688
##   Male     Adm-clerical       0.239254              1233      39.240065
##   Male     Craft-repair       0.23446               3877      42.443642
##   Male     Transport-moving   0.20637               1507      45.130723
##
```

```
## Encerra sessão
   virtualenv_remove(venv_python)
```

```
## Virtual environment "temp.venv" removed.
```