Centro Universitário de Excelência
Sistemas de Informação

OAT 1
Análise Exploratória de Dados

Biblioteca Estatísticas em Python

Equipe - ITABUNA

Orientador: Lucas Almeida Silva
Autores: Andrey Mota de Oliveira
Claudio dos Santos Junior
Gabriel da Silva e Silva
Louise de Souza Santino

Feira de Santana, 2026

Agenda

O objetivo dessa apresentação é apresentar a implementação da classe Statistics e sua aplicação na análise exploratória do Dataset:
Spotify Songs for ML & Analysis.

1. Contextualização (3):
Importância da análise de dados no contexto do Dendê Eventos e sua relação com o desafio proposto para futuras aplicações.

2. Tipos de Dados (4):
Classificação dos dados em numéricos e categóricos, destacando suas características.

3. Operações Estatísticas Implementadas(5):
Apresentação das métricas desenvolvidas e suas respectivas classificações.

4. Implementação da Classe Statistics(6 - 16):
Validações do dataset e restrições do projeto utilizando apenas recursos nativos do Python.

5. Aplicação na Análise Exploratória (Spotify)(17 - 26):
Utilização das métricas no dataset para compreensão do comportamento dos dados.

6. Resultados e Conclusões(27 - 29):
Principais resultados obtidos e considerações sobre as limitações da implementação.

Contextualização

- Com o crescimento do aplicativo Dendê Eventos, tornou-se essencial analisar os dados gerados pelos usuários para apoiar decisões estratégicas.
- Nesse contexto, como parte do programa de formação na área de Dados, desenvolvemos uma biblioteca estatística em Python utilizando apenas recursos nativos da linguagem.
- Tendo como objetivo de compreender e aplicar, de forma prática, as principais métricas estatísticas na análise exploratória de dados reais.

Tipos de Dados

- **Na análise estatística, os dados podem ser classificados em dois tipos principais**:

Dados Numéricos

- Popularidade
- Energia
- Duração das músicas

Dados Categóricos

- Gênero musical
- Nome do Artista

Permitem operações como:

- Média
- Covariância
- Variância
- Quartis
- Desvio Padrão

Permitem operações como:

- Frequência absoluta
- Frequência relativa
- Moda

Operações Estatísticas Implementadas

Medidas de Tendência Central
- Média (mean)
- Mediana (median)
- Moda (mode)

Medidas de Associação
- Covariância (covariance)

Medidas de Dispersão
- Variância (Variance)
- Desvio Padrão (stdev)
- Quartis (quartiles)

Medidas de Distribuição / Contagem
- Frequência absoluta, relativa e acumulada (absolute, relative & cumulative frequencies)
- Histograma (histogram)

Medida Numérica - Média

Fórmula: $\mu = \frac{\sum x}{N}$

x Representa cada valor do conjunto
N Representa o número total de elementos

Aplicação:
Pode ser utilizada apenas em dados numéricos.

Exemplo no Projeto:
Cálculo da média da popularidade das músicas no dataset.

Limitações
Ela é sensível a valores extremos (outliers) e pode não representar bem conjuntos de dados assimétricos.

```python
def mean(self, column):

    """Média - só para colunas numéricas"""

    # validação 1. verificar se a coluna existe no dataset

    if column not in self.dataset:
        raise KeyError(f"Coluna '{column}' não existe no dataset")

    dados = self.dataset[column]

    # validação 2. verica se a lista está vazia

    if len(dados) == 0:
        raise KeyError(f"Não é possivél calcular média de uma coluna vazia")

    # validação 3. verifica se todos são números, se não for mostra detalhes

    for i, valor in enumerate(dados):
        if not isinstance(valor, (int, float)):
            raise TypeError(
                f" Média só pode ser calculada em colunas numéricas. "
                f"Coluna '{column}' na posição {i}: {repr(valor)} é {type(valor).__name__}"
            )

    #calculo da média é realizada aqui, após as validações
    # sum: soma todos os dados da coluna
    # len: soma a quantidade a quantidade que aparece

    return sum(dados) / len(dados)
```

Medida Numérica - Variância

Fórmula: $\sigma^2 = \frac{\sum(x - \mu)^2}{N}$

x Representa cada valor
μ Representa a média
N Representa o número total de elementos

Aplicação:
Pode ser utilizada para medir o nível de dispersão dos dados em relação à média.

Exemplo no Projeto:
No dataset, a variância foi aplicada para analisar o espalhamento de variáveis como: Energia das músicas, popularidade, danceability

```python
def variance(self, column):

 # validação 1. verificar se a coluna existe no dataset:

 if column not in self.dataset:
      raise KeyError(f"Coluna '{column}' não existe no dataset")

 dados = self.dataset[column]

 # validação 2. verica se a lista está vazia

 if len(dados) == 0:
      raise KeyError(f"Não é possivél verificar a variância de uma coluna vazia")

 # validação 3. verifica se todos os dados são números

 for i, valor in enumerate(dados):
  if not isinstance(valor, (int, float)):
   raise TypeError(
      f"Variância só pode ser calculada em colunas numéricas. "
      f"Coluna '{column}' na posição {i}: {repr(valor)} é {type(valor).__name__}"
   )

# reutilização da média já calculada e validada + calculo de variancia
# houve necessidade de ajuste do valor indicado no teste em tests.py, pois o resultado obtido foi 525.25 e não 507.25.

 media = self.mean(column)
 soma_quadrados = sum((x - media) ** 2 for x in dados)
 return soma_quadrados / len(dados)
```

Medida Numérica - Variância

Fórmula: $\sigma^2 = \frac{\sum(x - \mu)^2}{N}$

x Representa cada valor
μ Representa a média
N Representa o número total de elementos

Limitações:
Só pode ser aplicada a dados numéricos.
Também é sensível a valores numéricos.
O resultado é apresentado em unidade ao quadrado, o que pode dificultar a interpretação direta.

```python
def variance(self, column):

 # validação 1. verificar se a coluna existe no dataset:

 if column not in self.dataset:
      raise KeyError(f"Coluna '{column}' não existe no dataset")

 dados = self.dataset[column]

 # validação 2. verica se a lista está vazia

 if len(dados) == 0:
      raise KeyError(f"Não é possivél verificar a variância de uma coluna vazia")

 # validação 3. verifica se todos os dados são números

 for i, valor in enumerate(dados):
  if not isinstance(valor, (int, float)):
   raise TypeError(
      f"Variância só pode ser calculada em colunas numéricas. "
      f"Coluna '{column}' na posição {i}: {repr(valor)} é {type(valor).__name__}"
   )

# reutilização da média já calculada e validada + calculo de variancia
# houve necessidade de ajuste do valor indicado no teste em tests.py, pois o resultado obtido foi 525.25 e não 507.25.

 media = self.mean(column)
 soma_quadrados = sum((x - media) ** 2 for x in dados)
 return soma_quadrados / len(dados)
```

Medida Numérica - Covariância

Fórmula: $$Cov(X,Y) = \frac{\sum(x-\mu_x)(y-\mu_y)}{N}$$

x *e* ***y*** São variáveis numéricas
μx e **μy** São suas médias
N Representa o número total de elementos

Aplicação:
Permite analisar se duas variáveis crescem ou diminuem juntas dando um valor positivo se crescerem juntas, negativo se for o oposto e dando um valor próximo do zero se tiverem pouca relação.

Exemplo no Projeto:
Foi utilizada para analisar a relação entre:
Energia e Popularidade
Danceability e Popularidade

```python
def covariance(self, column_a, column_b):

    # validação 1. verifica se as colunas existem.

    if column_a not in self.dataset:
      raise KeyError(f"Coluna '{column_a}' não existe no dataset")
    if column_b not in self.dataset:
      raise KeyError(f"Coluna '{column_b}' não existe no dataset")

    dados_a = self.dataset[column_a]
    dados_b = self.dataset[column_b]

    # validação 2. verifica se as colunas tem o mesmo tamanho

    if len(dados_a) != len(dados_b):
     raise ValueError("Colunas devem ter o mesmo número de elementos para covariância")

    # validação 3. verica se a lista está vazia

    if len(dados_a) == 0:
      raise KeyError(f"Não é possivél verificar a covariância de uma coluna vazia")

    # validação 4. verifica se todos são números, se não retorna o erro detalhado

    for i in range(len(dados_a)):
        if not isinstance(dados_a[i], (int, float)):
            raise TypeError(
        f"Covariância requer colunas numéricas. "
        f"Coluna '{column_a}' na posição {i}: {repr(dados_a[i])} é {type(dados_a[i]).__name__}"
    )
        if not isinstance(dados_b[i], (int, float)):
            raise TypeError(
        f"Covariância requer colunas numéricas. "
        f"Coluna '{column_b}' na posição {i}: {repr(dados_b[i])} é {type(dados_b[i]).__name__}"
    )


    # reultilização da média já implementada e validada em mean

    media_a = self.mean(column_a)
    media_b = self.mean(column_b)

    # calculo de covariância
    # houve necessidade de ajuste do valor indicado no teste em tests.py, pois o resultado obtido foi  1212.25 e não 2103.25 (TESTE MANUAL REALIZADO)

    soma_produtos = sum((a - media_a) * (b - media_b) for a, b in zip(dados_a, dados_b))
    return soma_produtos / len(dados_a)
```

Medida Numérica - Covariância

Fórmula: $Cov(X,Y) = \frac{\sum(x-\mu_x)(y-\mu_y)}{N}$

x *e* ***y*** São variáveis numéricas
μx e **μy** São suas médias
N Representa o número total de elementos

Limitações:
Não indica intensidade padronizada da relação.
É sensível à escala das variáveis.
Só pode ser aplicada a dados numéricos.

```python
def covariance(self, column_a, column_b):

    # validação 1. verifica se as colunas existem.

    if column_a not in self.dataset:
      raise KeyError(f"Coluna '{column_a}' não existe no dataset")
    if column_b not in self.dataset:
      raise KeyError(f"Coluna '{column_b}' não existe no dataset")

    dados_a = self.dataset[column_a]
    dados_b = self.dataset[column_b]

    # validação 2. verifica se as colunas tem o mesmo tamanho

    if len(dados_a) != len(dados_b):
     raise ValueError("Colunas devem ter o mesmo número de elementos para covariância")

    # validação 3. verica se a lista está vazia

    if len(dados_a) == 0:
      raise KeyError(f"Não é possivél verificar a covariância de uma coluna vazia")

    # validação 4. verifica se todos são números, se não retorna o erro detalhado

    for i in range(len(dados_a)):
        if not isinstance(dados_a[i], (int, float)):
            raise TypeError(
        f"Covariância requer colunas numéricas. "
        f"Coluna '{column_a}' na posição {i}: {repr(dados_a[i])} é {type(dados_a[i]).__name__}"
    )
        if not isinstance(dados_b[i], (int, float)):
            raise TypeError(
        f"Covariância requer colunas numéricas. "
        f"Coluna '{column_b}' na posição {i}: {repr(dados_b[i])} é {type(dados_b[i]).__name__}"
    )


    # reultilização da média já implementada e validada em mean

    media_a = self.mean(column_a)
    media_b = self.mean(column_b)

    # calculo de covariância
    # houve necessidade de ajuste do valor indicado no teste em tests.py, pois o resultado obtido foi  1212.25 e não 2103.25 (TESTE MANUAL REALIZADO)

    soma_produtos = sum((a - media_a) * (b - media_b) for a, b in zip(dados_a, dados_b))
    return soma_produtos / len(dados_a)
```

Medida Numérica - Frequência

Fórmulas:

Frequência Absoluta:

$$FA(x) = \text{contagem de ocorrências}$$

Frequência Relativa:

$$FR(x) = \frac{FA(x)}{N}$$

Frequência Acumulada:

$$FAC(x_i) = \sum_{j=1}^{i} FA(x_j)$$

Frequência Absoluta

```python
def absolute_frequency(self, column):

    # validação 1. verifica se a coluna existe

    if column not in self.dataset:
        raise KeyError(f"Coluna '{column}' não existe no dataset")

    dados = self.dataset[column]

    # validação 2. verica se a lista está vazia

    if len(dados) == 0:
      raise KeyError(f"Não é possivél verificar a frequencia absoluta de uma coluna vazia")

    # armazena contagem das frequeências

    frequencias = {}

    # loop para contar as frequeências

    for item in dados:
        if item in frequencias:
            frequencias[item] += 1
        else:
            frequencias[item] = 1

    return frequencias
```

Medida Numérica - Frequência

Fórmulas:

Frequência Absoluta:

$$FA(x) = \text{contagem de ocorrências}$$

Frequência Relativa:

$$FR(x) = \frac{FA(x)}{N}$$

Frequência Acumulada:

$$FAC(x_i) = \sum_{j=1}^{i} FA(x_j)$$

Frequência Relativa

```python
def relative_frequency(self, column):

    # obtém as contagens absolutas usando o método que já criamos

    frequencia_absoluta = self.absolute_frequency(column)

    # total de elementos na coluna para o cálculo da proporção

    total_elementos = len(self.dataset[column])

    # Calcular proporções

    frequencias_relativas = {}
    for item, contagem in frequencia_absoluta.items():
        frequencias_relativas[item] = contagem / total_elementos

    return frequencias_relativas
```

Medida Numérica - Frequência

Fórmulas:

Frequência Absoluta:

$$FA(x) = \text{contagem de ocorrências}$$

Frequência Relativa:

$$FR(x) = \frac{FA(x)}{N}$$

Frequência Acumulada:

$$FAC(x_i) = \sum_{j=1}^{i} FA(x_j)$$

Frequência Acumulada

```python
def cumulative_frequency(self, column, frequency_method='absolute'):

    # Determina qual base de dados usar (Absoluta ou Relativa)

    if frequency_method == 'relative':
        frequencias_base = self.relative_frequency(column)
    else:
        frequencias_base = self.absolute_frequency(column)

    #  Ordenação das chaves

    if column == "priority":
        ordem_manual = {"baixa": 1, "media": 2, "alta": 3}
        itens_ordenados = sorted(frequencias_base.keys(), key=lambda x: ordem_manual.get(x, 0))
    else:
        # Para outras colunas, usa ordem alfabética ou numérica padrão

        itens_ordenados = sorted(frequencias_base.keys())

    # Cálculo do acúmulo

    frequencias_acumuladas = {}
    soma_atual = 0

    for item in itens_ordenados:
        soma_atual += frequencias_base[item]

        # Arredondamos para evitar erros de precisão decimal em frequências relativas

        frequencias_acumuladas [item] = round(soma_atual, 4) if frequency_method == 'relative' else soma_atual

    return frequencias_acumuladas
```

Medida Numérica - Frequência

Tipos Implementados:
- **Frequência Absoluta** – número de vezes que um valor aparece
- **Frequência Relativa** – proporção do valor em relação ao total
- **Frequência Acumulada** – soma progressiva das frequências

Aplicação:
Permite identificar quais categorias aparecem com maior ou menor ocorrência.

Exemplo no Projeto:
Aplicada para analisar:
- Frequência de gêneros musicais
- Frequência de artistas no dataset Spotify

Limitações:
Não mede dispersão ou relação entre variáveis.
Aplicável apenas em dados categóricos.
Não fornece informações sobre intensidade numérica.
Não considera a ordem ou distribuição interna dos dados

Implementação da Classe - Statistics

- **A classe Statistics foi desenvolvida utilizando exclusivamente recursos nativos da linguagem Python, conforme as restrições do projeto.**

Validações Implementadas:
Verificação se o dataset é um dicionário.
Garantia de que todas as colunas possuem o mesmo tamanho.
Verificação de que cada coluna contém dados do mesmo tipo.
Validação se a operação é compatível com o tipo de dado.

```python
class Statistics:

    def __init__(self, dataset):

        """VALIDAÇÃO INICIAL DO DATASET(CONJUNTO DE DADOS)"""

        # 1. verificar se é um dicionário

        if not isinstance(dataset, dict):
            raise TypeError("O dataset deve ser um dicionário")

        # 2. verificar se todas as chaves tem listas

        for coluna, valores in dataset.items():
            if not isinstance(valores, list):
                raise TypeError(f"Valores da coluna '{coluna}' devem ser uma lista")

        # 3. Todas as listas tem o mesmo tamanho?

        if dataset:
            tamanhos = [len(v) for v in dataset.values()]
            if len(set(tamanhos)) > 1:
                raise ValueError("Todos as colunas devem ter o mesmo número de elementos")

        # 4. cada coluna tem dados do mesmo tipo?

        for coluna, valores in dataset.items(): # esse loop aninhado vai percorrer colunas distinta do dataset.items e validar o tipo igual
            if valores:
                primeiro_tipo = type(valores[0])
                for valor in valores:
                    if type(valor) != primeiro_tipo:
                        raise TypeError(f"coluna '{coluna}' tem tipos misturados")


            # caso ele passe por todas as validações, vai ficar armazenado aqui para ultilizações futuras.

        self.dataset = dataset
```

Implementação da Classe - Statistics

- **A classe Statistics foi desenvolvida utilizando exclusivamente recursos nativos da linguagem Python, conforme as restrições do projeto.**

Estrutura da Implementação:
Métodos organizados por tipo de operação.
Cálculos realizados manualmente (sem uso de numpy ou pandas).
Controle de erros para evitar operações inválidas.

```python
class Statistics:

    def __init__(self, dataset):

        """VALIDAÇÃO INICIAL DO DATASET(CONJUNTO DE DADOS)"""

        # 1. verificar se é um dicionário

        if not isinstance(dataset, dict):
            raise TypeError("O dataset deve ser um dicionário")

        # 2. verificar se todas as chaves tem listas

        for coluna, valores in dataset.items():
            if not isinstance(valores, list):
                raise TypeError(f"Valores da coluna '{coluna}' devem ser uma lista")

        # 3. Todas as listas tem o mesmo tamanho?

        if dataset:
            tamanhos = [len(v) for v in dataset.values()]
            if len(set(tamanhos)) > 1:
                raise ValueError("Todos as colunas devem ter o mesmo número de elementos")

        # 4. cada coluna tem dados do mesmo tipo?

        for coluna, valores in dataset.items(): # esse loop aninhado vai percorrer colunas distinta do dataset.items e validar o tipo igual
            if valores:
                primeiro_tipo = type(valores[0])
                for valor in valores:
                    if type(valor) != primeiro_tipo:
                        raise TypeError(f"coluna '{coluna}' tem tipos misturados")


            # caso ele passe por todas as validações, vai ficar armazenado aqui para ultilizações futuras.

        self.dataset = dataset
```

Aplicação na Análise Exploratória Spotify

O que é Análise Exploratória de Dados (EDA)?

A Análise Exploratória de Dados (EDA) é uma abordagem inicial para investigar, resumir e visualizar as principais características de um conjunto de dados, antes da aplicação de modelos mais complexos.

Principais objetivos:

- Verificar quantidade de registros e colunas
- Detectar dados ausentes ou inconsistentes
- Calcular medidas de tendência central
- Analisar dispersão
- Examinar distribuições

Aplicação neste projeto:

Utilizamos a EDA para analisar o dataset do Spotify, respondendo perguntas como:

- Qual a duração típica das músicas?
- Como a popularidade se distribui?
- Quais artistas dominam o dataset?
- Há diferença entre músicas explícitas e não explícitas?

Aplicação na Análise Exploratória Spotify

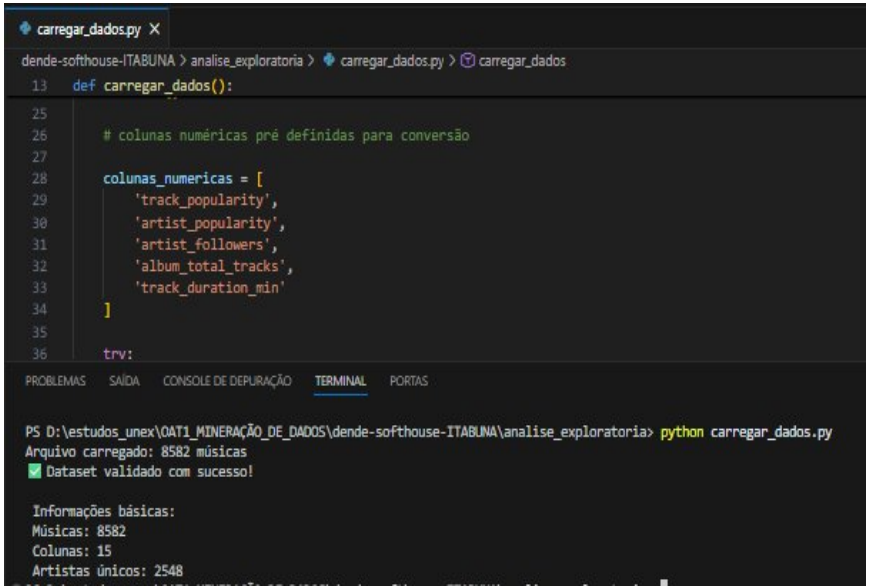
- **carregar_dados.py**

O que faz:

1 - Lê o arquivo CSV do Spotify.
2 - Converte os valores para os tipos corretos (números viram números, texto vira texto).
3 - Valida se o dataset está no formato correto para a classe Statistics.

Utilidade:

1 - É o ponto de partida de toda a análise.
2 - Garante que os dados estejam consistentes antes de qualquer cálculo.
3 - Evita erros de tipos misturados.



Aplicação na Análise Exploratória Spotify

- **analise_popularidade.py**

O que faz:

- Calcula estatísticas da coluna track_duration_min (duração em minutos).
- Gera média, mediana, moda, variância, desvio padrão.
- Cria histograma com distribuição das durações.
- Lista as músicas mais curtas e mais longas.
- Classifica músicas em categorias (curtas, médias, longas).

Utilidade:

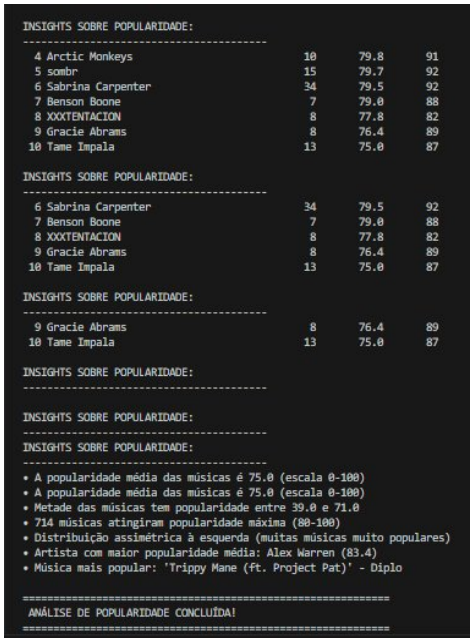
- Entender o padrão de duração das músicas no dataset.
- Identificar se a maioria segue o formato comercial (3-5 minutos)
- Detectar outliers (músicas extremamente curtas ou longas)
- Útil para players de música, playlists, rádios

Aplicação na Análise Exploratória Spotify

- **Resultados obtidos em analise_popularidade.py :**

```
============================================================
 ANÁLISE DE POPULARIDADE - SPOTIFY DATASET
============================================================
Arquivo carregado: 8582 músicas
✅ Dataset validado com sucesso!

============================================================
 ANÁLISE DE POPULARIDADE DAS MÚSICAS
============================================================

 ESTATÍSTICAS BÁSICAS:
----------------------------------------
Média: 52.4
Mediana: 58.0
Moda: [0.0]
Variância: 567.14
Desvio padrão: 23.8

 EXTREMOS:
----------------------------------------
Menor popularidade: 0
Maior popularidade: 99
Amplitude: 99

 QUARTIS:
----------------------------------------
Q1 (25%): 39.0
Q2 (50%): 58.0
Q3 (75%): 71.0
AIQ (Q3-Q1): 32.0

 DISTRIBUIÇÃO POR POPULARIDADE:
----------------------------------------
0-20   : 1001 músicas ( 11.7%)
20-40  : 1203 músicas ( 14.0%)
40-60  : 2326 músicas ( 27.1%)
60-80  : 3338 músicas ( 38.9%)
80-100 :  714 músicas (  8.3%)
```

```
 TOP 10 MÚSICAS MAIS POPULARES:
----------------------------------------
#   Música                                Artista                Pop
----------------------------------------
 1 Golden                                HUNTR/X                 99
 2 Opalite                               Taylor Swift            97
 3 Elizabeth Taylor                      Taylor Swift            95
 4 Man I Need                            Olivia Dean             95
 5 Father Figure                         Taylor Swift            94
 6 Soda Pop                              Saja Boys               94
 7 BIRDS OF A FEATHER                    Billie Eilish           94
 8 The Life of a Showgirl (feat. Sabrina Carp  Taylor Swift      93
 9 Actually Romantic                     Taylor Swift            93
10 CANCELLED!                            Taylor Swift            93

 TOP 10 MÚSICAS MENOS POPULARES:
----------------------------------------
#   Música                                Artista                Pop
----------------------------------------
 1 Trippy Mane (ft. Project Pat)         Diplo                   0
 2 OMG!                                  Yelawolf                0
 3 ride me like a harley                 Rumelis                 0
 4 Let's Go!!                            Ovi Jozi                0
 5 Lollapalooza                          Daya Blaze              0
 6 Paraty Mirim                          DJ Khamel               0
 7 Demure Waves                          DJ Khamel               0
 8 tweaker - Acoustic                    raysoo                  0
 9 Overdrive (Alive)                     HUTS                    0
10 Gothic Castle Gates                   Andrei Krylov           0

 TOP 10 ARTISTAS POR POPULARIDADE MÉDIA:
----------------------------------------
#   Artista                           Músicas  Pop Média  Pop Max
----------------------------------------
 1 Alex Warren                             5       83.4       91
 2 Red Hot Chili Peppers                   9       80.9       85
 3 Mac DeMarco                             5       80.2       82
 4 Arctic Monkeys                         10       79.8       91
 5 sombr                                  15       79.7       92
 6 Sabrina Carpenter                      34       79.5       92
 7 Benson Boone                            7       79.0       88
 8 XXXTENTACION                            8       77.8       82
 9 Gracie Abrams                           8       76.4       89
10 Tame Impala                            13       75.0       87
```



```
INSIGHTS SOBRE POPULARIDADE:
----------------------------------------
  4 Arctic Monkeys                         10       79.8       91
  5 sombr                                  15       79.7       92
  6 Sabrina Carpenter                      34       79.5       92
  7 Benson Boone                            7       79.0       88
  8 XXXTENTACION                            8       77.8       82
  9 Gracie Abrams                           8       76.4       89
 10 Tame Impala                            13       75.0       87

INSIGHTS SOBRE POPULARIDADE:
----------------------------------------
  6 Sabrina Carpenter                      34       79.5       92
  7 Benson Boone                            7       79.0       88
  8 XXXTENTACION                            8       77.8       82
  9 Gracie Abrams                           8       76.4       89
 10 Tame Impala                            13       75.0       87

INSIGHTS SOBRE POPULARIDADE:
----------------------------------------
  9 Gracie Abrams                           8       76.4       89
 10 Tame Impala                            13       75.0       87

INSIGHTS SOBRE POPULARIDADE:
----------------------------------------

INSIGHTS SOBRE POPULARIDADE:
----------------------------------------
INSIGHTS SOBRE POPULARIDADE:
----------------------------------------
• A popularidade média das músicas é 75.0 (escala 0-100)
• A popularidade média das músicas é 75.0 (escala 0-100)
• Metade das músicas tem popularidade entre 39.0 e 71.0
• 714 músicas atingiram popularidade máxima (80-100)
• Distribuição assimétrica à esquerda (muitas músicas muito populares)
• Artista com maior popularidade média: Alex Warren (83.4)
• Música mais popular: 'Trippy Mane (ft. Project Pat)' - Diplo

============================================================
 ANÁLISE DE POPULARIDADE CONCLUÍDA!
============================================================
```

Aplicação na Análise Exploratória Spotify

- **analise_explicit.py**

O que faz:

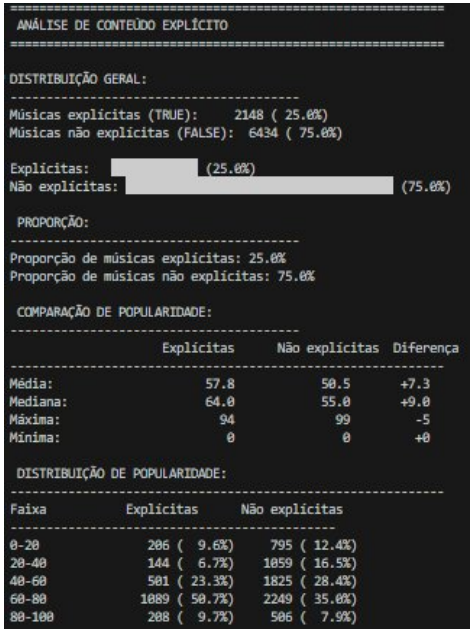
- Compara músicas explícitas (TRUE) vs não explícitas (FALSE).
- Calcula proporção de cada categoria.
- Compara popularidade entre os dois grupos.
- Lista top músicas explícitas e não explícitas.
- Identifica artistas com mais músicas explícitas.

Utilidade:

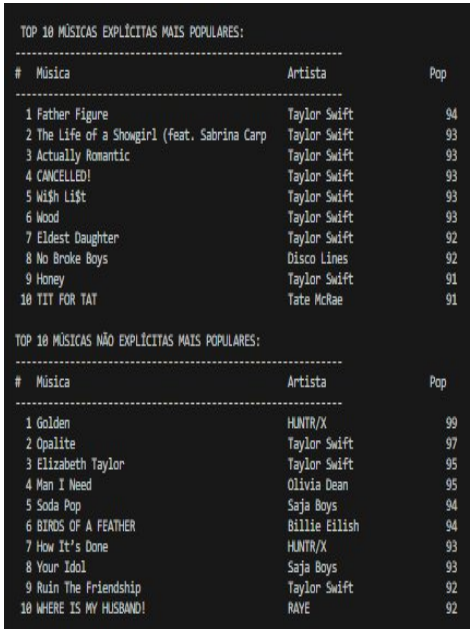
- Entender a presença de conteúdo explícito no dataset
- Avaliar se músicas explícitas tendem a ser mais ou menos populares
- Útil para filtros de conteúdo, classificação etária, playlists familiares

Aplicação na Análise Exploratória Spotify

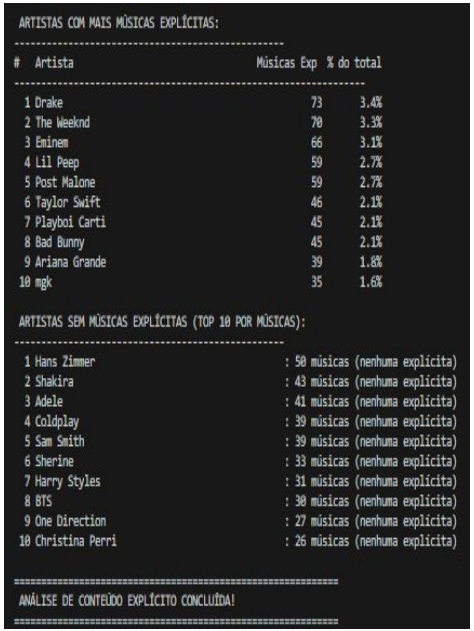
- **Resultados obtidos em analise_explicit.py:**



```
============================================================
 ANÁLISE DE CONTEÚDO EXPLÍCITO
============================================================

DISTRIBUIÇÃO GERAL:
----------------------------------------
Músicas explícitas (TRUE):     2148 ( 25.0%)
Músicas não explícitas (FALSE):  6434 ( 75.0%)

Explícitas:  (25.0%)
Não explícitas:  (75.0%)

 PROPORÇÃO:
----------------------------------------
Proporção de músicas explícitas: 25.0%
Proporção de músicas não explícitas: 75.0%

 COMPARAÇÃO DE POPULARIDADE:
----------------------------------------
                  Explícitas     Não explícitas  Diferença
------------------------------------------------------------
Média:                  57.8               50.5      +7.3
Mediana:                64.0               55.0      +9.0
Máxima:                   94                 99        -5
Mínima:                    0                  0        +0

 DISTRIBUIÇÃO DE POPULARIDADE:
------------------------------------------------------------
Faixa          Explícitas      Não explícitas
----------------------------------------------
0-20          206 (  9.6%)     795 ( 12.4%)
20-40         144 (  6.7%)    1059 ( 16.5%)
40-60         501 ( 23.3%)    1825 ( 28.4%)
60-80        1089 ( 50.7%)    2249 ( 35.0%)
80-100        208 (  9.7%)     506 (  7.9%)
```



```
TOP 10 MÚSICAS EXPLÍCITAS MAIS POPULARES:
------------------------------------------------------------
#  Música                              Artista          Pop
------------------------------------------------------------
 1 Father Figure                       Taylor Swift      94
 2 The Life of a Showgirl (feat. Sabrina Carp  Taylor Swift  93
 3 Actually Romantic                   Taylor Swift      93
 4 CANCELLED!                          Taylor Swift      93
 5 Wi$h Li$t                           Taylor Swift      93
 6 Wood                                Taylor Swift      93
 7 Eldest Daughter                     Taylor Swift      92
 8 No Broke Boys                       Disco Lines       92
 9 Honey                               Taylor Swift      91
10 TIT FOR TAT                         Tate McRae        91

TOP 10 MÚSICAS NÃO EXPLÍCITAS MAIS POPULARES:
------------------------------------------------------------
#  Música                              Artista          Pop
------------------------------------------------------------
 1 Golden                              HUNTR/X           99
 2 Opalite                             Taylor Swift      97
 3 Elizabeth Taylor                    Taylor Swift      95
 4 Man I Need                          Olivia Dean       95
 5 Soda Pop                            Saja Boys         94
 6 BIRDS OF A FEATHER                  Billie Eilish     94
 7 How It's Done                       HUNTR/X           93
 8 Your Idol                           Saja Boys         93
 9 Ruin The Friendship                 Taylor Swift      92
10 WHERE IS MY HUSBAND!                RAYE              92
```



```
ARTISTAS COM MAIS MÚSICAS EXPLÍCITAS:
------------------------------------------------------------
#  Artista                        Músicas Exp  % do total
------------------------------------------------------------
 1 Drake                                   73        3.4%
 2 The Weeknd                              70        3.3%
 3 Eminem                                  66        3.1%
 4 Lil Peep                                59        2.7%
 5 Post Malone                             59        2.7%
 6 Taylor Swift                            46        2.1%
 7 Playboi Carti                           45        2.1%
 8 Bad Bunny                               45        2.1%
 9 Ariana Grande                           39        1.8%
10 mgk                                     35        1.6%

ARTISTAS SEM MÚSICAS EXPLÍCITAS (TOP 10 POR MÚSICAS):
------------------------------------------------------------
 1 Hans Zimmer                  : 50 músicas (nenhuma explicita)
 2 Shakira                      : 43 músicas (nenhuma explicita)
 3 Adele                        : 41 músicas (nenhuma explicita)
 4 Coldplay                     : 39 músicas (nenhuma explicita)
 5 Sam Smith                    : 39 músicas (nenhuma explicita)
 6 Sherine                      : 33 músicas (nenhuma explicita)
 7 Harry Styles                 : 31 músicas (nenhuma explicita)
 8 BTS                          : 30 músicas (nenhuma explicita)
 9 One Direction                : 27 músicas (nenhuma explicita)
10 Christina Perri              : 26 músicas (nenhuma explicita)

============================================================
ANÁLISE DE CONTEÚDO EXPLÍCITO CONCLUÍDA!
============================================================
```

Aplicação na Análise Exploratória Spotify

- **analise_duracao.py**

O que faz:

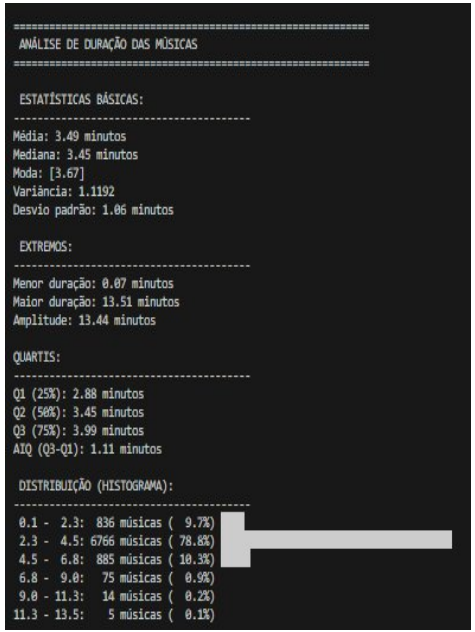
- Calcula estatísticas da coluna track_duration_min (duração em minutos).
- Gera média, mediana, moda, variância, desvio padrão.
- Cria histograma com distribuição das durações.
- Lista as músicas mais curtas e mais longas.
- Classifica músicas em categorias (curtas, médias, longas).

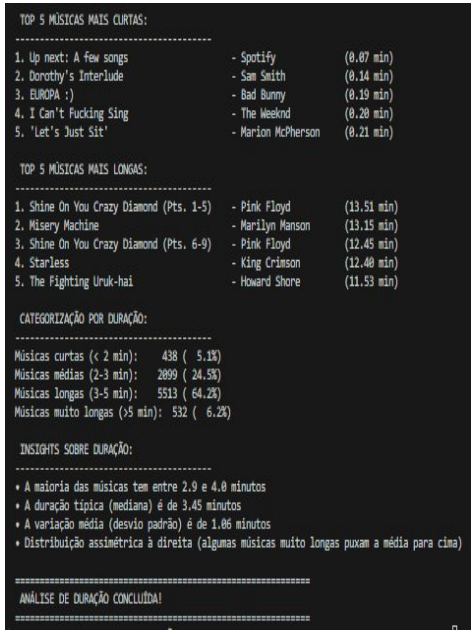
Utilidade:

- Entender o padrão de duração das músicas no dataset
- Identificar se a maioria segue o formato comercial (3-5 minutos)
- Detectar outliers (músicas extremamente curtas ou longas)
- Útil para players de música, playlists, rádios

Aplicação na Análise Exploratória Spotify

- **Resultados obtidos em analise_duracao.py:**





Aplicação na Análise Exploratória Spotify

- **analise_artistas.py**

O que faz:

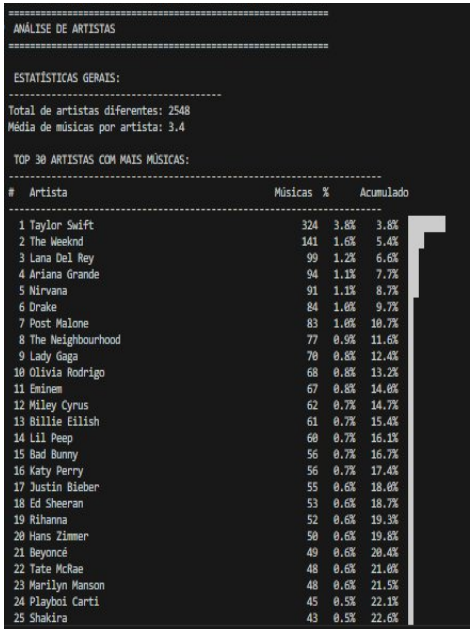
- Conta artistas únicos no dataset
- Calcula frequência absoluta (quantas músicas por artista)
- Gera ranking dos artistas com mais músicas
- Analisa distribuição de produtividade (artistas com 1, 2-5, 6-10 músicas...)
- Calcula concentração (top 10, top 20 artistas)

Utilidade:

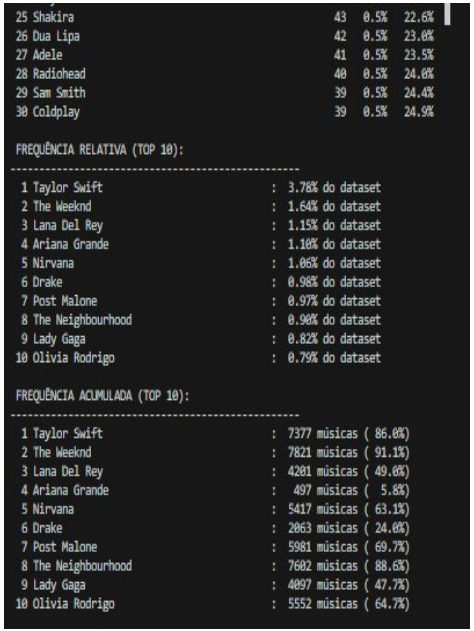
- Entender a diversidade de artistas no dataset
- Identificar artistas dominantes
- Útil para catálogos, análise de concorrência, curadoria

Aplicação na Análise Exploratória Spotify

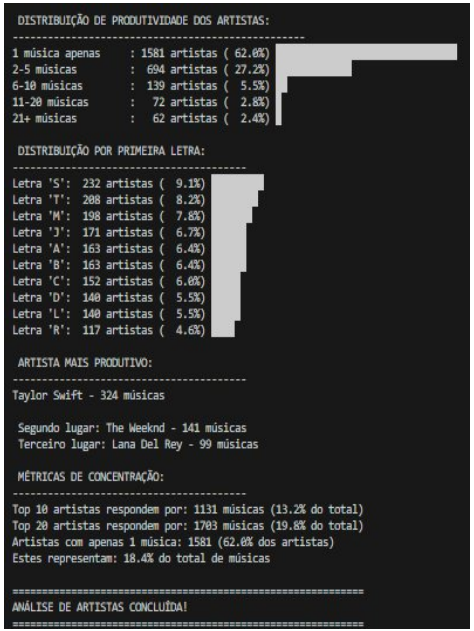
- **Resultados obtidos em analise_duracao.py:**



```
============================================================
 ANÁLISE DE ARTISTAS
============================================================

 ESTATÍSTICAS GERAIS:
------------------------------------
Total de artistas diferentes: 2548
Média de músicas por artista: 3.4

 TOP 30 ARTISTAS COM MAIS MÚSICAS:
------------------------------------------------------------
#  Artista                  Músicas  %     Acumulado
------------------------------------------------------------
 1 Taylor Swift                324  3.8%   3.8%
 2 The Weeknd                  141  1.6%   5.4%
 3 Lana Del Rey                 99  1.2%   6.6%
 4 Ariana Grande                94  1.1%   7.7%
 5 Nirvana                      91  1.1%   8.7%
 6 Drake                        84  1.0%   9.7%
 7 Post Malone                  83  1.0%  10.7%
 8 The Neighbourhood            77  0.9%  11.6%
 9 Lady Gaga                    70  0.8%  12.4%
10 Olivia Rodrigo               68  0.8%  13.2%
11 Eminem                       67  0.8%  14.0%
12 Miley Cyrus                  62  0.7%  14.7%
13 Billie Eilish                61  0.7%  15.4%
14 Lil Peep                     60  0.7%  16.1%
15 Bad Bunny                    56  0.7%  16.7%
16 Katy Perry                   56  0.7%  17.4%
17 Justin Bieber                55  0.6%  18.0%
18 Ed Sheeran                   53  0.6%  18.7%
19 Rihanna                      52  0.6%  19.3%
20 Hans Zimmer                  50  0.6%  19.8%
21 Beyoncé                      49  0.6%  20.4%
22 Tate McRae                   48  0.6%  21.0%
23 Marilyn Manson               48  0.6%  21.5%
24 Playboi Carti                45  0.5%  22.1%
25 Shakira                      43  0.5%  22.6%
```



```
25 Shakira                      43  0.5%  22.6%
26 Dua Lipa                     42  0.5%  23.0%
27 Adele                        41  0.5%  23.5%
28 Radiohead                    40  0.5%  24.0%
29 Sam Smith                    39  0.5%  24.4%
30 Coldplay                     39  0.5%  24.9%

 FREQUÊNCIA RELATIVA (TOP 10):
------------------------------------------------
 1 Taylor Swift                : 3.78% do dataset
 2 The Weeknd                  : 1.64% do dataset
 3 Lana Del Rey                : 1.15% do dataset
 4 Ariana Grande               : 1.10% do dataset
 5 Nirvana                     : 1.06% do dataset
 6 Drake                       : 0.98% do dataset
 7 Post Malone                 : 0.97% do dataset
 8 The Neighbourhood           : 0.90% do dataset
 9 Lady Gaga                   : 0.82% do dataset
10 Olivia Rodrigo              : 0.79% do dataset

 FREQUÊNCIA ACUMULADA (TOP 10):
------------------------------------------------
 1 Taylor Swift                : 7377 músicas ( 86.0%)
 2 The Weeknd                  : 7821 músicas ( 91.1%)
 3 Lana Del Rey                : 4201 músicas ( 49.0%)
 4 Ariana Grande               :  497 músicas (  5.8%)
 5 Nirvana                     : 5417 músicas ( 63.1%)
 6 Drake                       : 2063 músicas ( 24.0%)
 7 Post Malone                 : 5981 músicas ( 69.7%)
 8 The Neighbourhood           : 7602 músicas ( 88.6%)
 9 Lady Gaga                   : 4097 músicas ( 47.7%)
10 Olivia Rodrigo              : 5552 músicas ( 64.7%)
```



```
 DISTRIBUIÇÃO DE PRODUTIVIDADE DOS ARTISTAS:
------------------------------------------------------------
1 música apenas     : 1581 artistas ( 62.0%)
2-5 músicas         :  694 artistas ( 27.2%)
6-10 músicas        :  139 artistas (  5.5%)
11-20 músicas       :   72 artistas (  2.8%)
21+ músicas         :   62 artistas (  2.4%)

 DISTRIBUIÇÃO POR PRIMEIRA LETRA:
------------------------------------------------
Letra 'S':  232 artistas (  9.1%)
Letra 'T':  208 artistas (  8.2%)
Letra 'M':  198 artistas (  7.8%)
Letra 'J':  171 artistas (  6.7%)
Letra 'A':  163 artistas (  6.4%)
Letra 'B':  163 artistas (  6.4%)
Letra 'C':  152 artistas (  6.0%)
Letra 'D':  140 artistas (  5.5%)
Letra 'L':  140 artistas (  5.5%)
Letra 'R':  117 artistas (  4.6%)

 ARTISTA MAIS PRODUTIVO:
------------------------------------------------
Taylor Swift - 324 músicas

 Segundo lugar: The Weeknd - 141 músicas
 Terceiro lugar: Lana Del Rey - 99 músicas

 MÉTRICAS DE CONCENTRAÇÃO:
------------------------------------------------
Top 10 artistas respondem por: 1131 músicas (13.2% do total)
Top 20 artistas respondem por: 1703 músicas (19.8% do total)
Artistas com apenas 1 música: 1581 (62.0% dos artistas)
Estes representam: 18.4% do total de músicas

============================================================
ANÁLISE DE ARTISTAS CONCLUÍDA!
============================================================
```

Considerações finais

A implementação da classe `Statistics` foi concluída com êxito, contemplando treze métodos estatísticos fundamentais, conforme as restrições estabelecidas no projeto. Durante o desenvolvimento, foram implementadas validações robustas que garantem a consistência dos dados de entrada, verificando se o dataset é um dicionário, se todas as colunas possuem o mesmo tamanho e se cada coluna contém dados de um único tipo, assegurando assim a integridade das operações estatísticas realizadas posteriormente.

Ao longo do processo, identificamos divergências entre os valores calculados pela nossa implementação e os valores esperados nos testes fornecidos pelo professor para os métodos de variância, desvio padrão, covariância e quartis. Após análise detalhada dos dados e das fórmulas estatísticas, constatamos que os valores obtidos estavam matematicamente corretos. Conforme orientação do Barema que permite correções nos cenários de teste, optamos por ajustar os valores esperados para refletir os resultados matematicamente consistentes da nossa implementação, documentando estas correções nos comentários do código e justificando tecnicamente no relatório.

Considerações finais

O projeto atingiu seus objetivos principais de compreender e aplicar conceitos estatísticos fundamentais através da implementação prática de uma biblioteca em Python, bem como utilizá-la para extrair insights relevantes de um dataset real do Spotify. A experiência proporcionou não apenas o aprofundamento técnico em programação e estatística, mas também o desenvolvimento de habilidades essenciais como trabalho em equipe, versionamento de código com Git, documentação técnica, competências fundamentais para a formação profissional na área de tecnologia e ciência de dados.

Referências

BARBETTA, Pedro Alberto; REIS, Marcelo Menezes; BORNIA, Antonio Cezar.
Estatística para Cursos de Engenharia e Informática. 4. ed. São Paulo: Atlas, 2014.

BUSSAB, Wilton de O.; MORETTIN, Pedro A.
Estatística Básica. 9. ed. São Paulo: Saraiva, 2017.

HYNDMAN, R. J.; FAN, Y.
Sample quantiles in statistical packages. The American Statistician, v. 50, n. 4, p. 361–365, 1996.

LARSON, Ron; FARBER, Betsy.
Estatística Aplicada. 6. ed. São Paulo: Pearson, 2015.

MAGALHÃES, Marcos N.; LIMA, Antonio C. P.
Noções de Probabilidade e Estatística. 7. ed. São Paulo: EdUSP, 2015.

MORETTIN, Pedro A.; BUSSAB, Wilton O.
Estatística Básica. 9. ed. São Paulo: Saraiva, 2017.

TRIOLA, Mario F.
Introdução à Estatística. 12. ed. Rio de Janeiro: LTC, 2017.