📌 NOTEBOOK 2: MODELAGEM E ANÁLISE DE DADOS COM MODELOS DE NLP/LLM

Inferência de Gênero e Análise de Sentimento em Decisões Judiciais

🔧 1. CONFIGURAÇÃO DO AMBIENTE

1.1 Instalação de Dependência

```
!pip install transformers torch pandas scikit-learn scipy seaborn matplotlib
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.57.0)
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages (2.2.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.12/dist-packages (1.6.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.12/dist-packages (1.16.2)
Requirement already satisfied: seaborn in /usr/local/lib/python3.12/dist-packages (0.13.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (3.10.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers) (3.20.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.35.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.3)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers) (2.32.4)
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.22.1)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.6.2)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch) (4.15.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packages (from torch) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages (from torch) (2025.3.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.80)
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch) (9.10.2.21)
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.4.1)
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch) (11.3.0.4)
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch) (10.3.7.77)
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch) (11.7.1.2)
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch) (12.5.4.2)
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch) (0.7.1)
Requirement already satisfied: nvidia-nccl-cu12==2.27.3 in /usr/local/lib/python3.12/dist-packages (from torch) (2.27.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.85)
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch) (1.11.1.6)
Requirement already satisfied: triton==3.4.0 in /usr/local/lib/python3.12/dist-packages (from torch) (3.4.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.2)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (1.5.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.4.9)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (3.2.5)
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.34.0->transf
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch) (3.0.3)
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.4.3)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2025.10.5)
```

1.2 Importação de Bibliotecas Principais

```
import pandas as pd
import re
import numpy as np
import matplotlib.pyplot as plt
from transformers import pipeline
from itertools import combinations
```

📥 2. CARREGAMENTO DOS DADOS

2.1 Carregar Dataset com Textos Completos

Descrição: Carregue o arquivo `decisoes_extraidas.csv` gerado após scrapping de PDF's e extração do texto de decisão, feitos no NOTEBOOK 1.

Localização: `..\data\notebook1\decisoes_extraidas.csv`

Obs: Carregamento pode demorar um pouco

```
df = pd.read_csv("decisoes_extraidas.csv", sep=';')

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

```
Shape do dataset: (805, 3)
```

	arquivo	numero_processo	decisao_completa
0	00003031620178060215.pdf	0000303-16.2017.8.06.0215	Av Paulo Bastos, 802, Centro - CEP 62620-000, ...
1	00004983720188060127.pdf	0000498-37.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...
2	00006188020188060127.pdf	0000618-80.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...
3	00009657920198060127.pdf	0000965-79.2019.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...
4	00013144420188060151.pdf	0001314-44.2018.8.06.0151	1ª Vara da Comarca de Quixadá Av. Jesus, Maria...

Próximas etapas: Gerar código com `df` | New interactive sheet

🧠 3. EXTRAÇÃO DE ENTIDADES E INFERÊNCIA DE GÊNERO

3.1 Carregamento do Modelo NER (LeNER-BR)

Descrição: Carregamento do modelo LeNER-BR, um BERT fine-tuned para reconhecimento de entidades nomeadas em textos jurídicos brasileiros (identifica nomes de pessoas, organizações, locais)

Hugging Face 🤗: https://huggingface.co/pierreguillou/ner-bert-base-cased-pt-lenerbr

```
from transformers import pipeline

# Modelo NER especializado em textos jurídicos brasileiros
ner_pipeline = pipeline(
    "ner",
    model="pierreguillou/ner-bert-base-cased-pt-lenerbr",
    aggregation_strategy="simple"
)
```

```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
Device set to use cuda:0
```

3.2 Função de Detecção de Título (Juiz/Juíza)

Descrição: Função que busca por padrões de "Juiz" ou "Juíza" nos últimos 2000 caracteres do texto (onde geralmente está a assinatura). Usa regex com normalização para capturar variações ortográficas.

```
def detectar_titulo_juiz(texto):
    """
    Procura no texto ocorrências de 'Juiz' ou 'Juíza' e retorna:
      - 'Juiz'    se a última ocorrência for masculina
      - 'Juíza'   se a última ocorrência for feminina
      - 'N/A'     se não encontrar nada confiável
    """
    if not texto:
        return "N/A"

    # Olhando no fim do texto onde geralmente está a assinatura
    trecho_fim = texto[-2000:]

    # Padrão regex para capturar variações de Juiz/Juíza
    padrao = r"Ju[ií]z(a)?"

    # Busca pelo padrão ignorando capitalização
    judge_search = re.search(padrao, trecho_fim, flags=re.IGNORECASE)

    # Caso não encontrado -> N/A
```

```
    if not judge_search:
        return "N/A"

    # Recupera o termo encontrado
    judge = judge_search.group(0)

    # Decisão: se termina com 'a' → Juíza, senão → Juiz
    if "a" in judge.lower():
        return "Juíza"
    else:
```

3.3 Função de Inferência de Gênero do Juiz

Descrição: Converte o título detectado em categoria de gênero.

```
def inferir_genero_juiz_por_titulo(texto):
    """
    Usa o título para inferir o gênero do juiz(a).
    """
    titulo = detectar_titulo_juiz(texto)
    if titulo == "Juiz":
        return "Masculino"
    elif titulo == "Juíza":
        return "Feminino"
    else:
        return "N/A"
```

3.4 Aplicação das Funções ao Dataset

```
df["titulo_juiz"] = df["decisao_completa"].apply(detectar_titulo_juiz)
df["genero_juiz"] = df["decisao_completa"].apply(inferir_genero_juiz_por_titulo)

df[["numero_processo", "titulo_juiz", "genero_juiz"]].sample(10, random_state=42)
```

	numero_processo	titulo_juiz	genero_juiz
192	0200226-69.2022.8.06.0143	Juiz	Masculino
718	0272599-05.2023.8.06.0001	Juiz	Masculino
168	0200082-88.2023.8.06.0037	Juíza	Feminino
522	0242520-09.2024.8.06.0001	Juiz	Masculino
536	0244698-62.2023.8.06.0001	Juiz	Masculino
791	0291464-13.2022.8.06.0001	Juiz	Masculino
765	0281122-06.2023.8.06.0001	Juíza	Feminino
328	0204029-56.2022.8.06.0112	Juiz	Masculino
218	0200436-05.2022.8.06.0052	Juíza	Feminino
788	0288630-66.2024.8.06.0001	Juiz	Masculino

3.5 Função de Extração de Nomes via NER

Descrição: Função para extração de nomes de pessoas usando NER, com múltiplos filtros:

- Exige pelo menos 2 palavras (nome + sobrenome)
- Remove termos institucionais/genéricos
- Exige mínimo de 3 caracteres

Limitação: Alguns documentos não têm nomes identificáveis nos primeiros 1000 caracteres (cabeçalhos, citações longas, ruído), resultando em "Não identificado".

```
def extrair_pessoas_via_ner(texto):
    """
    Roda NER no texto e retorna lista de entidades do tipo PESSOA,
    filtrando tokens quebrados, exigindo pelo menos 2 palavras
    e ignorando termos institucionais/genéricos.
    """
    if not isinstance(texto, str) or texto.strip() == "":
        return []

    entities = ner_pipeline(texto)
    pessoas = []

    # Lista de termos que não queremos considerar como "Parte" (Pessoa Física)
    termos_proibidos = [
        "justiça pública", "justica publica",
        "ministério público", "ministerio publico",
        "município", "municipio",
```

```
        "estado do", "união", "uniao",
        "fazenda pública", "fazenda publica",
        "defensoria", "procuradoria",
        "banco", "seguradora", "imobiliária"
    ]

    for ent in entities:
        if ent.get("entity_group") != "PESSOA":
            continue

        nome = ent.get("word", "").strip()

        # Ignorar subwords esquisitos tipo '##no'
        if "##" in nome:
            continue

        # Remover caracteres estranhos
        nome_limpo = re.sub(r"[^A-Za-zÁÉÍÓÚÂÊÔÃÕÇáéíóúâêôãõç\s]", "", nome).strip()

        # Pelo menos 2 "palavras" (nome + sobrenome etc.)
        if len(nome_limpo.split()) < 2:
            continue

        # Pelo menos 1 letra "de verdade"
        if not re.search(r"[A-Za-zÁÉÍÓÚÂÊÔÃÕÇáéíóúâêôãõç]", nome_limpo):
            continue

        # Filtro de termos proibidos (institucionais)
        nome_lower = nome_limpo.lower()
        if any(term in nome_lower for term in termos_proibidos):
            continue

        pessoas.append({
            "nome": nome_limpo,
            "start": ent.get("start", None),
            "end": ent.get("end", None),
        })

    return pessoas
```

3.6 Função de Extração de Nomes das Partes (Estratégia Híbrida)

Descrição: Função principal de extração com estratégia híbrida em 3 camadas:

- Regex formal: Busca por "Requerente:", "Autor:", "Paciente:"
- Regex narrativa: Busca por "ajuizada por", "intentada por", "promovida por"
- NER fallback: Usa modelo de NLP quando padrões textuais falham

Importante: Analisa apenas os primeiros 1000 caracteres para otimizar performance.

```
import re

TERMOS_PROIBIDOS = [
    "ministério público", "ministerio publico",
    "justiça pública", "justica publica",
    "estado do", "município", "municipio",
    "união", "uniao", "fazenda pública", "fazenda publica",
    "defensoria", "procuradoria", "secretaria", "prefeitura",
    "tribunal", "comarca", "vara", "juizado",
]

FRASES_RUIDO = [
    "para emendar", "emendar a inicial", "para emenda", "emenda da inicial",
    "a suportar", "suportar dores", "dores de", "forte intensidade",
    "possui histórico", "historico de", "encontra se", "encontra-se",
    "dependente para", "atividades da vida",
]

STOPWORDS_NOME = {
    "para","a","o","as","os","de","da","do","das","dos","e","em","no","na","nos","nas",
    "por","com","sem","ao","aos","à","às"
}

def eh_proibido(nome: str) -> bool:
    if not isinstance(nome, str):
        return True
    n = nome.strip().lower()
    if any(t in n for t in TERMOS_PROIBIDOS):
        return True
    if any(fr in n for fr in FRASES_RUIDO):
        return True
    return False

def limpar_nome(nome: str) -> str:
    if not isinstance(nome, str):
        return ""
```

```
    nome = nome.strip()

    # remove prefixos comuns que vazam
    nome = re.sub(r"(?i)^(e\s+)?(o\s+)?(a\s+)?autor(a)?\b[:\s-]*", "", nome).strip()
    nome = re.sub(r"(?i)^(o\s+)?(a\s+)?requerente\b[:\s-]*", "", nome).strip()
    nome = re.sub(r"(?i)^(o\s+)?(a\s+)?paciente\b[:\s-]*", "", nome).strip()

    # corta em delimitadores
    nome = re.split(r"[\n;,]", nome)[0].strip()

    # remove lixo não-letra
    nome = re.sub(r"[^A-Za-zÁÉÍÓÚÂÊÔÃÕÇáéíóúâêôãõç\s]", " ", nome)
    nome = re.sub(r"\s+", " ", nome).strip()
    return nome

def parece_nome_pessoa(nome: str) -> bool:
    if not isinstance(nome, str):
        return False
    nome = nome.strip()
    parts = [p for p in nome.split() if p]
    if len(parts) < 2:
        return False

    conteudo = [p for p in parts if p.lower() not in STOPWORDS_NOME]
    if len(conteudo) < 2:
        return False

    good = sum((w[:1].isupper() or w.isupper()) for w in conteudo)
    if good < 2:
        return False

    if any((len(w) >= 12 and not (w[:1].isupper() or w.isupper())) for w in conteudo):
        return False

    return True

def extrair_pessoas_via_ner(texto: str, limite_chars: int = 1000):
    """
    Retorna lista de nomes candidatos (strings) vindos do NER,
    já limpos e filtrados.
    """
    if not isinstance(texto, str) or not texto.strip():
        return []

    trecho = texto[:limite_chars]

    # IMPORTANTE: truncation/max_length evita RuntimeError do BERT
    ents = ner_pipeline(trecho)

    candidatos = []
    for ent in ents:
        if ent.get("entity_group") != "PESSOA":
            continue

        nome = limpar_nome(ent.get("word", ""))
        if not nome:
            continue

        if eh_proibido(nome):
            continue

        if not parece_nome_pessoa(nome):
            continue

        candidatos.append(nome)

    # remove duplicados preservando ordem
    vistos = set()
    out = []
    for n in candidatos:
        k = n.lower()
        if k not in vistos:
            vistos.add(k)
            out.append(n)
    return out

def extrair_nome_parte(texto: str, limite_chars: int = 2500) -> str:
    if not isinstance(texto, str) or texto.strip() == "":
        return "Não identificado"

    trecho = texto[:limite_chars]

    # 1) Rótulos formais
    padroes_formais = [
        r"(?im)^\s*Paciente\s*:\s*([^\n]{3,150})",
        r"(?im)^\s*Requerente\s*:\s*([^\n]{3,150})",
        r"(?im)^\s*Autor(?:a)?\s*:\s*([^\n]{3,150})",
        r"(?im)^\s*Parte\s+autora\s*:\s*([^\n]{3,150})",
```

```
        r"(?im)^\s*Demandante\s*:\s*([^\n]{3,150})",
        r"(?im)^\s*Impetrante\s*:\s*([^\n]{3,150})",
        r"(?im)^\s*Exequente\s*:\s*([^\n]{3,150})",
    ]

    for p in padroes_formais:
        m = re.search(p, trecho)
        if m:
            cand = limpar_nome(m.group(1))
            if cand and (not eh_proibido(cand)) and parece_nome_pessoa(cand):
                return cand

    # 2) Narrativos
    padroes_narrativos = [
        r"(?i)\bajuizad[ao]\s+por\s+([^,.;\n]{3,150})",
        r"(?i)\bintentad[ao]\s+por\s+([^,.;\n]{3,150})",
        r"(?i)\bpropost[ao]\s+por\s+([^,.;\n]{3,150})",
        r"(?i)\bpromovid[ao]\s+por\s+([^,.;\n]{3,150})",
    ]

    for p in padroes_narrativos:
        m = re.search(p, trecho)
        if m:
            cand = limpar_nome(m.group(1))
            if cand and (not eh_proibido(cand)) and parece_nome_pessoa(cand):
                return cand

    # 3) Fallback NER
    candidatos = extrair_pessoas_via_ner(trecho, limite_chars=1000)
    if candidatos:
        # escolha robusta: o nome mais longo costuma ser o mais completo
        return max(candidatos, key=lambda s: len(s))

```

3.7 Aplicando no dataframe

```
df["nome_parte"] = df["decisao_completa"].apply(extrair_nome_parte)
```

```
Asking to truncate to max_length but no maximum length is provided and the model has no predefined maximum length. Default to no truncatio
You seem to be using the pipelines sequentially on GPU. In order to maximize efficiency please use a dataset
```

3.8 Porcentagem de nomes das partes não identificadas

```
nao_identificados = (df['nome_parte'] == "Não identificado").mean() * 100
n_nao_identificados = df['nome_parte'].eq("Não identificado").sum()

print(f"Porcentagem de nomes das partes não identificadas: {nao_identificados:.2f}%")
print(f"Quantidade: {n_nao_identificados}")

# Amostra
display(df[["numero_processo", "nome_parte"]].sample(5, random_state=42))
```

```
Porcentagem de nomes das partes não identificadas: 0.62%
Quantidade: 5
```

	numero_processo	nome_parte
192	0200226-69.2022.8.06.0143	MARIA IONEIDE DE ARAÚJO ANDRADE
718	0272599-05.2023.8.06.0001	Floriano Benevides Magalhaes
168	0200082-88.2023.8.06.0037	PEDRO LUCAS ALVES DOS SANTOS
522	0242520-09.2024.8.06.0001	Floriano Benevides Magalhaes
536	0244698-62.2023.8.06.0001	Jose Humberto Cortez Varela

3.9 Removendo registros não identificados do dataset

```
# Removendo registros onde o nome da parte é 'Não identificado'
df = df[df['nome_parte'] != "Não identificado"]

print(f"Registros restantes: {df.shape[0]}")
df.sample(5)
```

```
Registros restantes: 800
```

	arquivo	numero_processo	decisao_completa	titulo_juiz	genero_juiz	nome_parte
790	02910363120228060001.pdf	0291036-31.2022.8.06.0001	26ª Vara Cível (SEJUD 1º Grau) Rua Desembargad...	Juiz	Masculino	LUIS BARROS MONTENEGRO NETO em face da HAPVIDA...
791	02914641320228060001.pdf	0291464-13.2022.8.06.0001	25ª Vara Cível (SEJUD 1º Grau) Rua Desembargad...	Juiz	Masculino	Francisco Ferreira dos Santos
792	02915343020228060001.pdf	0291534-30.2022.8.06.0001	37ª Vara Cível (SEJUD 1º Grau) Rua Desembargad...	Juiz	Masculino	Floriano Benevides Magalhaes
793	02921466520228060001.pdf	0292146-65.2022.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Juíza	Feminino	Maria Clara Guimarães de Figueiredo
794	02940468320228060001.pdf	0294046-83.2022.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Juíza	Feminino	Francisco Claudio de Andrade Lima Filho
795	02964882220228060001.pdf	0296488-22.2022.8.06.0001	10ª Vara Cível (SEJUD 1º Grau) Rua Desembargad...	Juíza	Feminino	PEDRO EVANDRO BANDEIRA LESSA e outro em face d...
796	02965661620228060001.pdf	0296566-16.2022.8.06.0001	14ª Vara de Família (SEJUD 1º Grau) Rua Desemb...	Juiz	Masculino	Adriano José Nascimento de Oliveira em face de...
797	02969914320228060001.pdf	0296991-43.2022.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Juíza	Feminino	Floriano Benevides Magalhaes
798	08000022220228060163.pdf	0800002-22.2022.8.06.0163	2ª Vara da Comarca de São Benedito Av. Tabajar...	Juíza	Feminino	Maria Lívia Gonçalves do Nascimento
799	08000065120238060122.pdf	0800006-51.2023.8.06.0122	Rua Capitão Miguel Dantas, 1000, Centro - CEP ...	Juiz	Masculino	DA FELIPE LEITE
800	08000076520238060177.pdf	0800007-65.2023.8.06.0177	Rua Carlos Antônio Sales, nº 401, Centro - CEP...	Juiz	Masculino	Francisco de Assis Pereira Braga
801	08000088520248060057.pdf	0800008-85.2024.8.06.0057	Rua Coronel Francisco Linhares, S/N, Centro - ...	Juiz	Masculino	CE LIMA DE ALMEIDA
802	08000093520238060177.pdf	0800009-35.2023.8.06.0177	Rua Carlos Antônio Sales, nº 401, Centro - CEP...	Juiz	Masculino	Antônio Alves de Araújo
803	08000118820228060096.pdf	0800011-88.2022.8.06.0096	CEL. GUILHERMINO, S/N, PRAÇA DE CRISTO - CEP 6...	Juíza	Feminino	N CARLOS PONTES MOREIRA
804	08000178120228060133.pdf	0800017-81.2022.8.06.0133	2ª Vara da Comarca de Nova Russas Rua Leonardo...	Juiz	Masculino	Rosa Maria Albuquerque

3.10 Carregamento do Modelo Zero-Shot Classification (Bart-Large-MNLI)

Descrição: Carregamento do modelo facebook/bart-large-mnli, um modelo BART de grande porte treinado no dataset MultiNLI (MNLI), utilizado para classificação de textos em zero-shot para tarefas de classificação, análise de conteúdo e rotulagem. Em nosso caso, a análise do gênero baseado no nome da parte

Hugging Face 🤗: https://huggingface.co/facebook/bart-large-mnli

```
classifier = pipeline("zero-shot-classification", model="facebook/bart-large-mnli")

```

```
Device set to use cuda:0
```

3.11 Função de Inferência de Gênero da Parte (via Nome com Zero-Shot)

```
def identificar_genero(nome):
    """
    Usa zero-shot classification para inferir o gênero a partir do PRIMEIRO nome.
    Retorna 'Masculino' ou 'Feminino'.
    """
    if not isinstance(nome, str) or nome.strip() == "" or nome == "Não identificado":
        return "N/A"

    # Pegar apenas o primeiro nome para evitar ruído dos sobrenomes
    primeiro_nome = nome.strip().split()[0]

    labels = ["Masculino", "Feminino"]

    resultado = classifier(
        primeiro_nome,
```

```
        candidate_labels=labels,
        hypothesis_template="Este nome pertence ao gênero {}."
    )

    # O pipeline retorna ordenado por score decrescente, então o índice 0 é o vencedor
```

3.12 Aplicando a função de indentificação de gênero para a parte

```
df["genero_parte"] = df["nome_parte"].apply(identificar_genero)

df[["numero_processo", "nome_parte", "genero_parte", "titulo_juiz", "genero_juiz"]].head()
```

	numero_processo	nome_parte	genero_parte	titulo_juiz	genero_juiz
0	0000303-16.2017.8.06.0215	José Arteiro de Araújo	Masculino	Juiz	Masculino
1	0000498-37.2018.8.06.0127	Maria Jose Silva de Araujo	Feminino	Juíza	Feminino
2	0000618-80.2018.8.06.0127	Luiz Mateus Santos da Costa	Masculino	Juíza	Feminino
3	0000965-79.2019.8.06.0127	Francisco Roberto Rodrigues da Silva	Masculino	Juíza	Feminino
4	0001314-44.2018.8.06.0151	Y FERREIRA DE OLIVEIRA	Masculino	Juiz	Masculino

3.13 Verificando existência de valores nulos

```
# Contar quantidade de valores nulos (NaN) nas colunas de gênero
nulos_genero = df[['genero_juiz', 'genero_parte']].isna().sum()
print(nulos_genero)

display(df.head(4))
```

```
genero_juiz     0
genero_parte    0
dtype: int64
```

	arquivo	numero_processo	decisao_completa	titulo_juiz	genero_juiz	nome_parte	genero_parte
0	000030316201780602 15.pdf	0000303-16.2017.8.06.0215	Av Paulo Bastos, 802, Centro - CEP 62620-000, ...	Juiz	Masculino	José Arteiro de Araújo	Masculino
1	000049837201880601 27.pdf	0000498-37.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Juíza	Feminino	Maria Jose Silva de Araujo	Feminino
2	000061880201880601 27.pdf	0000618-80.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Juíza	Feminino	Luiz Mateus Santos da Costa	Masculino
3	000096579201980601 27.pdf	0000965-79.2019.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Juíza	Feminino	Francisco Roberto Rodrigues da Silva	Masculino

3.14 Removendo colunas parciais de nome

```
df = df.drop(columns=["nome_parte", "titulo_juiz"])
```

```
display(df)
```

	arquivo	numero_processo	decisao_completa	genero_juiz	genero_parte
0	00003031620178060215.pdf	0000303-16.2017.8.06.0215	Av Paulo Bastos, 802, Centro - CEP 62620-000, ...	Masculino	Masculino
1	00004983720188060127.pdf	0000498-37.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Feminino	Feminino
2	00006188020188060127.pdf	0000618-80.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Feminino	Masculino
3	00009657920198060127.pdf	0000965-79.2019.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Feminino	Masculino
4	00013144420188060151.pdf	0001314-44.2018.8.06.0151	1ª Vara da Comarca de Quixadá Av. Jesus, Maria...	Masculino	Masculino
...	...	...	...	...	...
800	08000076520238060177.pdf	0800007-65.2023.8.06.0177	Rua Carlos Antônio Sales, nº 401, Centro - CEP...	Masculino	Masculino
801	08000088520248060057.pdf	0800008-85.2024.8.06.0057	Rua Coronel Francisco Linhares, S/N, Centro - ...	Masculino	Masculino
802	08000093520238060177.pdf	0800009-35.2023.8.06.0177	Rua Carlos Antônio Sales, nº 401, Centro - CEP...	Masculino	Masculino
803	08000118820228060096.pdf	0800011-88.2022.8.06.0096	CEL. GUILHERMINO, S/N, PRAÇA DE CRISTO - CEP 6...	Feminino	Masculino
804	08000178120228060133.pdf	0800017-81.2022.8.06.0133	2ª Vara da Comarca de Nova Russas Rua Leonardo...	Masculino	Feminino

800 rows × 5 columns

Próximas etapas: Gerar código com df | New interactive sheet

3.15 Salvando resultados parciais

```
df.to_csv("decisoes_com_genero.csv", index=False, sep=";")
```

🎯 4. CLASSIFICAR SENTENÇAS EM PROCEDENTES E IMPROCEDENTES

↳ 8 células ocultas

📚 5. PRÉ-PROCESSAMENTO DO TEXTO

- https://garyeckstein.com/clean-text-for-data-analysis/

↳ 29 células ocultas

🤖 6. ANÁLISE DE SENTIMENTO COM LLM

Observação: Pule para o resultados gerados no item `6.8`, apenas veja os resultados gerados.

6.1 Carregando o csv com decisões pré-processadas e gêneros inferidos

```
df = pd.read_csv("decisoes_processadas_generfdo.csv", sep=";")
```

6.2 Instalando OpenRouter

Descrição: O OpenRouter funciona como um hall de modelos, facilitando o uso de diferentes LLMs.

```
%pip install llama-index-llms-openrouter
!pip install llama-index
```

```
Collecting llama-index-llms-openrouter
  Downloading llama_index_llms_openrouter-0.4.2-py3-none-any.whl.metadata (2.3 kB)
Collecting llama-index-core<0.15,>=0.13.0 (from llama-index-llms-openrouter)
  Downloading llama_index_core-0.14.10-py3-none-any.whl.metadata (2.5 kB)
Collecting llama-index-llms-openai-like<0.6,>=0.5.0 (from llama-index-llms-openrouter)
  Downloading llama_index_llms_openai_like-0.5.3-py3-none-any.whl.metadata (1.1 kB)
Requirement already satisfied: aiohttp<4,>=3.8.6 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-ind
Collecting aiosqlite (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading aiosqlite-0.22.0-py3-none-any.whl.metadata (4.3 kB)
Collecting banks<3,>=2.2.0 (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading banks-2.2.0-py3-none-any.whl.metadata (12 kB)
Collecting dataclasses-json (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading dataclasses_json-0.6.7-py3-none-any.whl.metadata (25 kB)
Collecting deprecated>=1.2.9.3 (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading deprecated-1.3.1-py2.py3-none-any.whl.metadata (5.9 kB)
Collecting dirtyjson<2,>=1.0.8 (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading dirtyjson-1.0.8-py3-none-any.whl.metadata (11 kB)
Collecting filetype<2,>=1.2.0 (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading filetype-1.2.0-py2.py3-none-any.whl.metadata (6.5 kB)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-ind
Requirement already satisfied: httpx in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-llms-ope
Collecting llama-index-workflows!=2.9.0,<3,>=2 (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading llama_index_workflows-2.11.5-py3-none-any.whl.metadata (4.7 kB)
Requirement already satisfied: nest-asyncio<2,>=1.5.8 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llar
Requirement already satisfied: networkx>=3.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-
Requirement already satisfied: nltk>3.8.1 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-llm
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-llms-ope
Requirement already satisfied: pillow>=9.0.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-
Requirement already satisfied: platformdirs in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-l
Requirement already satisfied: pydantic>=2.8.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-inde
Requirement already satisfied: pyyaml>=6.0.1 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-
Requirement already satisfied: requests>=2.31.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-ind
Collecting setuptools>=80.9.0 (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading setuptools-80.9.0-py3-none-any.whl.metadata (6.6 kB)
Requirement already satisfied: sqlalchemy>=1.4.49 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy[asyncio]>=1.4.49->llama-inde
Requirement already satisfied: tenacity!=8.4.0,<10.0.0,>=8.2.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.
Requirement already satisfied: tiktoken>=0.7.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-inde
Requirement already satisfied: tqdm<5,>=4.66.1 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-inde
Requirement already satisfied: typing-extensions>=4.5.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->l
Collecting typing-inspect>=0.8.0 (from llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading typing_inspect-0.9.0-py3-none-any.whl.metadata (1.5 kB)
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15,>=0.13.0->llama-index-llms-ope
Collecting llama-index-llms-openai<0.7,>=0.6.0 (from llama-index-llms-openai-like<0.6,>=0.5.0->llama-index-llms-openrouter)
  Downloading llama_index_llms_openai-0.6.12-py3-none-any.whl.metadata (3.0 kB)
Requirement already satisfied: transformers<5,>=4.37.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-llms-openai-like<0.6,>
Requirement already satisfied: aiohappyeyeballs>=2.5.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-cor
Requirement already satisfied: aiosignal>=1.4.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15,
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15,>=0
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15,
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15
Collecting griffe (from banks<3,>=2.2.0->llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading griffe-1.15.0-py3-none-any.whl.metadata (5.2 kB)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from banks<3,>=2.2.0->llama-index-core<0.15,>=0.13.0->ll
Requirement already satisfied: openai<3,>=1.108.1 in /usr/local/lib/python3.12/dist-packages (from llama-index-llms-openai<0.7,>=0.6.0->ll
Collecting llama-index-instrumentation>=0.1.0 (from llama-index-workflows!=2.9.0,<3,>=2->llama-index-core<0.15,>=0.13.0->llama-index-llms-o
  Downloading llama_index_instrumentation-0.4.2-py3-none-any.whl.metadata (1.1 kB)
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages (from nltk>3.8.1->llama-index-core<0.15,>=0.13.0->llama-ind
Requirement already satisfied: joblib in /usr/local/lib/python3.12/dist-packages (from nltk>3.8.1->llama-index-core<0.15,>=0.13.0->llama-i
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.12/dist-packages (from nltk>3.8.1->llama-index-core<0.15,>=0.13.0
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.12/dist-packages (from pydantic>=2.8.0->llama-index-core<0
Requirement already satisfied: pydantic-core==2.33.2 in /usr/local/lib/python3.12/dist-packages (from pydantic>=2.8.0->llama-index-core<0.
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.12/dist-packages (from pydantic>=2.8.0->llama-index-core
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests>=2.31.0->llama-index-cor
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests>=2.31.0->llama-index-core<0.15,>=0.1
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests>=2.31.0->llama-index-core<0.15
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests>=2.31.0->llama-index-core<0.15
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy>=1.4.49->sqlalchemy[asyncio]>=1.4.49
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers<5,>=4.37.0->llama-index-llms-openai-
Requirement already satisfied: huggingface-hub<1.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from transformers<5,>=4.37.0->llam
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers<5,>=4.37.0->llama-index-llms-o
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers<5,>=4.37.0->llama
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers<5,>=4.37.0->llama-index-llm
Collecting mypy-extensions>=0.3.0 (from typing-inspect>=0.8.0->llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading mypy_extensions-1.1.0-py3-none-any.whl.metadata (1.1 kB)
Collecting marshmallow<4.0.0,>=3.18.0 (from dataclasses-json->llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading marshmallow-3.26.1-py3-none-any.whl.metadata (7.3 kB)
Requirement already satisfied: anyio in /usr/local/lib/python3.12/dist-packages (from httpx->llama-index-core<0.15,>=0.13.0->llama-index-l
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.12/dist-packages (from httpx->llama-index-core<0.15,>=0.13.0->llama
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.12/dist-packages (from httpcore==1.*->httpx->llama-index-core<0.15,>=0.
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.34.0->transfo
Requirement already satisfied: distro<2,>=1.7.0 in /usr/local/lib/python3.12/dist-packages (from openai<3,>=1.108.1->llama-index-llms-open
Requirement already satisfied: jiter<1,>=0.4.0 in /usr/local/lib/python3.12/dist-packages (from openai<3,>=1.108.1->llama-index-llms-opena
Requirement already satisfied: sniffio in /usr/local/lib/python3.12/dist-packages (from openai<3,>=1.108.1->llama-index-llms-openai<0.7,>=0
Collecting colorama>=0.4 (from griffe->banks<3,>=2.2.0->llama-index-core<0.15,>=0.13.0->llama-index-llms-openrouter)
  Downloading colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->banks<3,>=2.2.0->llama-index-core<0
Downloading llama_index_llms_openrouter-0.4.2-py3-none-any.whl (4.5 kB)
Downloading llama_index_core-0.14.10-py3-none-any.whl (11.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.9/11.9 MB 25.9 MB/s eta 0:00:00
Downloading llama_index_llms_openai_like-0.5.3-py3-none-any.whl (4.7 kB)
Downloading banks-2.2.0-py3-none-any.whl (29 kB)
Downloading deprecated-1.3.1-py2.py3-none-any.whl (11 kB)
Downloading dirtyjson-1.0.8-py3-none-any.whl (25 kB)
Downloading filetype-1.2.0-py2.py3-none-any.whl (19 kB)
Downloading llama_index_llms_openai-0.6.12-py3-none-any.whl (26 kB)
```

```
Downloading llama_index_workflows-2.11.5-py3-none-any.whl (91 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 92.0/92.0 kB 16.0 MB/s eta 0:00:00
Downloading setuptools-80.9.0-py3-none-any.whl (1.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.2/1.2 MB 98.4 MB/s eta 0:00:00
Downloading typing_inspect-0.9.0-py3-none-any.whl (8.8 kB)
Downloading aiosqlite-0.22.0-py3-none-any.whl (17 kB)
Downloading dataclasses_json-0.6.7-py3-none-any.whl (28 kB)
Downloading llama_index_instrumentation-0.4.2-py3-none-any.whl (15 kB)
Downloading marshmallow-3.26.1-py3-none-any.whl (50 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.9/50.9 kB 6.3 MB/s eta 0:00:00
Downloading mypy_extensions-1.1.0-py3-none-any.whl (5.0 kB)
Downloading griffe-1.15.0-py3-none-any.whl (150 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 150.7/150.7 kB 24.1 MB/s eta 0:00:00
Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: filetype, dirtyjson, setuptools, mypy-extensions, marshmallow, deprecated, colorama, aiosqlite, typing-insp
  Attempting uninstall: setuptools
    Found existing installation: setuptools 75.2.0
    Uninstalling setuptools-75.2.0:
      Successfully uninstalled setuptools-75.2.0
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of
ipython 7.34.0 requires jedi>=0.16, which is not installed.
Successfully installed aiosqlite-0.22.0 banks-2.2.0 colorama-0.4.6 dataclasses-json-0.6.7 deprecated-1.3.1 dirtyjson-1.0.8 filetype-1.2.0 g
WARNING: The following packages were previously imported in this runtime:
  [_distutils_hack]
You must restart the runtime in order to use newly installed versions.
```

RESTART SESSION

```
Collecting llama-index
  Downloading llama_index-0.14.10-py3-none-any.whl.metadata (13 kB)
Collecting llama-index-cli<0.6,>=0.5.0 (from llama-index)
  Downloading llama_index_cli-0.5.3-py3-none-any.whl.metadata (1.4 kB)
Requirement already satisfied: llama-index-core<0.15.0,>=0.14.10 in /usr/local/lib/python3.12/dist-packages (from llama-index) (0.14.10)
Collecting llama-index-embeddings-openai<0.6,>=0.5.0 (from llama-index)
  Downloading llama_index_embeddings_openai-0.5.1-py3-none-any.whl.metadata (400 bytes)
Collecting llama-index-indices-managed-llama-cloud>=0.4.0 (from llama-index)
  Downloading llama_index_indices_managed_llama_cloud-0.9.4-py3-none-any.whl.metadata (3.7 kB)
Requirement already satisfied: llama-index-llms-openai<0.7,>=0.6.0 in /usr/local/lib/python3.12/dist-packages (from llama-index) (0.6.12)
Collecting llama-index-readers-file<0.6,>=0.5.0 (from llama-index)
  Downloading llama_index_readers_file-0.5.5-py3-none-any.whl.metadata (5.7 kB)
Collecting llama-index-readers-llama-parse>=0.4.0 (from llama-index)
  Downloading llama_index_readers_llama_parse-0.5.1-py3-none-any.whl.metadata (3.1 kB)
Requirement already satisfied: nltk>3.8.1 in /usr/local/lib/python3.12/dist-packages (from llama-index) (3.9.1)
Requirement already satisfied: aiohttp<4,>=3.8.6 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama
Requirement already satisfied: aiosqlite in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-index)
Requirement already satisfied: banks<3,>=2.2.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-i
Requirement already satisfied: dataclasses-json in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-
Requirement already satisfied: deprecated>=1.2.9.3 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llar
Requirement already satisfied: dirtyjson<2,>=1.0.8 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llar
Requirement already satisfied: filetype<2,>=1.2.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llam
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-
Requirement already satisfied: httpx in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-index) (0.2
Requirement already satisfied: llama-index-workflows!=2.9.0,<3,>=2 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0
Requirement already satisfied: nest-asyncio<2,>=1.5.8 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->
Requirement already satisfied: networkx>=3.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-ind
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-index) (2.0
Requirement already satisfied: pillow>=9.0.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-ind
Requirement already satisfied: platformdirs in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-inde
Requirement already satisfied: pydantic>=2.8.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-i
Requirement already satisfied: pyyaml>=6.0.1 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-ind
Requirement already satisfied: requests>=2.31.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-
Requirement already satisfied: setuptools>=80.9.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llam
Requirement already satisfied: sqlalchemy>=1.4.49 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy[asyncio]>=1.4.49->llama-inde
Requirement already satisfied: tenacity!=8.4.0,<10.0.0,>=8.2.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=
Requirement already satisfied: tiktoken>=0.7.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-i
Requirement already satisfied: tqdm<5,>=4.66.1 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-i
Requirement already satisfied: typing-extensions>=4.5.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10
Requirement already satisfied: typing-inspect>=0.8.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->l
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from llama-index-core<0.15.0,>=0.14.10->llama-index) (1.1
Requirement already satisfied: openai>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-embeddings-openai<0.6,>=0.5.0->l
Collecting deprecated>=1.2.9.3 (from llama-index-core<0.15.0,>=0.14.10->llama-index)
  Downloading Deprecated-1.2.18-py2.py3-none-any.whl.metadata (5.7 kB)
Collecting llama-cloud==0.1.35 (from llama-index-indices-managed-llama-cloud>=0.4.0->llama-index)
  Downloading llama_cloud-0.1.35-py3-none-any.whl.metadata (1.2 kB)
Requirement already satisfied: certifi>=2024.7.4 in /usr/local/lib/python3.12/dist-packages (from llama-cloud==0.1.35->llama-index-indices
Requirement already satisfied: beautifulsoup4<5,>=4.12.3 in /usr/local/lib/python3.12/dist-packages (from llama-index-readers-file<0.6,>=0
Requirement already satisfied: defusedxml>=0.7.1 in /usr/local/lib/python3.12/dist-packages (from llama-index-readers-file<0.6,>=0.5.0->ll
Requirement already satisfied: pandas<2.3.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-readers-file<0.6,>=0.5.0->llama-i
Collecting pypdf<7,>=6.1.3 (from llama-index-readers-file<0.6,>=0.5.0->llama-index)
  Downloading pypdf-6.4.2-py3-none-any.whl.metadata (7.1 kB)
Collecting striprtf<0.0.27,>=0.0.26 (from llama-index-readers-file<0.6,>=0.5.0->llama-index)
  Downloading striprtf-0.0.26-py3-none-any.whl.metadata (2.1 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.88-py3-none-any.whl.metadata (6.6 kB)
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages (from nltk>3.8.1->llama-index) (8.3.0)
Requirement already satisfied: joblib in /usr/local/lib/python3.12/dist-packages (from nltk>3.8.1->llama-index) (1.5.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.12/dist-packages (from nltk>3.8.1->llama-index) (2024.11.6)
Requirement already satisfied: aiohappyeyeballs>=2.5.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-cor
Requirement already satisfied: aiosignal>=1.4.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15.0
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15.0,>
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15.0
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp<4,>=3.8.6->llama-index-core<0.15
Requirement already satisfied: griffe in /usr/local/lib/python3.12/dist-packages (from banks<3,>=2.2.0->llama-index-core<0.15.0,>=0.14.10-
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from banks<3,>=2.2.0->llama-index-core<0.15.0,>=0.14.10-
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.12/dist-packages (from beautifulsoup4<5,>=4.12.3->llama-index-reade
```

```
Requirement already satisfied: anyio in /usr/local/lib/python3.12/dist-packages (from httpx->llama-index-core<0.15.0,>=0.14.10->llama-index
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.12/dist-packages (from httpx->llama-index-core<0.15.0,>=0.14.10->ll
Requirement already satisfied: idna in /usr/local/lib/python3.12/dist-packages (from httpx->llama-index-core<0.15.0,>=0.14.10->llama-index
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.12/dist-packages (from httpcore==1.*->httpx->llama-index-core<0.15.0,>=
Requirement already satisfied: llama-index-instrumentation>=0.1.0 in /usr/local/lib/python3.12/dist-packages (from llama-index-workflows!=
Collecting llama-cloud-services>=0.6.88 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.88-py3-none-any.whl.metadata (3.3 kB)
Requirement already satisfied: distro<2,>=1.7.0 in /usr/local/lib/python3.12/dist-packages (from openai>=1.1.0->llama-index-embeddings-ope
Requirement already satisfied: jiter<1,>=0.4.0 in /usr/local/lib/python3.12/dist-packages (from openai>=1.1.0->llama-index-embeddings-open
Requirement already satisfied: sniffio in /usr/local/lib/python3.12/dist-packages (from openai>=1.1.0->llama-index-embeddings-openai<0.6,>
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas<2.3.0->llama-index-readers-f
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas<2.3.0->llama-index-readers-file<0.6,>=
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas<2.3.0->llama-index-readers-file<0.6,
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.12/dist-packages (from pydantic>=2.8.0->llama-index-core<0
Requirement already satisfied: pydantic-core==2.33.2 in /usr/local/lib/python3.12/dist-packages (from pydantic>=2.8.0->llama-index-core<0.
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.12/dist-packages (from pydantic>=2.8.0->llama-index-core
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests>=2.31.0->llama-index-cor
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests>=2.31.0->llama-index-core<0.15
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy>=1.4.49->sqlalchemy[asyncio]>=1.4.4
Requirement already satisfied: mypy-extensions>=0.3.0 in /usr/local/lib/python3.12/dist-packages (from typing-inspect>=0.8.0->llama-index-
Requirement already satisfied: marshmallow<4.0.0,>=3.18.0 in /usr/local/lib/python3.12/dist-packages (from dataclasses-json->llama-index-c
INFO: pip is looking at multiple versions of llama-cloud-services to determine which version is compatible with other requirements. This c
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.87-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.87 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.87-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.86-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.86 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.86-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.85-py3-none-any.whl.metadata (6.6 kB)
INFO: pip is still looking at multiple versions of llama-cloud-services to determine which version is compatible with other requirements. T
Collecting llama-cloud-services>=0.6.85 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.85-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.84-py3-none-any.whl.metadata (6.6 kB)
INFO: This is taking longer than usual. You might need to provide the dependency resolver with stricter constraints to reduce runtime. See
Collecting llama-cloud-services>=0.6.84 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.84-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.83-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.82 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.83-py3-none-any.whl.metadata (3.3 kB)
  Downloading llama_cloud_services-0.6.82-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.82-py3-none-any.whl.metadata (6.6 kB)
  Downloading llama_parse-0.6.81-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.81 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.81-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.80-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.80 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.80-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.79-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.79 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.79-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.78-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.78 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.78-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.77-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.77 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.77-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.76-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.76 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.76-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.75-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.75 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.75-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.74-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.74 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.74-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.73-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.73 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.73-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.72-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.72 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.72-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.71-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.71 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.71-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.70-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.70 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_cloud_services-0.6.70-py3-none-any.whl.metadata (3.3 kB)
Collecting llama-parse>=0.5.0 (from llama-index-readers-llama-parse>=0.4.0->llama-index)
  Downloading llama_parse-0.6.69-py3-none-any.whl.metadata (6.6 kB)
Collecting llama-cloud-services>=0.6.69 (from llama-parse>=0.5.0->llama-index-readers-llama-parse>=0.4.0->llama-index)
```

6.3 Importando bibliotecas

```
from llama_index.llms.openrouter import OpenRouter
from llama_index.core.llms import ChatMessage
```

6.4 Configurando API Key do OpenRouter

```
import os
from google.colab import userdata

os.environ['OPENROUTER_API_KEY'] = userdata.get("Open-Router_API-KEY")
```

6.5 Configuração dos modelos

Descrição: Foram avaliados 5 modelos disponíveis no OpenRouter, permitindo comparação de estabilidade e coerência:

- mistralai/mixtral-8x7b-instruct
- openai/gpt-oss-120b
- openai/gpt-4o-mini
- meta-llama/llama-3.3-70b-instruct
- deepseek/deepseek-v3.2
- x-ai/grok-4.1-fast

Referência: https://openrouter.ai/docs

```
llm_mistral = OpenRouter(model="mistralai/mixtral-8x7b-instruct")
llm_gpt = OpenRouter(model="openai/gpt-oss-120b")
llm_gpt_mini = OpenRouter(model="openai/gpt-4o-mini")
llm_llama = OpenRouter(model="meta-llama/llama-3.3-70b-instruct")
llm_deepseek = OpenRouter(model="deepseek/deepseek-v3.2")
llm_grok = OpenRouter(model="x-ai/grok-4.1-fast")
```

6.6 Função de prompt

Descrição: O prompt foi cuidadosamente projetado para:

- Restringir o modelo ao texto pré-processado;
- Fornecer contexto estruturado mínimo (resultado da sentença, gênero do juiz e da parte);
- Evitar explicações, justificativas ou vieses externos.

A classificação é feita em **três categorias** exclusivas:

- **EMPÁTICA** – Linguagem humanizada e sensível ao contexto da parte;
- **TÉCNICA** – Linguagem neutra, objetiva e predominantemente jurídica;
- **RIGOROSA** – Linguagem dura, formal ou censória, com

```
import re
import time
from llama_index.core.llms import ChatMessage

def montar_prompt(resultado_sentenca, genero_juiz, genero_parte, decisao_processada) -> str:
    return f"""
Você é um especialista em análise de linguagem jurídica e argumentação judicial.

Sua tarefa é classificar o **TOM DA DECISÃO JUDICIAL** com base **EXCLUSIVAMENTE**:
- no texto da decisão (pré-processado)
- no contexto estruturado fornecido

# DIRETRIZES
- NÃO avalie correção jurídica, justiça da decisão ou resultado do processo.
- NÃO use conhecimentos externos.
- NÃO explique sua resposta.

————————————————————
# CONTEXTO ESTRUTURADO
————————————————————
Resultado da sentença: {resultado_sentenca}
Gênero do(a) juiz(a): {genero_juiz}
Gênero da parte: {genero_parte}

————————————————————
# TEXTO DA DECISÃO (pré-processado)
————————————————————
\"\"\"{decisao_processada}\"\"\"

————————————————————
# CRITÉRIOS DE CLASSIFICAÇÃO
```

```
————————————————————————————

Classifique o TOM predominante do texto em **APENAS UMA** das categorias abaixo:

• **EMPATICA**
  - Linguagem humanizada ou acolhedora
  - Demonstra preocupação com dignidade, sofrimento, vulnerabilidade ou situação pessoal da parte
  - Uso de expressões que reconhecem contexto social, humano ou existencial
  - Mesmo quando técnica, apresenta sensibilidade explícita

• **TECNICA**
  - Linguagem neutra e impessoal
  - Fundamentação jurídica objetiva
  - Predomínio de dispositivos legais, precedentes, conceitos jurídicos
  - Ausência de juízo emocional positivo ou negativo

• **RIGOROSA**
  - Linguagem dura, fria ou excessivamente formal
  - Tom repreensivo, censório ou de pouca concessão
  - Ênfase em limites legais sem contextualização humana
  - Eventual censura moral, reprovação ou desconsideração da situação da parte

————————————————————————————
# FORMATO OBRIGATÓRIO DE SAÍDA
————————————————————————————

Retorne **EXATAMENTE UMA PALAVRA**, em letras maiúsculas, sem pontuação, sem explicações adicionais:

EMPATICA
TECNICA
RIGOROSA
"""

def extrair_sentimento(texto):
    t = texto.strip().upper()
    # pega a primeira ocorrência de uma classe válida
    m = re.search(r"\b(EMPATICA|TECNICA|RIGOROSA)\b", t)
    return m.group(1)

def classificar_tom(row, llm, max_retries=3) -> str:
    prompt = montar_prompt(
        row["resultado_sentenca"],
        row["genero_juiz"],
        row["genero_parte"],
        row["decisao_processada"],
    )

    msg = ChatMessage(role="user", content=prompt)

    for _ in range(max_retries):
        try:
            resp = llm.chat([msg])
            return extrair_sentimento(str(resp))

        except Exception:
            time.sleep(1)

    return "ERRO"
```

6.7 Aplicação no dataframe a análise de cada um dos modelos

Mistral

```
# @title Mistral
df["sentimento_llm_mistral"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_mistral), axis = 1)
```

```
ERROR:root:Internal Python error in the inspect module.
Below is the traceback from this internal error.

Traceback (most recent call last):
  File "/usr/local/lib/python3.12/dist-packages/IPython/core/interactiveshell.py", line 3553, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "/tmp/ipython-input-4134012958.py", line 2, in <cell line: 0>
    df["sentimento_llm_mistral"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_mistral), axis = 1)
                                   ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/pandas/core/frame.py", line 10374, in apply
    return op.apply().__finalize__(self, method="apply")
           ^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/pandas/core/apply.py", line 916, in apply
    return self.apply_standard()
           ^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/pandas/core/apply.py", line 1063, in apply_standard
    results, res_index = self.apply_series_generator()
                         ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/pandas/core/apply.py", line 1081, in apply_series_generator
    results[i] = self.func(v, *self.args, **self.kwargs)
                 ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/tmp/ipython-input-4134012958.py", line 2, in <lambda>
    df["sentimento_llm_mistral"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_mistral), axis = 1)
                                                              ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/tmp/ipython-input-2057410643.py", line 83, in classificar_tom
    resp = llm.chat([msg])
           ^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/llama_index_instrumentation/dispatcher.py", line 335, in wrapper
    result = func(*args, **kwargs)
             ^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/llama_index/llms/openai_like/base.py", line 173, in chat
    return super().chat(messages, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/llama_index_instrumentation/dispatcher.py", line 335, in wrapper
    result = func(*args, **kwargs)
             ^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/llama_index/core/llms/callbacks.py", line 175, in wrapped_llm_chat
    f_return_val = f(_self, messages, **kwargs)
                   ^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/llama_index/llms/openai/base.py", line 398, in chat
    return chat_fn(messages, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/llama_index/llms/openai/base.py", line 113, in wrapper
    return retry(f)(self, *args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/tenacity/__init__.py", line 336, in wrapped_f
    return copy(f, *args, **kw)
           ^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/tenacity/__init__.py", line 475, in __call__
    do = self.iter(retry_state=retry_state)
         ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/tenacity/__init__.py", line 376, in iter
    result = action(retry_state)
             ^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/tenacity/__init__.py", line 398, in <lambda>
    self._add_action_func(lambda rs: rs.outcome.result())
                                     ^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.12/concurrent/futures/_base.py", line 449, in result
    return self.__get_result()
           ^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.12/concurrent/futures/_base.py", line 401, in __get_result
    raise self._exception
  File "/usr/local/lib/python3.12/dist-packages/tenacity/__init__.py", line 478, in __call__
    result = fn(*args, **kwargs)
             ^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/llama_index/llms/openai/base.py", line 494, in _chat
    response = client.chat.completions.create(
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/openai/_utils/_utils.py", line 286, in wrapper
    return func(*args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/openai/resources/chat/completions/completions.py", line 1147, in create
    return self._post(
           ^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/openai/_base_client.py", line 1259, in post
    return cast(ResponseT, self.request(cast_to, opts, stream=stream, stream_cls=stream_cls))
                           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/openai/_base_client.py", line 982, in request
    response = self._client.send(
               ^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/httpx/_client.py", line 914, in send
    response = self._send_handling_auth(
               ^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/httpx/_client.py", line 942, in _send_handling_auth
    response = self._send_handling_redirects(
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/httpx/_client.py", line 979, in _send_handling_redirects
    response = self._send_single_request(request)
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/httpx/_client.py", line 1014, in _send_single_request
    response = transport.handle_request(request)
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.12/dist-packages/httpx/_transports/default.py", line 250, in handle_request
```

Próximas etapas: Explicar o erro

GPT

```
# @title GPT
df["sentimento_llm-gpt"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_gpt), axis = 1)
```

```
raise exc from None
```

GPT Mini

```
# @title GPT Mini
df["sentimento_llm-gpt-mini"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_gpt_mini), axis = 1)
```

Llama

```
# @title Llama
df["sentimento_llm-llama"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_llama), axis = 1)
```

DeepSeek

```
# @title DeepSeek
df["sentimento_llm-deep-seek"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_llama), axis = 1)
```

Grok

```
# @title Grok
df["sentimento_llm-grok"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_grok), axis = 1)
```

6.8 Geração dos resultados encontrados

Lendo o csv com as decisões já processadas

```
# @title Lendo o csv com as decisões já processadas
import pandas as pd
df = pd.read_csv("decisoes_classificadas_cs_llm.csv", sep=";")
```

```
display(df['sentimento_llm_mistral'].value_counts())
display(df['sentimento_llm-gpt'].value_counts())
display(df['sentimento_llm-gpt-mini'].value_counts())
display(df['sentimento_llm-llama'].value_counts())
display(df['sentimento_llm-deep-seek'].value_counts())
display(df['sentimento_llm-grok'].value_counts())

display(df.head(4))
```

```
KeyboardInterrupt
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
    [... skipping hidden 1 frame]

/tmp/ipython-input-4134012958.py in <cell line: 0>()
      1 # @title Mistral
----> 2 df["sentimento_llm_mistral"] = df.apply(lambda sentiment: classificar_tom(sentiment, llm_mistral), axis = 1)

45 frames

KeyboardInterrupt:

During handling of the above exception, another exception occurred:

AttributeError                            Traceback (most recent call last)
AttributeError: 'KeyboardInterrupt' object has no attribute '_render_traceback_'

During handling of the above exception, another exception occurred:

TypeError                                 Traceback (most recent call last)
    [... skipping hidden 1 frame]

/usr/local/lib/python3.12/dist-packages/IPython/core/ultratb.py in find_recursion(etype, value, records)
    380     # first frame (from in to out) that looks different.
    381     if not is_recursion_error(etype, value, records):
--> 382         return len(records), 0
    383
    384     # Select filename, lineno, func_name to track frames with

TypeError: object of type 'NoneType' has no len()
```

sentimento_llm_mistral	count
EMPATICA	663
TECNICA	89

dtype: int64

sentimento_llm-gpt	count
TECNICA	722
EMPATICA	22
ERRO	7
RIGOROSA	1

dtype: int64

sentimento_llm-gpt-mini	count
EMPATICA	382
TECNICA	224
RIGOROSA	146

dtype: int64

sentimento_llm-llama	count
TECNICA	708
EMPATICA	44

dtype: int64

6.9 Salvando resultados das decisoes classificadas

```
# Salvando em csv:

df.to_csv("decisoes_classificadas_cs_llm.csv", index=False, sep=";")
```

	EMPATICA	41

dtype: int64

6.10 Analisando custos por modelo

Descrição Análise de preço, tempo de uso para comparação entre modelos.

```
import pandas as pd

df_open_router = pd.read_csv("open_router.csv")
```

```
df_open_router.columns
```

```
Index(['generation_id', 'created_at', 'cost_total', 'cost_web_search',
       'cost_cache', 'cost_file_processing', 'byok_usage_inference',
       'tokens_prompt', 'tokens_completion', 'tokens_reasoning',
       'model_permaslug', 'provider_name', 'variant', 'cancelled', 'streamed',
       'user', 'finish_reason_raw', 'finish_reason_normalized',
       'generation_time_ms', 'time_to_first_token_ms', 'app_name',
       'api_key_name'],
      dtype='object')
```

dtype: int64

	arquivo	numero_processo	decisao_completa	genero_juiz	genero_parte	resultado_sentenca	decisao_processada	sentimen
0	00004983720188060127.pdf	0000498-37.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Feminino	Feminino	Procedente	PRAÇA LUIZ ALVES MESQUITA CENTRO CEP 63780 000...	

```
import pandas as pd

df_open_router = pd.read_csv("open_router.csv")

# tempo total por linha (ms)
df_open_router["tempo_ms"] = (
    df_open_router["generation_time_ms"].fillna(0)
    + df_open_router["time_to_first_token_ms"].fillna(0)
)

resumo_modelo = (
    df_open_router
    .groupby("model_permaslug", as_index=False)
    .agg(
        chamadas=("generation_id", "count"),
        preco_total=("cost_total", "sum"),
        tempo_total_ms=("tempo_ms", "sum"),
        tokens_prompt=("tokens_prompt", "sum"),
        tokens_completion=("tokens_completion", "sum"),
        tokens_reasoning=("tokens_reasoning", "sum"),
    )
)
```

```
PARAMETROS = {
    "mistralai/mixtral-8x7b-instruct": 32768,
    "meta-llama/llama-3.3-70b-instruct": 131072,
    "x-ai/grok-4.1-fast": 2000000,
    "openai/gpt-4o-mini": 128000,
    "openai/gpt-oss-120b": 131072,
}

# métricas derivadas
resumo_modelo["tempo_total_s"] = resumo_modelo["tempo_total_ms"] / 1000
resumo_modelo["preco_medio_por_chamada"] = resumo_modelo["preco_total"] / resumo_modelo["chamadas"]
resumo_modelo["tempo_medio_ms_por_chamada"] = resumo_modelo["tempo_total_ms"] / resumo_modelo["chamadas"]

resumo_modelo['parametros'] = resumo_modelo['model_permaslug'].map(PARAMETROS)

```

	model_permaslug	chamadas	preco_total	tempo_total_ms	tokens_prompt	tokens_completion	tokens_reasoning	tempo_total_s	preco_medio_p
1	mistralai/mixtral-8x7b-instruct	778	2.570629	960200	4507889	5670	0	960.200	
0	meta-llama/llama-3.3-70b-instruct	1504	1.743052	972154	7418807	6005	0	972.154	
4	x-ai/grok-4.1-fast	752	0.879856	13381442	3120306	562734	560473	13381.442	
2	openai/gpt-4o-mini	752	0.450083	548738	2993397	2402	0	548.738	
3	openai/gpt-oss-120b	811	0.238687	2365620	3243936	146497	151410	2365.620	

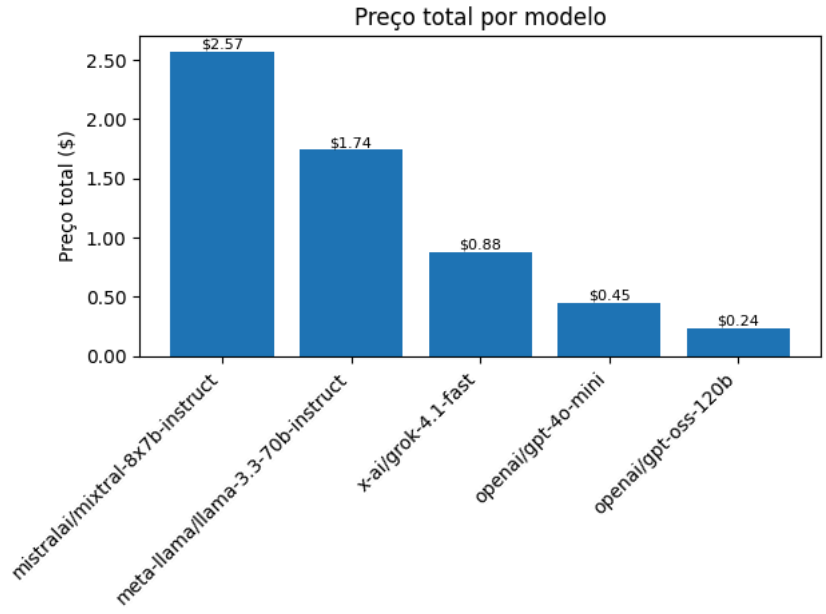
6.11 Salvando

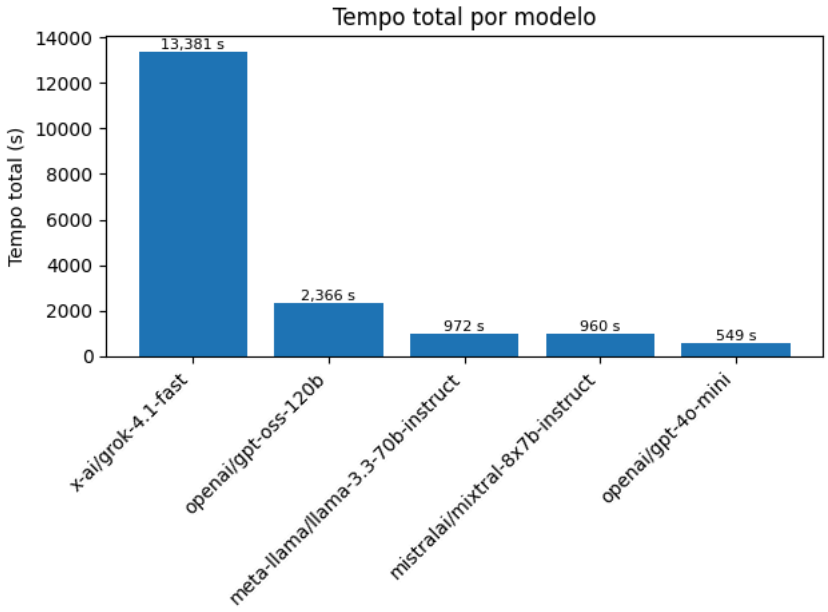
```
cols = [
    "model_permaslug",
    "parametros",
    "chamadas",
    "preco_total",
    "tempo_total_s",
    "preco_medio_por_chamada",
    "tempo_medio_ms_por_chamada",
    "tokens_prompt",
    "tokens_completion",
    "tokens_reasoning",
]

resumo_modelo[cols].to_csv("open_router_estatisticas.csv", index=False)
```

6.12 Gráficos comparativos entre dados do Open Router

```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

# 1) Ler o CSV
df = pd.read_csv("open_router_estatisticas.csv")

# 2) Garantir que as colunas numéricas estejam como número
colunas_numericas = ["preco_total", "tempo_total_s", "tempo_total_ms"]
for col in colunas_numericas:
    if col in df.columns:
        df[col] = pd.to_numeric(df[col], errors="coerce")

# 3) Criar a coluna tempo_total (prioriza segundos; se não existir, usa ms)
if "tempo_total_s" in df.columns and df["tempo_total_s"].notna().any():
    df["tempo_total"] = df["tempo_total_s"]
    unidade_tempo = "s"
elif "tempo_total_ms" in df.columns and df["tempo_total_ms"].notna().any():
    df["tempo_total"] = df["tempo_total_ms"]
    unidade_tempo = "ms"
else:
    raise ValueError("Não encontrei 'tempo_total_s' nem 'tempo_total_ms' no CSV para criar 'tempo_total'.")

# 4) Ordenar para facilitar a leitura dos gráficos
df_preco = df.sort_values("preco_total", ascending=False).reset_index(drop=True)
df_tempo = df.sort_values("tempo_total", ascending=False).reset_index(drop=True)

# 5) Formatação de moeda no eixo Y
formato_dolar = FuncFormatter(lambda x, pos: f"{x:,.2f}")

# -------------------------
# GRÁFICO 1: Preço total por modelo
```

```python
# -------------------------
plt.figure()
barras = plt.bar(df_preco["model_permaslug"], df_preco["preco_total"])
plt.xticks(rotation=45, ha="right")
plt.ylabel("Preço total ($)")
plt.title("Preço total por modelo")
plt.gca().yaxis.set_major_formatter(formato_dolar)

# Rótulos em cima das barras
for barra, valor in zip(barras, df_preco["preco_total"]):
    plt.text(
        barra.get_x() + barra.get_width() / 2,
        barra.get_height(),
        f"${valor:,.2f}",
        ha="center",
        va="bottom",
        fontsize=8
    )

plt.tight_layout()
plt.show()

# -------------------------
# GRÁFICO 2: Tempo total por modelo
# -------------------------
plt.figure()
barras2 = plt.bar(df_tempo["model_permaslug"], df_tempo["tempo_total"])
plt.xticks(rotation=45, ha="right")
plt.ylabel(f"Tempo total ({unidade_tempo})")
plt.title("Tempo total por modelo")

# Rótulos em cima das barras
for barra, valor in zip(barras2, df_tempo["tempo_total"]):
    plt.text(
        barra.get_x() + barra.get_width() / 2,
        barra.get_height(),
        f"{valor:,.0f} {unidade_tempo}",
        ha="center",
        va="bottom",
        fontsize=8
    )

plt.tight_layout()
plt.show()

# -------------------------
# Tabela resumida (formatada)
# -------------------------
tabela = df_preco[["model_permaslug", "preco_total", "tempo_total"]].copy()
tabela["preco_total"] = tabela["preco_total"].map(lambda x: f"${x:,.2f}")
tabela["tempo_total"] = tabela["tempo_total"].map(lambda x: f"{x:,.2f} {unidade_tempo}")

```



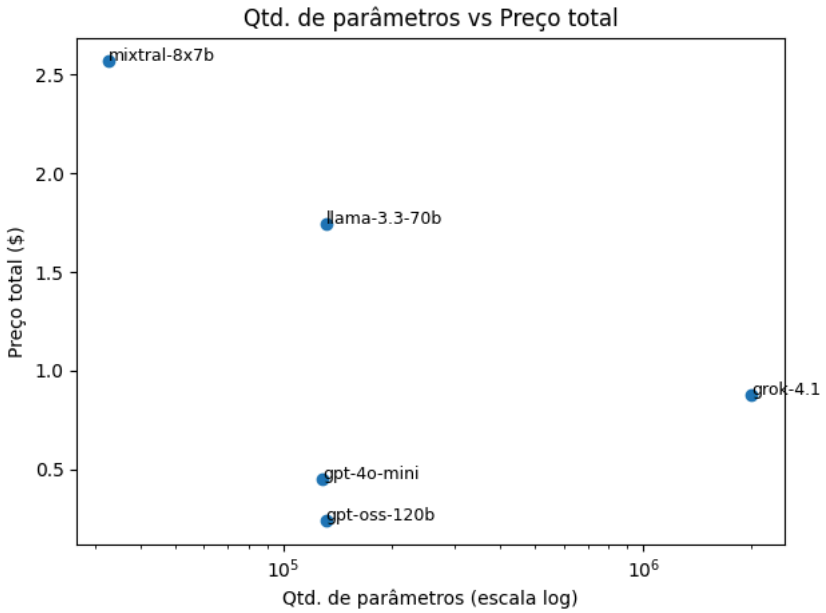


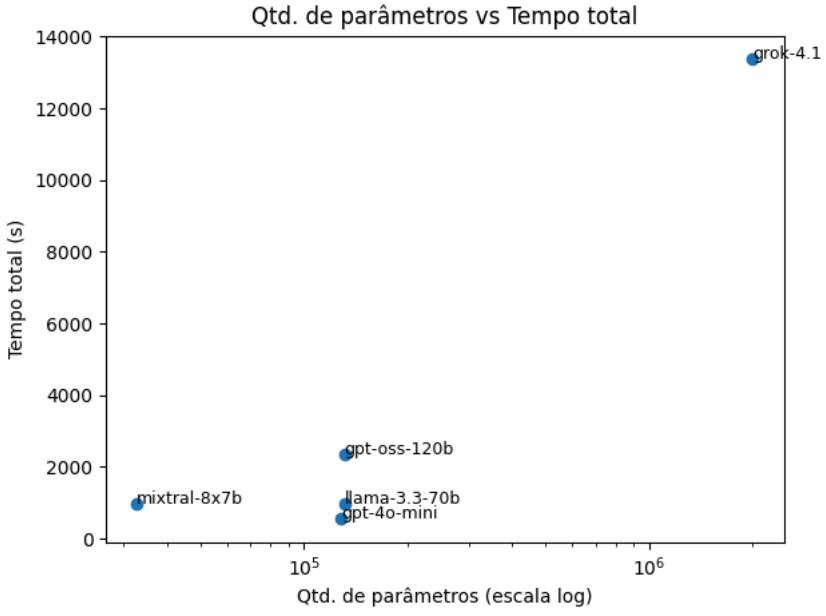
	model_permaslug	preco_total	tempo_total
0	mistralai/mixtral-8x7b-instruct	$2.57	960.20 s
1	meta-llama/llama-3.3-70b-instruct	$1.74	972.15 s
2	x-ai/grok-4.1-fast	$0.88	13,381.44 s
3	openai/gpt-4o-mini	$0.45	548.74 s
4	openai/gpt-oss-120b	$0.24	2,365.62 s

Próximas etapas: Gerar código com `tabela` | New interactive sheet

```python
import pandas as pd
import matplotlib.pyplot as plt

# df = pd.read_csv("open_router_estatisticas.csv")  # se ainda não tiver carregado

mapa_nomes = {
    "x-ai/grok-4.1-fast": "grok-4.1",
    "openai/gpt-4o-mini": "gpt-4o-mini",
    "openai/gpt-oss-120b": "gpt-oss-120b",
    "meta-llama/llama-3.3-70b-instruct": "llama-3.3-70b",
    "mistralai/mixtral-8x7b-instruct": "mixtral-8x7b",  # (se quiser manter)
}

# Garantir numéricos
for col in ["parametros", "preco_total", "tempo_total_s", "tempo_total_ms"]:
    if col in df.columns:
        df[col] = pd.to_numeric(df[col], errors="coerce")

df["tempo_total"] = df["tempo_total_s"]

# Criar nome curto (fallback: pega o que vem depois da barra)
df["modelo_curto"] = df["model_permaslug"].map(mapa_nomes)

base = df.dropna(subset=["parametros", "preco_total", "tempo_total"]).copy()

```

```
# -------------------------
# Gráfico: Parâmetros vs Preço total
# -------------------------
plt.figure()
plt.scatter(base["parametros"], base["preco_total"])

for _, r in base.iterrows():
    plt.annotate(r["modelo_curto"], (r["parametros"], r["preco_total"]), fontsize=9)

plt.xscale("log")
plt.xlabel("Qtd. de parâmetros (escala log)")
plt.ylabel("Preço total ($)")
plt.title("Qtd. de parâmetros vs Preço total")
plt.tight_layout()
plt.show()

# -------------------------
# Gráfico: Parâmetros vs Tempo total
# -------------------------
plt.figure()
plt.scatter(base["parametros"], base["tempo_total"])

for _, r in base.iterrows():
    plt.annotate(r["modelo_curto"], (r["parametros"], r["tempo_total"]), fontsize=9)

plt.xscale("log")
plt.xlabel("Qtd. de parâmetros (escala log)")
plt.ylabel(f"Tempo total ({unidade_tempo})")
plt.title("Qtd. de parâmetros vs Tempo total")
plt.tight_layout()
plt.show()
```





🔍 7. EXTRAÇÃO DO TRECHO DECISÓRIO VIA REGEX

7.1 Definição de Padrões de Decisão e Âncoras

Descrição: Padrões regex para identificar termos decisórios e âncoras textuais que marcam o início da parte dispositiva da sentença.

```python
import re

# 1) Decisões
padroes_decisao = re.compile(
    r"(?is)\b("
    r"julgo|"
    r"defiro|indefiro|"
    r"procedente|improcedente|"
    r"parcialmente\s+procedente|"
    r"extingo|"
    r"condeno|"
    r"homologo|"
    r"nego\s+provimento|dou\s+provimento|"
    r"rejeito|acolho"
    r")\b"
)

# 2) Âncoras para decisões
padroes_ancora = re.compile(
    r"(?im)\b("
    r"(?:do\s+)?dispositivo\b(?!s)|"
    r"ante\s+o\s+exposto|"
    r"isso\s+posto|isto\s+posto|"
    r"pelo\s+exposto|por\s+todo\s+o\s+exposto|"
    r"diante\s+disso|diante\s+do\s+exposto"
    r")\b"
)
```

7.2 Função de Extração do Trecho Decisório

Descrição: Extrai o trecho relevante da decisão judicial (dispositivo) usando âncoras textuais e padrões de decisão.

Estratégia:

1. Analisa apenas o final do documento (últimos 20.000 caracteres)
2. Prioridade: última âncora encontrada
3. Fallback: última semente de decisão
4. Fallback final: final do texto

Justificativa: O sentimento mais relevante está na parte decisória (dispositivo), não no relatório ou fundamentação.

```python
def extrair_trecho_decisao_simples(texto: str, n_chars: int = 4096, janela_back: int = 20000, lado: str = "inicio") -> str:
    """
    Retorna APENAS o trecho decisório.
    lado:
      - "inicio": pega os PRIMEIROS n_chars após âncora/decisão (ideal p/ 512)
      - "fim":    pega os ÚLTIMOS n_chars do trecho após âncora/decisão (ok p/ 4096)
    """
    if not isinstance(texto, str) or not texto.strip():
        return ""

    tail = texto[-janela_back:] if len(texto) > janela_back else texto
    offset = len(texto) - len(tail)

    start = None

    # 1) tenta âncora
    anc = list(padroes_ancora.finditer(tail))
    if anc:
        m = anc[-1]
        start = offset + m.end()
    else:
        # 2) tenta decisão
        dec = list(padroes_decisao.finditer(tail))
        if dec:
            m = dec[-1]
            start = offset + m.start()

    # 3) fallback
    if start is None:
        trecho = texto[-janela_back:]
    else:
        trecho = texto[start:]

    if lado == "fim":
        return trecho[-n_chars:].strip()
    else:  # "inicio"
        return trecho[:n_chars].strip()
```

7.3 Aplicação da Extração para Diferentes Tamanhos

Descrição: Extração de trechos decisórios em dois tamanhos diferentes:

- **512 caracteres**: Para modelos BERT padrão
- **4096 caracteres**: Para modelos Longformer

```
df = pd.read_csv("decisoes_classificadas_cs_llm.csv", sep=";")

df["trecho_decisao_4096"] = df["decisao_completa"].astype(str).apply(
    lambda t: extrair_trecho_decisao_simples(t, n_chars=4096)
)

df["trecho_decisao_512"] = df["decisao_completa"].astype(str).apply(
    lambda t: extrair_trecho_decisao_simples(t, n_chars=512)
)
```

7.4 Verificação da Qualidade das Extrações

Descrição: Validação de que os trechos extraídos contêm palavras decisórias.

```
df["trecho_tem_palavra_decisao_4096"] = df["trecho_decisao_4096"].apply(
    lambda s: bool(padroes_decisao.search(s)) if isinstance(s, str) else False
)

df["trecho_tem_palavra_decisao_512"] = df["trecho_decisao_512"].apply(
    lambda s: bool(padroes_decisao.search(s)) if isinstance(s, str) else False
)

# Porcentagens de qualidade
print(f"4096 - contém decisão: {(df['trecho_tem_palavra_decisao_4096'].mean() * 100):.2f} %")
print(f"512  - contém decisão: {(df['trecho_tem_palavra_decisao_512'].mean() * 100):.2f} %")
display(df.sample(5))
```

```
4096 - contém decisão: 94.68 %
512  - contém decisão: 91.76 %
```

	arquivo	numero_processo	decisao_completa	genero_juiz	genero_parte	resultado_sentenca	decisao_processada	sentim
283	02026247420238060071.pdf	0202624-74.2023.8.06.0071	2ª Vara Cível da Comarca de Crato Rua Álvaro P...	Masculino	Masculino	Procedente	Vara Cível Comarca Crato Rua Álvaro Peixoto Sã...	
461	02370725520248060001.pdf	0237072-55.2024.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Masculino	Feminino	Procedente	Vara Infância Juventude Rua Desembargador Flor...	
468	02399837420238060001.pdf	0239983-74.2023.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Feminino	Masculino	Procedente	Vara Infância Juventude Rua Desembargador Flor...	
33	00504278020218060047.pdf	0050427-80.2021.8.06.0047	2ª Vara Cível da Comarca de Baturité Av. Ouvid...	Masculino	Masculino	Procedente	Vara Cível Comarca Baturité Ouvidor Mor Vitori...	
289	02029337720248060001.pdf	0202933-77.2024.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Feminino	Masculino	Procedente	Vara Infância Juventude Rua Desembargador Flor...	

7.5 Removendo colunas intermediárias

```
df = df.drop(columns=["trecho_tem_palavra_decisao_4096", "trecho_tem_palavra_decisao_512"])
```

7.6 Verificando resultados

```
display(df.sample(5))
```

	arquivo	numero_processo	decisao_completa	genero_juiz	genero_parte	resultado_sentenca	decisao_processada	sentim
483	02420106420228060001.pdf	0242010-64.2022.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Feminino	Masculino	Procedente	Vara Infância Juventude Rua Desembargador Flor...	
396	02213987120238060001.pdf	0221398-71.2023.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Masculino	Masculino	Procedente	Vara Infância Juventude Rua Desembargador Flor...	
422	02267073920248060001.pdf	0226707-39.2024.8.06.0001	3ª Vara da Infância e Juventude Rua Desembarga...	Feminino	Feminino	Procedente	Vara Infância Juventude Rua Desembargador Flor...	
589	02572188820228060001.pdf	0257218-88.2022.8.06.0001	19ª Vara Cível (SEJUD 1º Grau) Rua Desembargad...	Feminino	Masculino	Procedente	19ª Vara Cível SEJUD Grau Rua Desembargador Fl...	

7.7 Salvando resultados

```
df.to_csv("decisoes_classificadas_cs_llm_regex.csv", index=False, sep=";")
```

8. ANÁLISE DE SENTIMENTO COM BERT e RoBERTa

8.1 Carregamento dos Modelos BERT e RoBERTa

Descrição: Modelo BERT fine-tuned para análise de sentimento (512 tokens) em avaliações de produtos em 6 idiomas (incluindo português). Prediz sentimento em escala de 1 a 5 estrelas. Modelo RoBERTa que contém maior janela de tokens para análise (4096 tokens).

Hugging Face 🤗:

- BERT: https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment
- RoBERTa: https://huggingface.co/markussagen/xlm-roberta-longformer-base-4096

```
!pip uninstall -y transformers tokenizers huggingface-hub safetensors
!pip install -U --no-cache-dir "transformers==4.56.2" "tokenizers==0.22.1" "huggingface-hub==0.35.3" "safetensors>=0.4.3"

from transformers import pipeline

bert_pipeline = pipeline(
    "sentiment-analysis",
    model="nlptown/bert-base-multilingual-uncased-sentiment",
    truncation=True,
    max_length=512
  )
```

```
Found existing installation: transformers 4.57.0
Uninstalling transformers-4.57.0:
  Successfully uninstalled transformers-4.57.0
Found existing installation: tokenizers 0.22.1
Uninstalling tokenizers-0.22.1:
  Successfully uninstalled tokenizers-0.22.1
Found existing installation: huggingface-hub 0.35.3
Uninstalling huggingface-hub-0.35.3:
  Successfully uninstalled huggingface-hub-0.35.3
Found existing installation: safetensors 0.6.2
Uninstalling safetensors-0.6.2:
  Successfully uninstalled safetensors-0.6.2
Collecting transformers==4.56.2
  Downloading transformers-4.56.2-py3-none-any.whl.metadata (40 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 40.1/40.1 kB 134.3 MB/s eta 0:00:00
Collecting tokenizers==0.22.1
  Downloading tokenizers-0.22.1-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (6.8 kB)
Collecting huggingface-hub==0.35.3
  Downloading huggingface_hub-0.35.3-py3-none-any.whl.metadata (14 kB)
Collecting safetensors>=0.4.3
  Downloading safetensors-0.7.0-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (4.1 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers==4.56.2) (3.20.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from transformers==4.56.2) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers==4.56.2) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers==4.56.2) (6.0.3)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers==4.56.2) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers==4.56.2) (2.32.4)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers==4.56.2) (4.67.1)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub==0.35.3) (2025.3.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub==0.35.3) (4.15
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub==0.35.3) (1.1.10)
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->transformers==4.56.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests->transformers==4.56.2) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->transformers==4.56.2) (2.5.0
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests->transformers==4.56.2) (2025.
Downloading transformers-4.56.2-py3-none-any.whl (11.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.6/11.6 MB 301.5 MB/s eta 0:00:00
Downloading tokenizers-0.22.1-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.3/3.3 MB 400.2 MB/s eta 0:00:00
Downloading huggingface_hub-0.35.3-py3-none-any.whl (564 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 564.3/564.3 kB 392.2 MB/s eta 0:00:00
Downloading safetensors-0.7.0-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (507 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 507.2/507.2 kB 379.7 MB/s eta 0:00:00
Installing collected packages: safetensors, huggingface-hub, tokenizers, transformers
Successfully installed huggingface-hub-0.35.3 safetensors-0.7.0 tokenizers-0.22.1 transformers-4.56.2
WARNING: The following packages were previously imported in this runtime:
  [huggingface_hub,safetensors,tokenizers,transformers]
You must restart the runtime in order to use newly installed versions.
```

RESTART SESSION

```
config.json: 100%                                   953/953 [00:00<00:00, 128kB/s]
model.safetensors: 100%                             669M/669M [00:04<00:00, 220MB/s]
tokenizer_config.json: 100%                         39.0/39.0 [00:00<00:00, 4.74kB/s]
vocab.txt:  872k/? [00:00<00:00, 43.9MB/s]
special_tokens_map.json: 100%                       112/112 [00:00<00:00, 15.4kB/s]
Device set to use cuda:0
```

```python
roberta_pipeline = pipeline(
    "sentiment-analysis",
    model="markussagen/xlm-roberta-longformer-base-4096",
    truncation=True,
    max_length=4096
)
```

```
config.json: 100%                                   773/773 [00:00<00:00, 104kB/s]
pytorch_model.bin: 100%                             1.12G/1.12G [00:07<00:00, 238MB/s]
model.safetensors: 100%                             1.12G/1.12G [00:06<00:00, 259MB/s]
Some weights of XLMRobertaForSequenceClassification were not initialized from the model checkpoint at markussagen/xlm-roberta-longformer-b
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
tokenizer_config.json: 100%                         212/212 [00:00<00:00, 28.4kB/s]
sentencepiece.bpe.model: 100%                       5.07M/5.07M [00:01<00:00, 4.04MB/s]
special_tokens_map.json: 100%                       150/150 [00:00<00:00, 13.0kB/s]
Device set to use cuda:0
```

8.2 Testando análise de sentimentos antes de prosseguir para o dataset completo

```python
textos_teste = [
    # Texto 1 - Empático / Procedente
    "Diante da vulnerabilidade da parte autora e da comprovada necessidade do medicamento, "
    " procedente o pedido, assegurando o direito fundamental à saúde.",
```

```
    # Texto 2 – Rigoroso / Improcedente
    "Não restou comprovado o direito alegado. Ausente prova mínima dos fatos constitutivos, "
    "julgo improcedente o pedido, nos termos da legislação vigente."
]

# BERT (512)
saida_bert = bert_pipeline(textos_teste)

# RoBERTa Longformer (4096)
saida_roberta = roberta_pipeline(textos_teste)

print("=== BERT ===")
for t, s in zip(textos_teste, saida_bert):
    print("\nTexto:", t)
    print("Saída:", s)

print("\n=== RoBERTa ===")
for t, s in zip(textos_teste, saida_roberta):
    print("\nTexto:", t)
    print("Saída:", s)
```

```
=== BERT ===

Texto: Diante da vulnerabilidade da parte autora e da comprovada necessidade do medicamento,  procedente o pedido, assegurando o direito f
Saída: {'label': '5 stars', 'score': 0.2671985328197479}

Texto: Não restou comprovado o direito alegado. Ausente prova mínima dos fatos constitutivos, julgo improcedente o pedido, nos termos da l
Saída: {'label': '1 star', 'score': 0.5797569155693054}

=== RoBERTa ===

Texto: Diante da vulnerabilidade da parte autora e da comprovada necessidade do medicamento,  procedente o pedido, assegurando o direito f
Saída: {'label': 'LABEL_0', 'score': 0.6176493167877197}

Texto: Não restou comprovado o direito alegado. Ausente prova mínima dos fatos constitutivos, julgo improcedente o pedido, nos termos da l
Saída: {'label': 'LABEL_0', 'score': 0.6198321580886841}
```

8.3 Conclusão sobre testes realizados acima

Descrição: Os testes indicaram que, embora utilize rótulos genéricos de sentimento, o modelo BERT apresentou respostas coerentes com a polaridade dos textos analisados. Já o RoBERTa não distinguiu adequadamente textos semanticamente distintos, por conta disso, continuaremos somente com as análises do modelo BERT.

8.4 Funções Auxiliares para Processamento em Batch

Descrição: Função para processar múltiplos textos em batches, otimizando performance.

```
def _rodar_pipeline_em_batch(textos, pipe, batch_size=8):
    """
    Roda pipeline em batches e retorna lista de dicts [{'label':..., 'score':...}, ...]
    Mantém alinhamento com a lista de entrada.
    """
    textos = list(textos)
    out = []
    for i in range(0, len(textos), batch_size):
        batch = textos[i:i+batch_size]
        # garante string
        batch = ["" if (not isinstance(t, str)) else t for t in batch]
        preds = pipe(batch)
        out.extend(preds)
    return out
```

8.5 Funções de Transformação de Resultados

Descrição: Conversão dos labels do BERT (1-5 stars) para categorias interpretáveis.

Mapeamento:

- 1-2 stars → RIGOROSA (linguagem restritiva, negativa)
- 3 stars → TECNICA (linguagem neutra, formal)
- 4-5 stars → EMPATICA (linguagem favorável, acolhedora)

```
def bert_label_to_int(label: str):
    # '1 star'...'5 stars' -> 1..5
    import re
    m = re.search(r"(\d)", label)
    return int(m.group(1))

def map_1_5_to_sentimento(valor_1_5: float):
    """
```

```
    Mapeamento simples p/ ficar comparável com LLM:
    1-2 -> RIGOROSA
    3   -> TECNICA
    4-5 -> EMPATICA
    """
    v = int(valor_1_5)
    if v <= 2:
        return "RIGOROSA"
    elif v == 3:
        return "TECNICA"
    else:
        return "EMPATICA"
```

8.6 Aplicação do BERT no Dataset Completo

Descrição: Análise de sentimento em todos os trechos decisórios de 512 caracteres.

```
import pandas as pd
df = pd.read_csv("decisoes_classificadas_cs_llm_regex.csv", sep=";")

# --- BERT (512) ---
preds_bert = _rodar_pipeline_em_batch(
    df["trecho_decisao_512"].tolist(),
    bert_pipeline,
    batch_size=16
)

df["analise_bert_label"] = [p.get("label") for p in preds_bert]
df["analise_bert_1a5"] = df["analise_bert_label"].apply(bert_label_to_int)
df["analise_bert"] = df["analise_bert_1a5"].apply(map_1_5_to_sentimento)
```

8.7 Verificando resultados

```
display(df.head(5))
```

	arquivo	numero_processo	decisao_completa	genero_juiz	genero_parte	resultado_sentenca	decisao_processada	sentimen
0	00004983720188060127.pdf	0000498-37.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Feminino	Feminino	Procedente	PRAÇA LUIZ ALVES MESQUITA CENTRO CEP 63780 000...	
1	00006188020188060127.pdf	0000618-80.2018.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Feminino	Masculino	Procedente	PRAÇA LUIZ ALVES MESQUITA CENTRO CEP 63780 000...	
2	00009657920198060127.pdf	0000965-79.2019.8.06.0127	PRAÇA LUIZ ALVES DE MESQUITA, S/N, CENTRO - CE...	Feminino	Masculino	Procedente	PRAÇA LUIZ ALVES MESQUITA CENTRO CEP 63780 000...	
3	00013144420188060151.pdf	0001314-44.2018.8.06.0151	1ª Vara da Comarca de Quixadá Av. Jesus, Maria...	Masculino	Masculino	Improcedente	Vara Comarca Quixadá Jesus Maria José Jardim M...	
4	00018334020188060047.pdf	0001833-40.2018.8.06.0047	1ª Vara da Comarca de Baturité Av. Ouvidor Mor...	Masculino	Masculino	Procedente	Vara Comarca Baturité Ouvidor Mor Vitoriano So...	

8.8 Gerando análise entre resultados

```
import pandas as pd
import numpy as np

SENT_VALIDOS = {"EMPATICA", "TECNICA", "RIGOROSA"}

MODELOS_SENTIMENTO = {
    "BERT": "analise_bert",
    "mistralai/mixtral-8x7b-instruct": "sentimento_llm_mistral",
    "openai/gpt-oss-120b": "sentimento_llm-gpt",
    "openai/gpt-4o-mini": "sentimento_llm-gpt-mini",
    "meta-llama/llama-3.3-70b-instruct": "sentimento_llm-llama",
    "deepseek/deepseek-v3.2": "sentimento_llm-deep-seek",
    "x-ai/grok-4.1-fast": "sentimento_llm-grok",
}

def tabela_sentimento_resultado_juiz_parte(df, col_modelo):
    """
    Tabela detalhada por:
      resultado × sentimento × genero_juiz × genero_parte
```

```
    Retorna:
      - valor_absoluto
      - percentual_total (sobre total válido do modelo)
      - percentual_dentro_resultado (soma 100% dentro de Procedente/Improcedente)
      - percentual_dentro_resultado_sentimento (soma 100% dentro de cada resultado+sentimento)
    """
    tmp = df.copy()

    # normaliza e filtra sentimento válido
    tmp["sentimento"] = tmp[col_modelo].astype(str).str.strip().str.upper()
    tmp = tmp[tmp["sentimento"].isin(SENT_VALIDOS)].copy()

    # resultado limpo
    tmp["resultado"] = tmp["resultado_sentenca"].astype(str).str.strip()

    # ---- Contagem base: resultado x sentimento x juiz x parte ----
    tab = (
        tmp
        .groupby(["resultado", "sentimento", "genero_juiz", "genero_parte"])
        .size()
        .rename("valor_absoluto")
        .reset_index()
    )

    # ---- Percentual do total válido do modelo ----
    total_validos = tab["valor_absoluto"].sum()
    tab["percentual_total"] = (tab["valor_absoluto"] / total_validos * 100) if total_validos else 0

    # ---- Percentual dentro do resultado (Procedente / Improcedente) ----
    denom_resultado = tab.groupby("resultado")["valor_absoluto"].transform("sum").replace(0, np.nan)
    tab["percentual_dentro_resultado"] = (tab["valor_absoluto"] / denom_resultado * 100).fillna(0)

    # ---- Percentual dentro de (resultado + sentimento) ----
    denom_rs = tab.groupby(["resultado", "sentimento"])["valor_absoluto"].transform("sum").replace(0, np.nan)
    tab["percentual_dentro_resultado_sentimento"] = (tab["valor_absoluto"] / denom_rs * 100).fillna(0)

    # ---- Ordenação amigável ----
    tab["resultado"] = pd.Categorical(tab["resultado"], categories=["Procedente", "Improcedente"], ordered=True)
    tab["sentimento"] = pd.Categorical(tab["sentimento"], categories=["RIGOROSA", "TECNICA", "EMPATICA"], ordered=True)
    tab["genero_juiz"] = pd.Categorical(tab["genero_juiz"], categories=["Masculino", "Feminino"], ordered=True)
    tab["genero_parte"] = pd.Categorical(tab["genero_parte"], categories=["Masculino", "Feminino"], ordered=True)

    tab = tab.sort_values(["resultado", "sentimento", "genero_juiz", "genero_parte"]).reset_index(drop=True)

    return tab

# --- Rodar para todos os modelos ---
resultados_detalhados = {}

for nome_modelo, col_modelo in MODELOS_SENTIMENTO.items():
    out = tabela_sentimento_resultado_juiz_parte(df, col_modelo)
    resultados_detalhados[nome_modelo] = out

    print("\n" + "="*100)
    print(f"Modelo: {nome_modelo} | Coluna: {col_modelo}")
    print("="*100)
    display(out)
```

```
================================================================================================
Modelo: BERT | Coluna: analise_bert
================================================================================================
```

	resultado	sentimento	genero_juiz	genero_parte	valor_absoluto	percentual_total	percentual_dentro_resultado	percentual_dentro_re
0	Procedente	RIGOROSA	Masculino	Masculino	124	16.489362	23.005566	
1	Procedente	RIGOROSA	Masculino	Feminino	145	19.281915	26.901670	
2	Procedente	RIGOROSA	Feminino	Masculino	121	16.090426	22.448980	
3	Procedente	RIGOROSA	Feminino	Feminino	101	13.430851	18.738404	
4	Procedente	TECNICA	Masculino	Masculino	2	0.265957	0.371058	
5	Procedente	TECNICA	Masculino	Feminino	5	0.664894	0.927644	
6	Procedente	TECNICA	Feminino	Feminino	1	0.132979	0.185529	
7	Procedente	EMPATICA	Masculino	Masculino	13	1.728723	2.411874	
8	Procedente	EMPATICA	Masculino	Feminino	20	2.659574	3.710575	
9	Procedente	EMPATICA	Feminino	Masculino	5	0.664894	0.927644	
10	Procedente	EMPATICA	Feminino	Feminino	2	0.265957	0.371058	
11	Improcedente	RIGOROSA	Masculino	Masculino	44	5.851064	20.657277	
12	Improcedente	RIGOROSA	Masculino	Feminino	54	7.180851	25.352113	
13	Improcedente	RIGOROSA	Feminino	Masculino	56	7.446809	26.291080	
14	Improcedente	RIGOROSA	Feminino	Feminino	47	6.250000	22.065728	
15	Improcedente	TECNICA	Masculino	Feminino	2	0.265957	0.938967	
16	Improcedente	TECNICA	Feminino	Feminino	3	0.398936	1.408451	
17	Improcedente	EMPATICA	Masculino	Masculino	2	0.265957	0.938967	
18	Improcedente	EMPATICA	Masculino	Feminino	2	0.265957	0.938967	
19	Improcedente	EMPATICA	Feminino	Masculino	1	0.132979	0.469484	
20	Improcedente	EMPATICA	Feminino	Feminino	2	0.265957	0.938967	

```
================================================================================================
Modelo: mistralai/mixtral-8x7b-instruct | Coluna: sentimento_llm_mistral
================================================================================================
```

	resultado	sentimento	genero_juiz	genero_parte	valor_absoluto	percentual_total	percentual_dentro_resultado	percentual_dentro_re
0	Procedente	TECNICA	Masculino	Masculino	19	2.526596	3.525046	
1	Procedente	TECNICA	Masculino	Feminino	13	1.728723	2.411874	
2	Procedente	TECNICA	Feminino	Masculino	4	0.531915	0.742115	
3	Procedente	TECNICA	Feminino	Feminino	1	0.132979	0.185529	
4	Procedente	EMPATICA	Masculino	Masculino	120	15.957447	22.263451	
5	Procedente	EMPATICA	Masculino	Feminino	157	20.877660	29.128015	
6	Procedente	EMPATICA	Feminino	Masculino	122	16.223404	22.634508	
7	Procedente	EMPATICA	Feminino	Feminino	103	13.696809	19.109462	
8	Improcedente	TECNICA	Masculino	Masculino	16	2.127660	7.511737	
9	Improcedente	TECNICA	Masculino	Feminino	22	2.925532	10.328638	
10	Improcedente	TECNICA	Feminino	Masculino	9	1.196809	4.225352	
11	Improcedente	TECNICA	Feminino	Feminino	5	0.664894	2.347418	
12	Improcedente	EMPATICA	Masculino	Masculino	30	3.989362	14.084507	
13	Improcedente	EMPATICA	Masculino	Feminino	36	4.787234	16.901408	
14	Improcedente	EMPATICA	Feminino	Masculino	48	6.382979	22.535211	
15	Improcedente	EMPATICA	Feminino	Feminino	47	6.250000	22.065728	

```
================================================================================================
Modelo: openai/gpt-oss-120b | Coluna: sentimento_llm_gpt
================================================================================================
```

	resultado	sentimento	genero_juiz	genero_parte	valor_absoluto	percentual_total	percentual_dentro_resultado	percentual_dentro_re
0	Procedente	TECNICA	Masculino	Masculino	134	17.986577	25.140713	
1	Procedente	TECNICA	Masculino	Feminino	161	21.610738	30.206379	
2	Procedente	TECNICA	Feminino	Masculino	121	16.241611	22.701689	
3	Procedente	TECNICA	Feminino	Feminino	98	13.154362	18.386492	
4	Procedente	EMPATICA	Masculino	Masculino	5	0.671141	0.938086	
5	Procedente	EMPATICA	Masculino	Feminino	6	0.805369	1.125704	
6	Procedente	EMPATICA	Feminino	Masculino	2	0.268456	0.375235	

7	Procedente	EMPATICA	Feminino	Feminino	6	0.805369	1.125704
8	Improcedente	RIGOROSA	Feminino	Masculino	1	0.134228	0.471698
9	Improcedente	TECNICA	Masculino	Masculino	45	6.040268	21.226415
10	Improcedente	TECNICA	Masculino	Feminino	57	7.651007	26.886792
11	Improcedente	TECNICA	Feminino	Masculino	55	7.382550	25.943396
12	Improcedente	TECNICA	Feminino	Feminino	51	6.845638	24.056604
13	Improcedente	EMPATICA	Masculino	Masculino	1	0.134228	0.471698
14	Improcedente	EMPATICA	Masculino	Feminino	1	0.134228	0.471698
15	Improcedente	EMPATICA	Feminino	Feminino	1	0.134228	0.471698

```
================================================================================================
Modelo: openai/gpt-4o-mini | Coluna: sentimento_llm-gpt-mini
================================================================================================
```

	resultado	sentimento	genero_juiz	genero_parte	valor_absoluto	percentual_total	percentual_dentro_resultado	percentual_dentro_re
0	Procedente	RIGOROSA	Masculino	Masculino	9	1.196809	1.669759	
1	Procedente	RIGOROSA	Masculino	Feminino	10	1.329787	1.855288	
2	Procedente	RIGOROSA	Feminino	Masculino	17	2.260638	3.153989	
3	Procedente	RIGOROSA	Feminino	Feminino	13	1.728723	2.411874	
4	Procedente	TECNICA	Masculino	Masculino	50	6.648936	9.276438	
5	Procedente	TECNICA	Masculino	Feminino	67	8.909574	12.430427	
6	Procedente	TECNICA	Feminino	Masculino	9	1.196809	1.669759	
7	Procedente	TECNICA	Feminino	Feminino	6	0.797872	1.113173	
8	Procedente	EMPATICA	Masculino	Masculino	80	10.638298	14.842301	
9	Procedente	EMPATICA	Masculino	Feminino	93	12.367021	17.254174	
10	Procedente	EMPATICA	Feminino	Masculino	100	13.297872	18.552876	
11	Procedente	EMPATICA	Feminino	Feminino	85	11.303191	15.769944	
12	Improcedente	RIGOROSA	Masculino	Masculino	20	2.659574	9.389671	
13	Improcedente	RIGOROSA	Masculino	Feminino	30	3.989362	14.084507	
14	Improcedente	RIGOROSA	Feminino	Masculino	28	3.723404	13.145540	
15	Improcedente	RIGOROSA	Feminino	Feminino	19	2.526596	8.920188	
16	Improcedente	TECNICA	Masculino	Masculino	25	3.324468	11.737089	
17	Improcedente	TECNICA	Masculino	Feminino	25	3.324468	11.737089	
18	Improcedente	TECNICA	Feminino	Masculino	17	2.260638	7.981221	
19	Improcedente	TECNICA	Feminino	Feminino	25	3.324468	11.737089	
20	Improcedente	EMPATICA	Masculino	Masculino	1	0.132979	0.469484	
21	Improcedente	EMPATICA	Masculino	Feminino	3	0.398936	1.408451	
22	Improcedente	EMPATICA	Feminino	Masculino	12	1.595745	5.633803	
23	Improcedente	EMPATICA	Feminino	Feminino	8	1.063830	3.755869	

Próximas etapas: Gerar código com out | New interactive sheet | Gerar código com out | New interactive sheet | Gerar código com out | New interactive sheet

```
================================================================================================
Modelo: meta-llama/llama-3.3-70b-instruct | Coluna: sentimento_llm-llama
================================================================================================
```

	resultado	sentimento	genero_juiz	genero_parte	valor_absoluto	percentual_total	percentual_dentro_resultado	percentual_dentro_re
0	Procedente	TECNICA	Masculino	Masculino	135	17.952128	25.046382	
1	Procedente	TECNICA	Masculino	Feminino	151	20.079787	28.014842	
2	Procedente	TECNICA	Feminino	Masculino	116	15.425532	21.521336	
3	Procedente	TECNICA	Feminino	Feminino	93	12.367021	17.254174	
4	Procedente	EMPATICA	Masculino	Masculino	4	0.531915	0.742115	
5	Procedente	EMPATICA	Masculino	Feminino	19	2.526596	3.525046	
6	Procedente	EMPATICA	Feminino	Masculino	10	1.329787	1.855288	
7	Procedente	EMPATICA	Feminino	Feminino	11	1.462766	2.040816	
8	Improcedente	TECNICA	Masculino	Masculino	46	6.117021	21.596244	
9	Improcedente	TECNICA	Masculino	Feminino	58	7.712766	27.230047	
10	Improcedente	TECNICA	Feminino	Masculino	57	7.579787	26.760563	
11	Improcedente	TECNICA	Feminino	Feminino	52	6.914894	24.413146	

```
================================================================================================
Modelo: deepseek/deepseek-v3.2 | Coluna: sentimento_llm-deep-seek
```

8.8 Baixando resultados parciais

```
df.to_csv("decisoes_classificadas_cs_llm_bert.csv", index=False, sep=";")
```

	resultado	sentimento	genero_juiz	genero_parte	valor_absoluto	percentual_total	percentual_dentro_resultado	percentual_dentro_re
0	Procedente	TECNICA	Masculino	Masculino	136	18.085106	25.231911	
2	Procedente	TECNICA	Feminino	Masculino	118	15.691489	21.892393	
3	Procedente	TECNICA	Feminino	Feminino	92	12.234043	17.068646	
4	Procedente	EMPATICA	Masculino	Masculino	3	0.398936	0.556586	
5	Procedente	EMPATICA	Masculino	Feminino	18	2.393617	3.339518	
6	Procedente	EMPATICA	Feminino	Masculino	8	1.063830	1.484230	
7	Procedente	EMPATICA	Feminino	Feminino	12	1.595745	2.226345	
6	Procedente	EMPATICA	Feminino	Masculino	89	11.835106	16.512059	
7	Procedente	EMPATICA	Feminino	Feminino	78	10.372340	14.471243	
8	Improcedente	RIGOROSA	Masculino	Masculino	4	0.531915	1.877934	

📊 9. VISUALIZAÇÕES E ANÁLISE EXPLORATÓRIA

9.1 Leitura do Dataset, Configuração de Cores e Ordenações

Descrição: Definição de cores semânticas e ordenações para visualizações consistentes.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

df = pd.read_csv("decisoes_classificadas_cs_llm_bert.csv", sep=";")

CORES = {
    "RIGOROSA": "#b22222",  # vermelho escuro
    "TECNICA":  "#808080",  # cinza
    "EMPATICA": "#2e8b57",  # verde
}

ORDEM_SENTIMENTO = ["RIGOROSA", "TECNICA", "EMPATICA"]

ORDEM_CATEGORIA = [
    "Juiz homem / Parte homem",
    "Juiz homem / Parte mulher",
    "Juiz mulher / Parte homem",
    "Juiz mulher / Parte mulher",
]

```

9.2 Funções de Normalização (Gênero e Sentimento)

```
def norm_genero(x):
    if pd.isna(x):
        return None
    s = str(x).strip().lower()
    if "masc" in s or s in {"m", "homem", "male"}:
        return "homem"
    if "fem" in s or s in {"f", "mulher", "female"}:
        return "mulher"
    return None

def norm_sentimento(x):
    if pd.isna(x):
        return None
    s = str(x).strip().upper()

    # mantém o padrão do projeto
    if s in {"RIGOROSA", "TECNICA", "EMPATICA"}:
        return s

    # casos ruins/erro (ex.: GPT pode ter "ERRO")
    if s in {"ERRO", "ERROR", "NONE", "NAN", ""}:
        return None

    return None
```

9.3 Geração de Gráficos (por resultado e por modelo)

```
def plot_8_barras_empilhadas(df, col_sentimento, nome_modelo=None, percentual=False,
                             mostrar_labels=True, min_label_abs=3, min_label_pct=5):
    """
    8 barras empilhadas por sentimento.
    Se mostrar_labels=True, escreve rótulos dentro de cada segmento.
      - absoluto: mostra se valor >= min_label_abs
      - percentual: mostra se valor >= min_label_pct (em %)
    """
    nome = nome_modelo or col_sentimento

    tmp = df.copy()
    tmp["resultado"] = tmp["resultado_sentenca"].astype(str).str.strip()
    tmp = tmp[tmp["resultado"].isin(["Procedente", "Improcedente"])].copy()

    tmp["sent"] = tmp[col_sentimento].apply(norm_sentimento)
    tmp = tmp[tmp["sent"].isin(ORDEM_SENTIMENTO)].copy()

    tmp["juiz"] = tmp["genero_juiz"].apply(norm_genero)
```

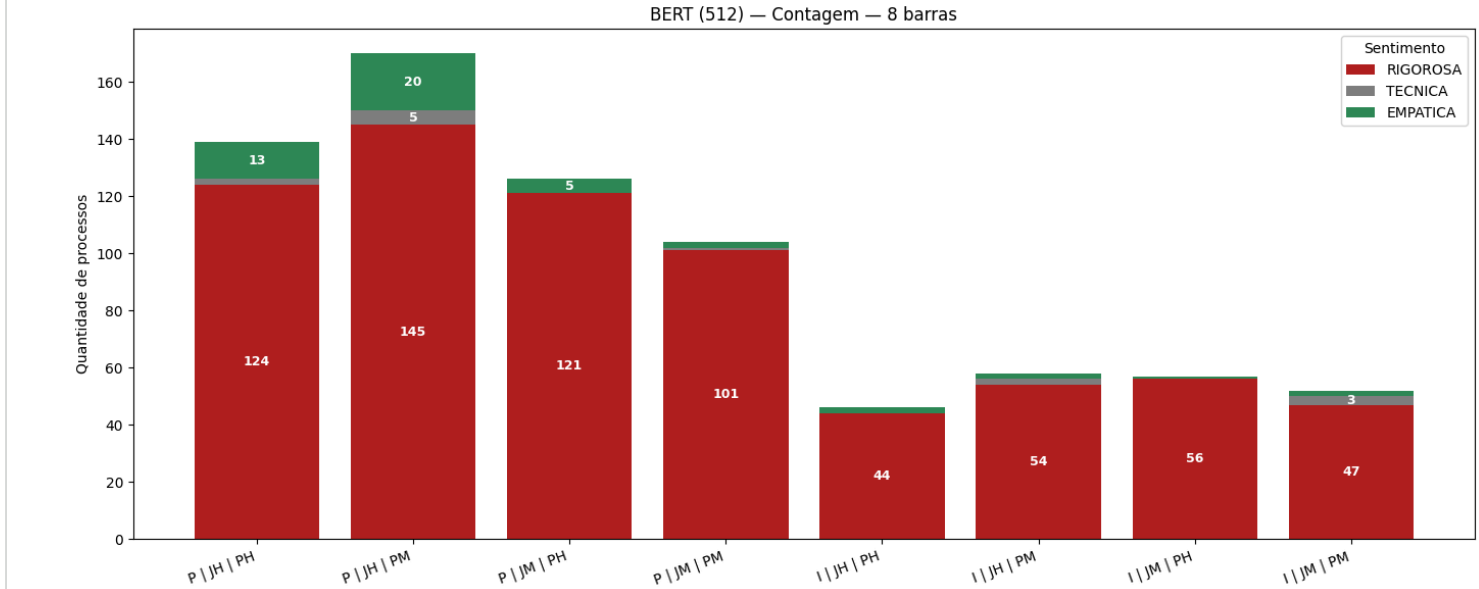
```
    tmp["parte"] = tmp["genero_parte"].apply(norm_genero)
    tmp = tmp[tmp["juiz"].isin(["homem", "mulher"]) & tmp["parte"].isin(["homem", "mulher"])].copy()

    MAP_RES = {"Procedente": "P", "Improcedente": "I"}
    MAP_J = {"homem": "JH", "mulher": "JM"}
    MAP_P = {"homem": "PH", "mulher": "PM"}

    tmp["label_x"] = (
        tmp["resultado"].map(MAP_RES) + " | " +
        tmp["juiz"].map(MAP_J) + " | " +
        tmp["parte"].map(MAP_P)
    )

    ordem_x = [
        "P | JH | PH", "P | JH | PM", "P | JM | PH", "P | JM | PM",
        "I | JH | PH", "I | JH | PM", "I | JM | PH", "I | JM | PM",
    ]

    tab = (
        tmp.pivot_table(
            index="label_x",
            columns="sent",
            values="numero_processo",
            aggfunc=pd.Series.nunique,
            fill_value=0
        )
        .reindex(ordem_x, fill_value=0)
    )

    for s in ORDEM_SENTIMENTO:
        if s not in tab.columns:
            tab[s] = 0
    tab = tab[ORDEM_SENTIMENTO]

    if percentual:
        tab_plot = tab.div(tab.sum(axis=1).replace(0, np.nan), axis=0) * 100
        tab_plot = tab_plot.fillna(0)
        ylabel = "Percentual de processos (%)"
        titulo = f"{nome} — Percentual (100% empilhado) — 8 barras"
    else:
        tab_plot = tab
        ylabel = "Quantidade de processos"
        titulo = f"{nome} — Contagem — 8 barras"

    x = np.arange(len(tab_plot.index))
    bottom = np.zeros(len(tab_plot.index))

    plt.figure(figsize=(14, 6))

    for s in ORDEM_SENTIMENTO:
        vals = tab_plot[s].values
        bars = plt.bar(x, vals, bottom=bottom, label=s, color=CORES[s])

        if mostrar_labels:
            for i, (bar, v) in enumerate(zip(bars, vals)):
                if percentual:
                    if v >= min_label_pct:
                        y_text = bottom[i] + v / 2
                        plt.text(
                            bar.get_x() + bar.get_width()/2, y_text, f"{v:.0f}%",
                            ha="center", va="center", fontsize=9,
                            color="white", fontweight="bold"
                        )
                else:
                    if v >= min_label_abs:
                        y_text = bottom[i] + v / 2
                        plt.text(
                            bar.get_x() + bar.get_width()/2, y_text, f"{int(v)}",
                            ha="center", va="center", fontsize=9,
                            color="white", fontweight="bold"
                        )

        bottom += vals

    plt.xticks(x, tab_plot.index, rotation=20, ha="right")
    plt.ylabel(ylabel)
    if percentual:
        plt.ylim(0, 100)
    plt.title(titulo)
    plt.legend(title="Sentimento")
    plt.tight_layout()
    plt.show()

    return tab
```

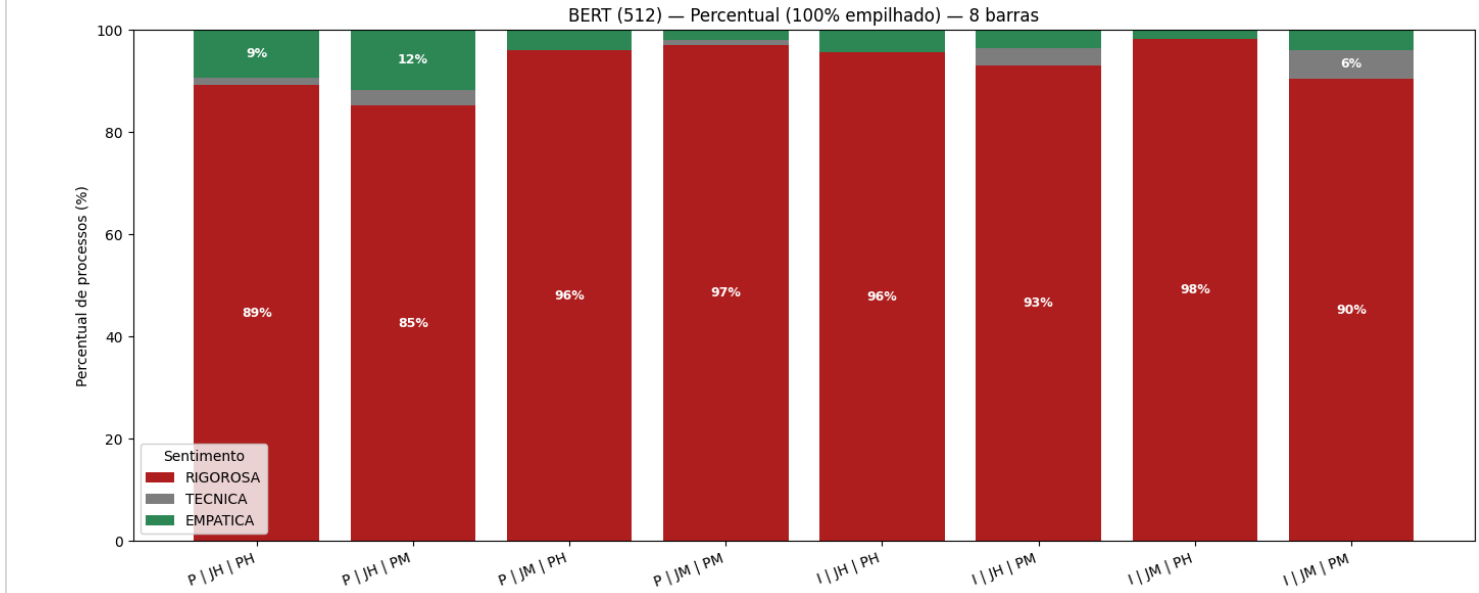
9.4 Criando dicionário chave:valor MODELOS

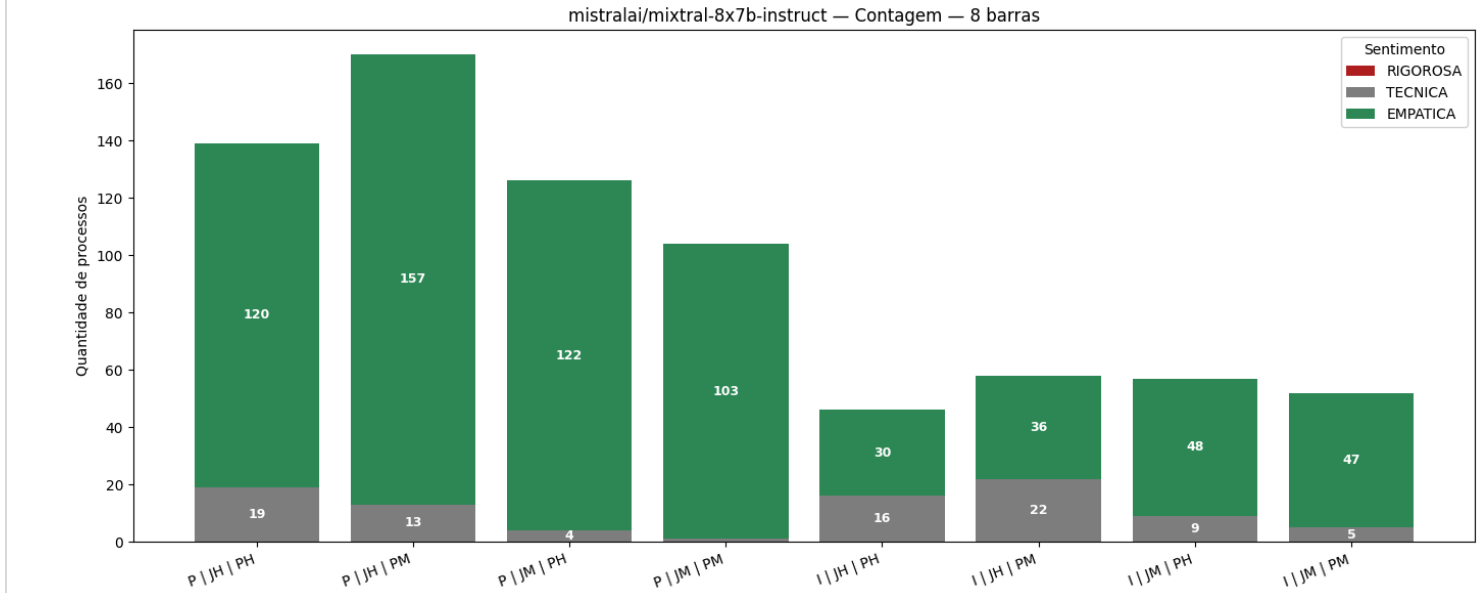
```
MODELOS = {
    "BERT (512)": "analise_bert",
    "mistralai/mixtral-8x7b-instruct": "sentimento_llm_mistral",
    "openai/gpt-oss-120b": "sentimento_llm-gpt",
    "openai/gpt-4o-mini": "sentimento_llm-gpt-mini",
    "meta-llama/llama-3.3-70b-instruct": "sentimento_llm-llama",
    "deepseek/deepseek-v3.2": "sentimento_llm-deep-seek",
    "x-ai/grok-4.1-fast": "sentimento_llm-grok",
}
```

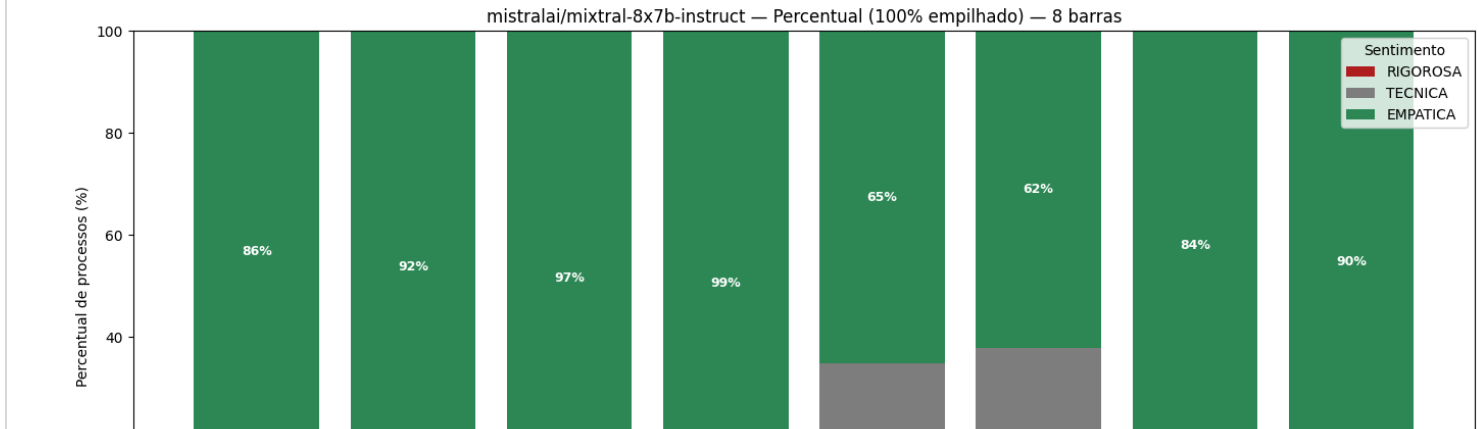
9.5 Gerando gráficos empilhados para cada um dos modelos

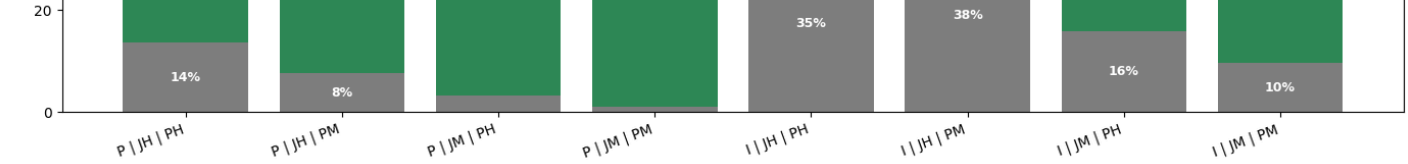
```
for nome_modelo, col in {
    "BERT (512)": "analise_bert",
    "mistralai/mixtral-8x7b-instruct": "sentimento_llm_mistral",
    "openai/gpt-oss-120b": "sentimento_llm-gpt",
    "openai/gpt-4o-mini": "sentimento_llm-gpt-mini",
    "meta-llama/llama-3.3-70b-instruct": "sentimento_llm-llama",
    "deepseek/deepseek-v3.2": "sentimento_llm-deep-seek",
    "x-ai/grok-4.1-fast": "sentimento_llm-grok",
}.items():
    plot_8_barras_empilhadas(df, col_sentimento=col, nome_modelo=nome_modelo, percentual=False)
    plot_8_barras_empilhadas(df, col_sentimento=col, nome_modelo=nome_modelo, percentual=True)
```


BERT (512) — Contagem — 8 barras
Quantidade de processos
Sentimento
RIGOROSA
TECNICA
EMPATICA
124
13
145
5
20
121
5
101
44
54
56
47
3
P | JH | PH
P | JH | PM
P | JM | PH
P | JM | PM
I | JH | PH
I | JH | PM
I | JM | PH
I | JM | PM

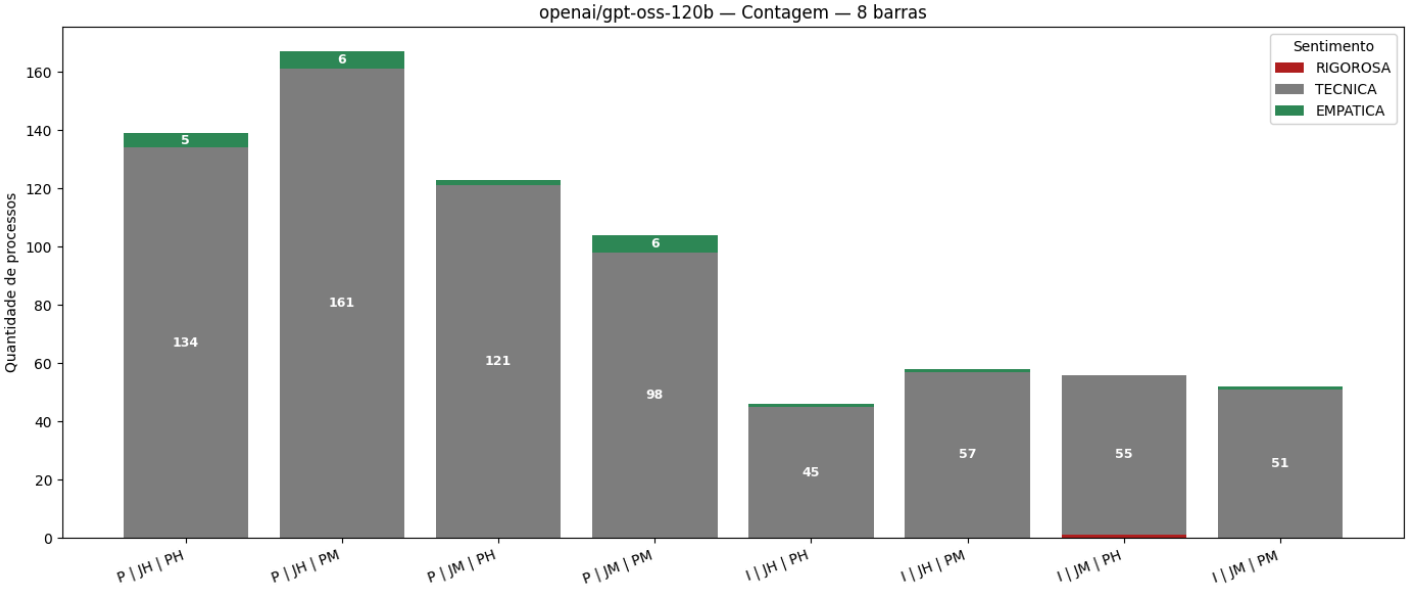

BERT (512) — Percentual (100% empilhado) — 8 barras
Percentual de processos (%)
Sentimento
RIGOROSA
TECNICA
EMPATICA
89%
9%
85%
12%
96%
97%
96%
93%
98%
90%
6%
P | JH | PH
P | JH | PM
P | JM | PH
P | JM | PM
I | JH | PH
I | JH | PM
I | JM | PH
I | JM | PM


mistralai/mixtral-8x7b-instruct — Contagem — 8 barras
Quantidade de processos
Sentimento
RIGOROSA
TECNICA
EMPATICA
120
19
157
13
122
4
103
30
16
36
22
48
9
47
5
P | JH | PH
P | JH | PM
P | JM | PH
P | JM | PM
I | JH | PH
I | JH | PM
I | JM | PH
I | JM | PM

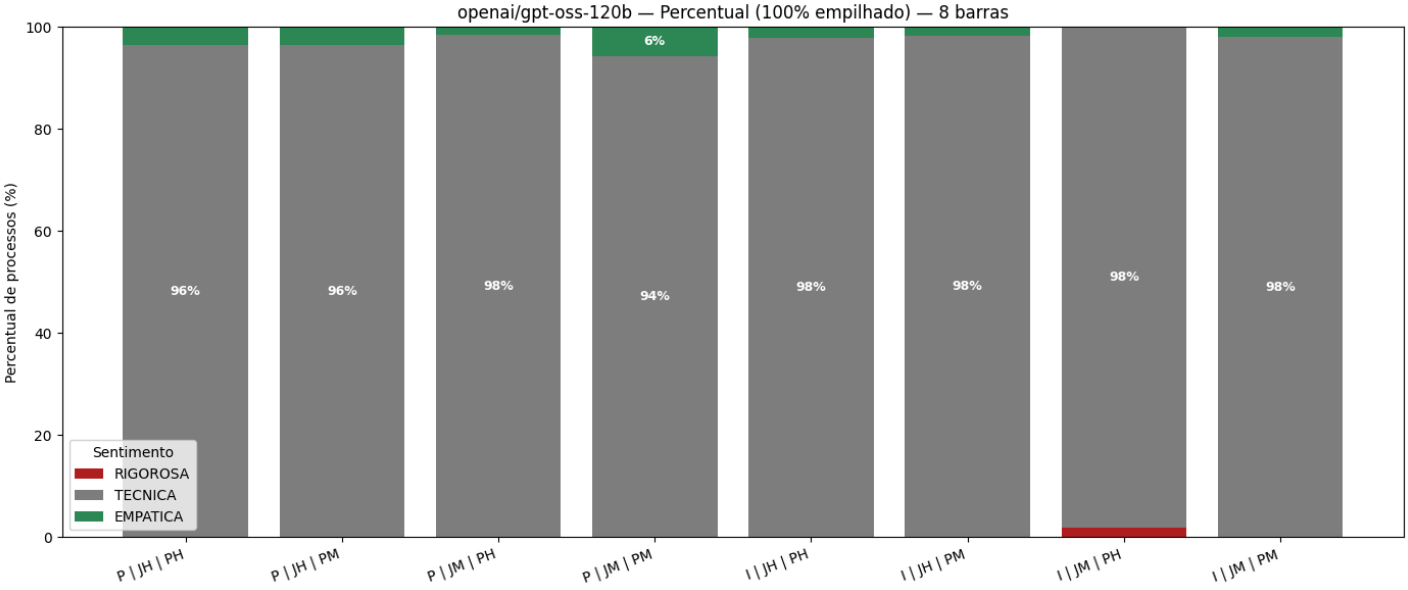

mistralai/mixtral-8x7b-instruct — Percentual (100% empilhado) — 8 barras
Percentual de processos (%)
Sentimento
RIGOROSA
TECNICA
EMPATICA
86%
92%
97%
99%
65%
62%
84%
90%



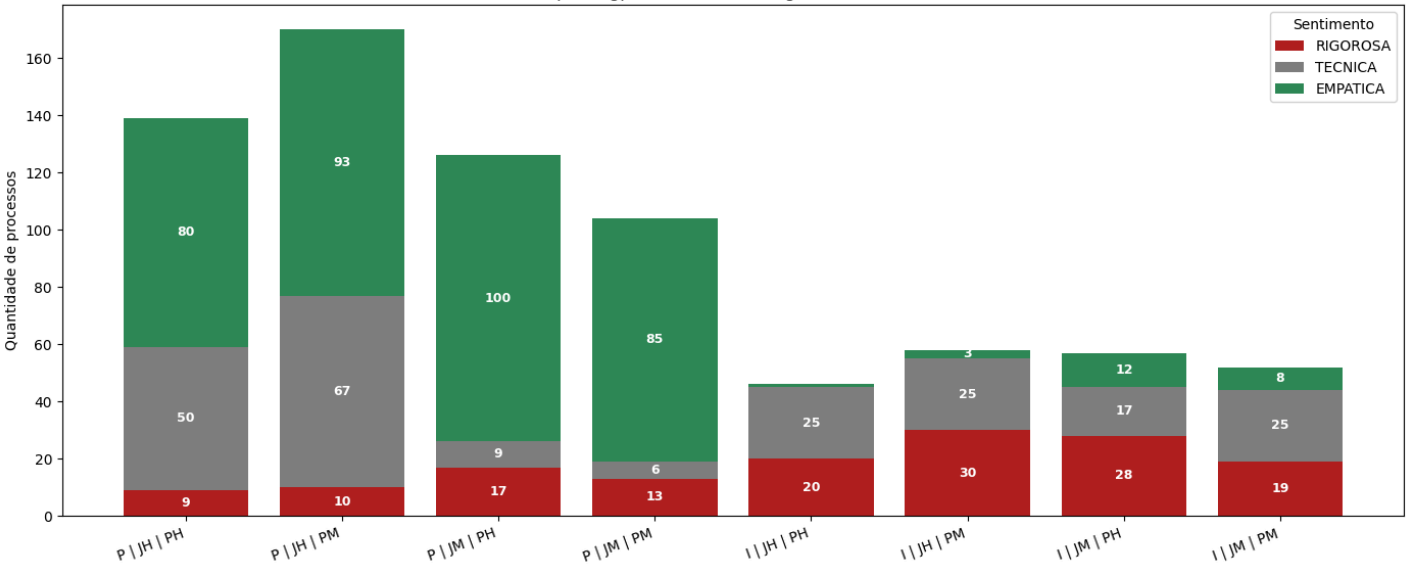
openai/gpt-oss-120b — Contagem — 8 barras



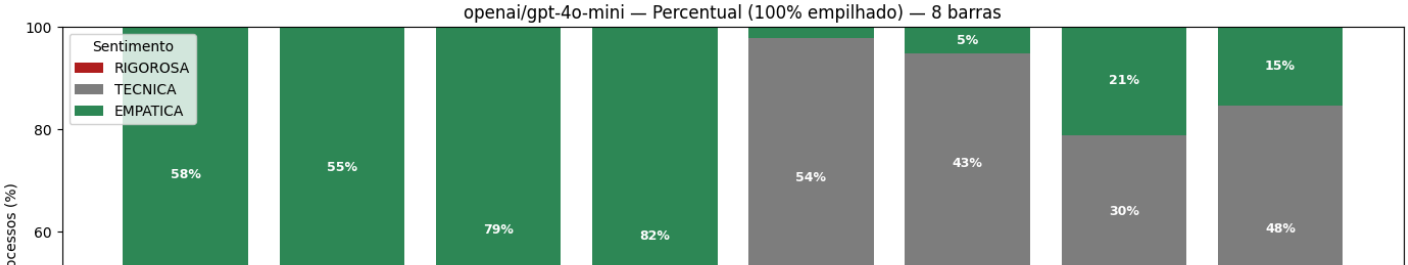
openai/gpt-oss-120b — Percentual (100% empilhado) — 8 barras

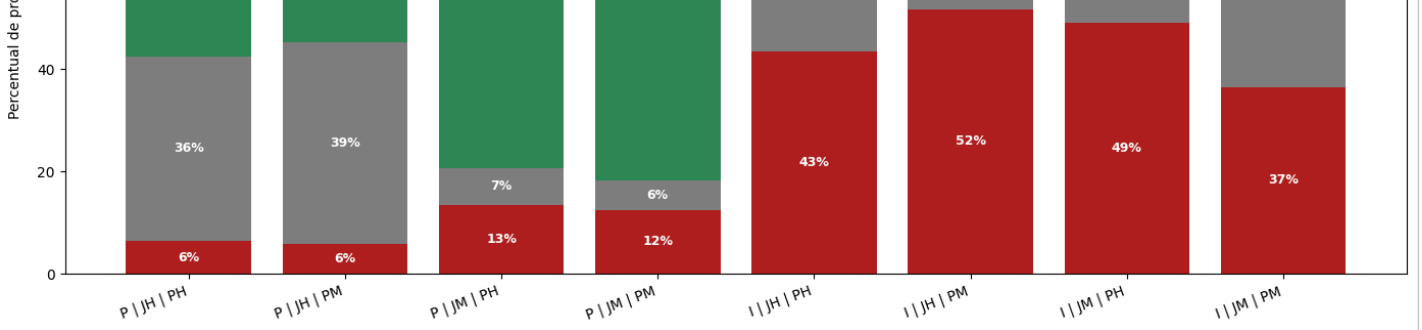


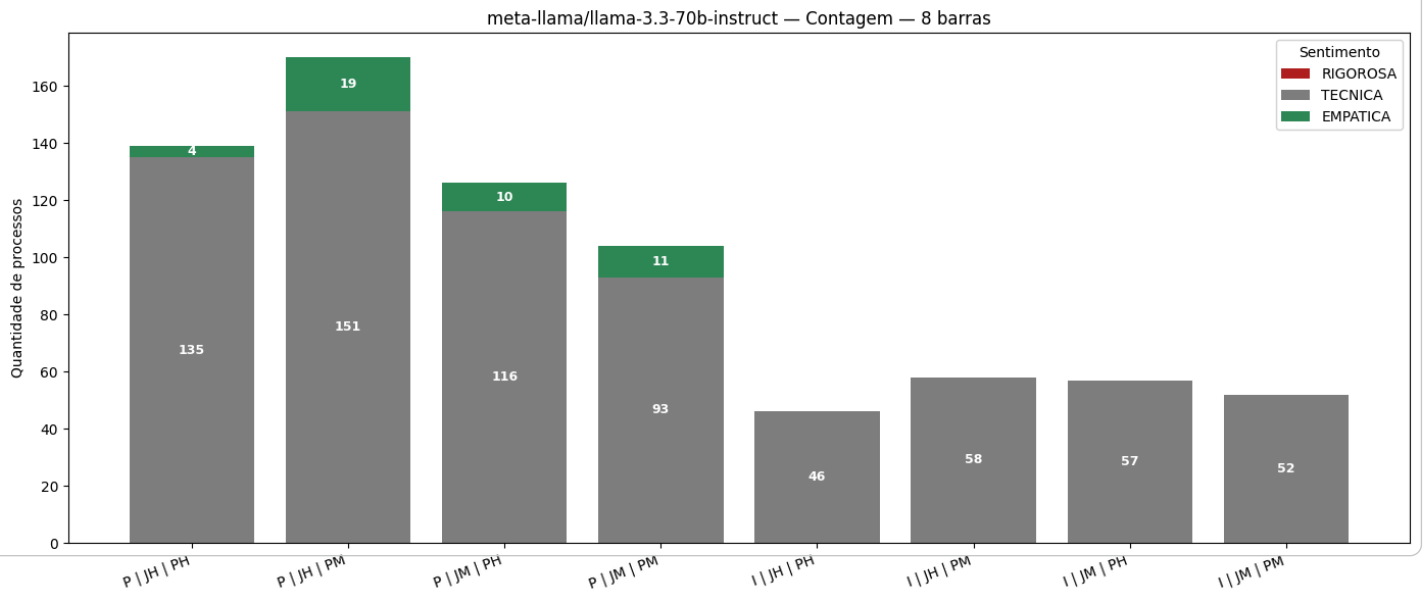
openai/gpt-4o-mini — Contagem — 8 barras

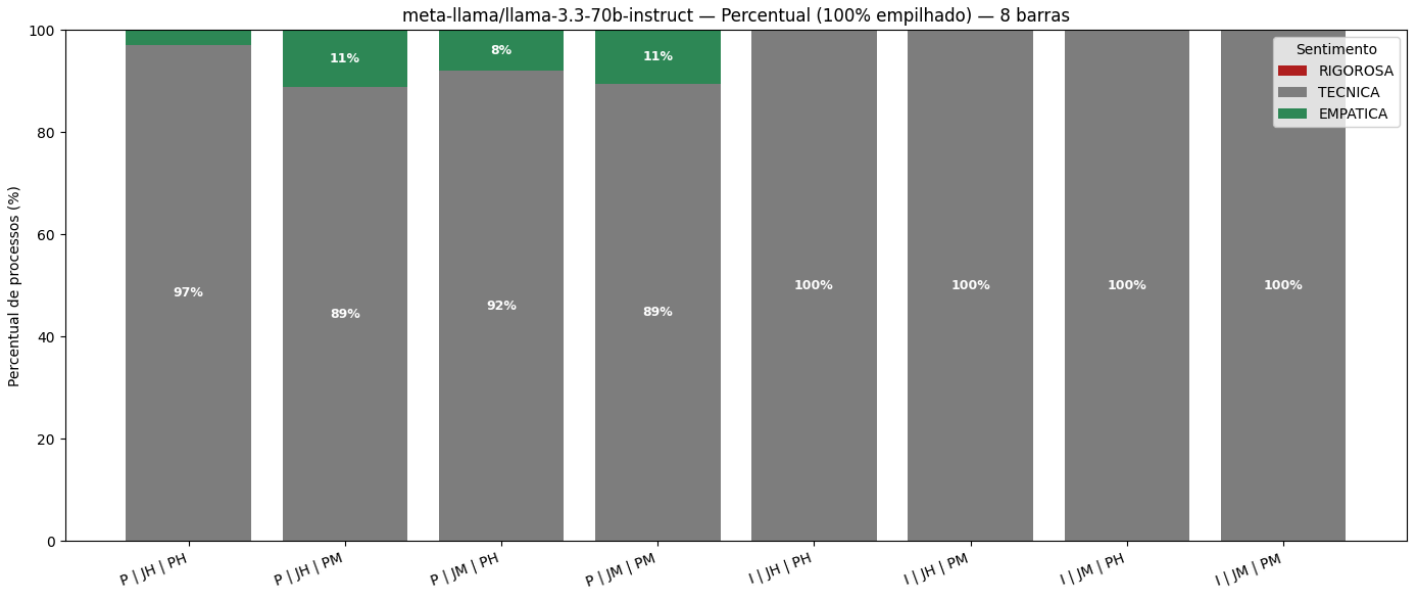


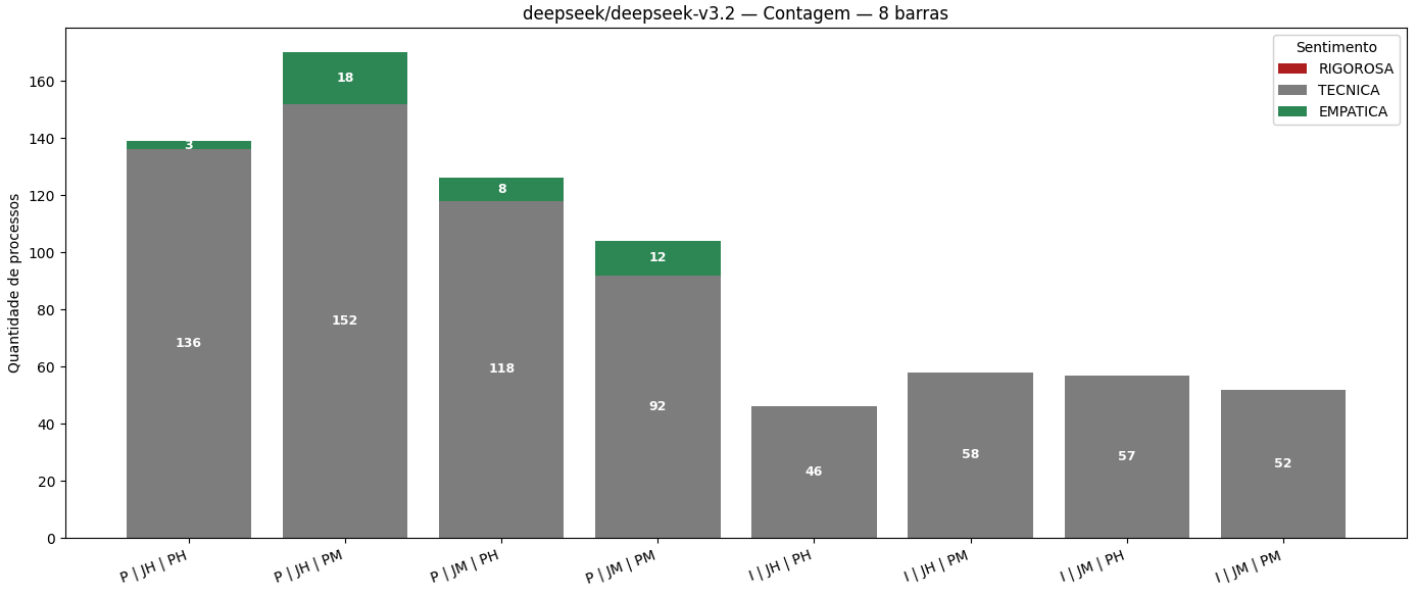
openai/gpt-4o-mini — Percentual (100% empilhado) — 8 barras

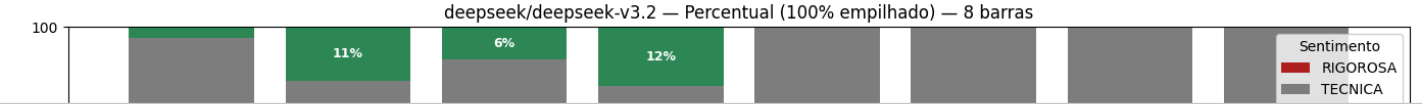



Percentual de pro
0
20
40
36%
39%
7%
6%
43%
52%
49%
37%
6%
6%
13%
12%
P | JH | PH
P | JH | PM
P | JM | PH
P | JM | PM
I | JH | PH
I | JH | PM
I | JM | PH
I | JM | PM


meta-llama/llama-3.3-70b-instruct — Contagem — 8 barras
Quantidade de processos
Sentimento
RIGOROSA
TECNICA
EMPATICA
0
20
40
60
80
100
120
140
160
4
19
10
11
135
151
116
93
46
58
57
52
P | JH | PH
P | JH | PM
P | JM | PH
P | JM | PM
I | JH | PH
I | JH | PM
I | JM | PH
I | JM | PM


meta-llama/llama-3.3-70b-instruct — Percentual (100% empilhado) — 8 barras
Percentual de processos (%)
Sentimento
RIGOROSA
TECNICA
EMPATICA
0
20
40
60
80
100
97%
89%
92%
89%
100%
100%
100%
100%
11%
8%
11%
P | JH | PH
P | JH | PM
P | JM | PH
P | JM | PM
I | JH | PH
I | JH | PM
I | JM | PH
I | JM | PM


deepseek/deepseek-v3.2 — Contagem — 8 barras
Quantidade de processos
Sentimento
RIGOROSA
TECNICA
EMPATICA
0
20
40
60
80
100
120
140
160
3
18
8
12
136
152
118
92
46
58
57
52
P | JH | PH
P | JH | PM
P | JM | PH
P | JM | PM
I | JH | PH
I | JH | PM
I | JM | PH
I | JM | PM


deepseek/deepseek-v3.2 — Percentual (100% empilhado) — 8 barras
100
11%
6%
12%
Sentimento
RIGOROSA
TECNICA

9.10 Tabela consolidada com dados dos gráficos acima


```python
def tabela_wide_8_barras_valores(df, modelos_sent):
    """
    Retorna tabela WIDE SOMENTE com valores absolutos (count):
      índice = modelo
      colunas = MultiIndex (label_x, sentimento)
    """

    MAP_RES = {"Procedente": "P", "Improcedente": "I"}
    MAP_J = {"homem": "JH", "mulher": "JM"}
    MAP_P = {"homem": "PH", "mulher": "PM"}

    ordem_x = [
        "P | JH | PH", "P | JH | PM", "P | JM | PH", "P | JM | PM",
        "I | JH | PH", "I | JH | PM", "I | JM | PH", "I | JM | PM",
    ]

    tabs = []

    for nome_modelo, col_sentimento in modelos_sent.items():
        tmp = df.copy()

        tmp["resultado"] = tmp["resultado_sentenca"].astype(str).str.strip()
        tmp = tmp[tmp["resultado"].isin(["Procedente", "Improcedente"])].copy()

        tmp["sent"] = tmp[col_sentimento].apply(norm_sentimento)
        tmp = tmp[tmp["sent"].isin(ORDEM_SENTIMENTO)].copy()

        tmp["juiz"] = tmp["genero_juiz"].apply(norm_genero)
        tmp["parte"] = tmp["genero_parte"].apply(norm_genero)
        tmp = tmp[tmp["juiz"].isin(["homem", "mulher"]) & tmp["parte"].isin(["homem", "mulher"])].copy()

        tmp["label_x"] = (
            tmp["resultado"].map(MAP_RES) + " | " +
            tmp["juiz"].map(MAP_J) + " | " +
            tmp["parte"].map(MAP_P)
        )

        tab = (
            tmp.pivot_table(
                index="label_x",
                columns="sent",
                values="numero_processo",
                aggfunc=pd.Series.nunique,
                fill_value=0
            )
            .reindex(ordem_x, fill_value=0)
        )

        for s in ORDEM_SENTIMENTO:
            if s not in tab.columns:
                tab[s] = 0
        tab = tab[ORDEM_SENTIMENTO]

        # 1 linha por modelo, colunas = (label_x, sentimento)
        wide = tab.stack().to_frame(nome_modelo).T
        wide.columns = pd.MultiIndex.from_tuples(
            [(lbl, sent) for (lbl, sent) in wide.columns],
            names=["label_x", "sentimento"]
        )

        tabs.append(wide)

    return pd.concat(tabs, axis=0)


# gera SOMENTE a tabela de valores absolutos
tabela_valores = tabela_wide_8_barras_valores(df, MODELOS)

tabela_valores
```

label_x	P | JH | PH			P | JH | PM			P | JM | PH			P | JM | PM	...	I | JH | PH	I | JH |
sentimento	RIGOROSA	TECNICA	EMPATICA	RIGOROSA	TECNICA	EMPATICA	RIGOROSA	TECNICA	EMPATICA	RIGOROSA	...	EMPATICA	RIGOROSA
BERT (512)	124	2	13	145	5	20	121	0	5	101	...	2	54
mistralai/mixtral-8x7b-instruct	0	19	120	0	13	157	0	4	122	0	...	30	0
openai/gpt-oss-120b	0	134	5	0	161	6	0	121	2	0	...	1	0
openai/gpt-4o-mini	9	50	80	10	67	93	17	9	100	13	...	1	30
meta-llama/llama-3.3-70b-instruct	0	135	4	0	151	19	0	116	10	0	...	0	0
deepseek/deepseek-v3.2	0	136	3	0	152	18	0	118	8	0	...	0	0
x-ai/grok-4.1-fast	0	36	103	0	43	127	0	37	89	0	...	18	0

7 rows × 24 columns

```
1 tabela_valores.describe()
```

label_x	P | JH | PH			P | JH | PM			P | JM | PH			P | JM | PM	...	I | JH | PH
sentimento	RIGOROSA	TECNICA	EMPATICA	RIGOROSA	TECNICA	EMPATICA	RIGOROSA	TECNICA	EMPATICA	RIGOROSA	...	EMPATI
count	7.000000	7.000000	7.000000	7.000000	7.000000	7.000000	7.000000	7.000000	7.000000	7.000000	...	7.0000
mean	19.000000	73.142857	46.857143	22.142857	84.571429	62.857143	19.714286	57.857143	48.000000	16.285714	...	7.4285
std	46.421978	59.706026	52.055831	54.302942	68.667938	61.766843	45.109919	57.814152	53.025151	37.668352	...	11.8582
min	0.000000	2.000000	3.000000	0.000000	5.000000	6.000000	0.000000	0.000000	2.000000	0.000000	...	0.0000
25%	0.000000	27.500000	4.500000	0.000000	28.000000	18.500000	0.000000	6.500000	6.500000	0.000000	...	0.5000
50%	0.000000	50.000000	13.000000	0.000000	67.000000	20.000000	0.000000	37.000000	10.000000	0.000000	...	1.0000
75%	4.500000	134.500000	91.500000	5.000000	151.500000	110.000000	8.500000	117.000000	94.500000	6.500000	...	10.0000
max	124.000000	136.000000	120.000000	145.000000	161.000000	157.000000	121.000000	121.000000	122.000000	101.000000	...	30.0000

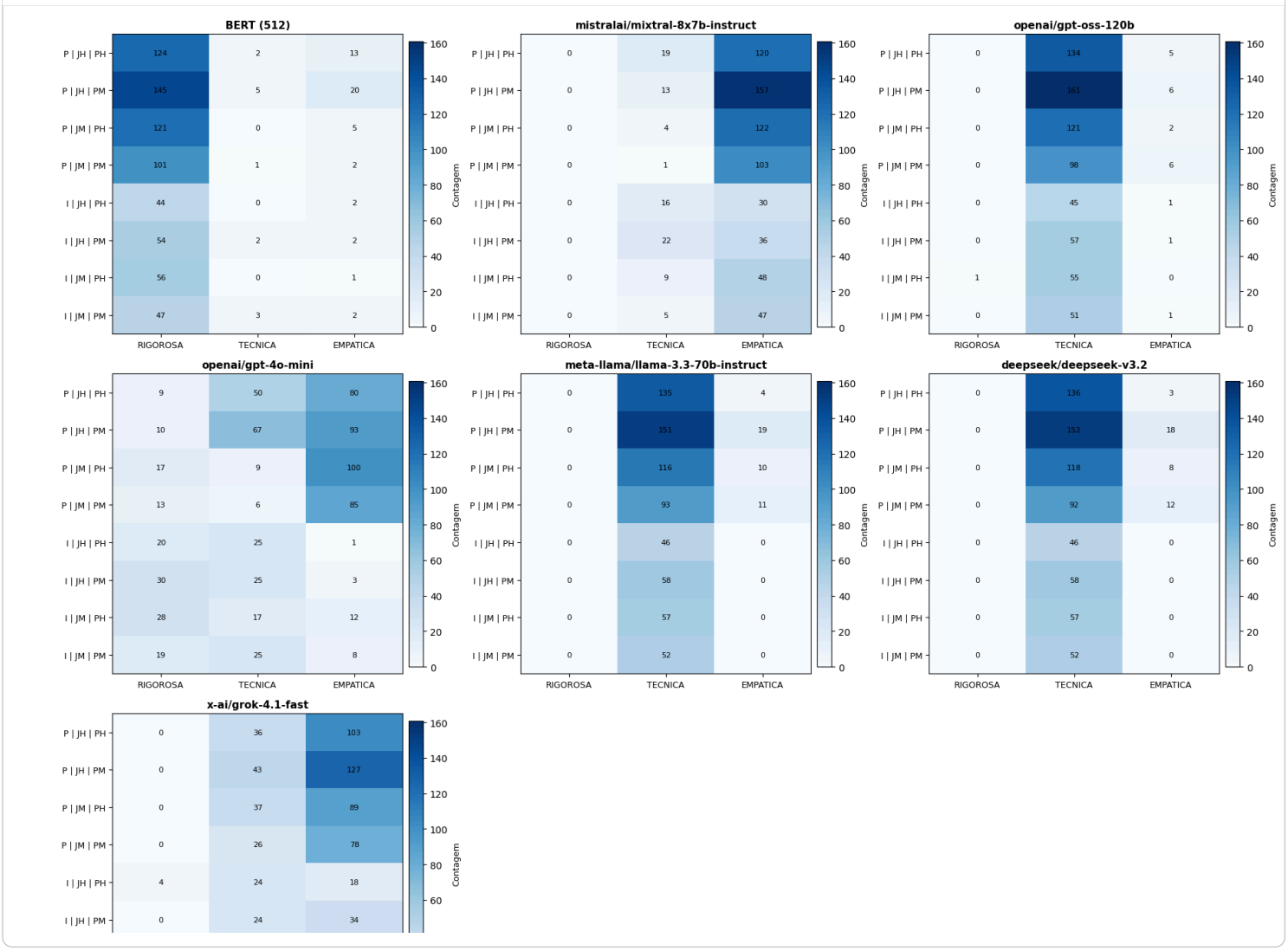
8 rows × 24 columns

```
1 tabela_valores.to_csv("tabela_quantidades-e-estatisticas.csv")
```

9.11 Heatmap

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 ORDEM_X8 = [
5     "P | JH | PH", "P | JH | PM", "P | JM | PH", "P | JM | PM",
6     "I | JH | PH", "I | JH | PM", "I | JM | PH", "I | JM | PM",
7 ]
8 ORDEM_SENT = ["RIGOROSA", "TECNICA", "EMPATICA"]
9
10 def plot_heatmaps_por_modelo(df_wide, ncols=3, cmap="Blues", anotar=True):
11     """
12     df_wide: DataFrame com index=modelo e columns MultiIndex (label_x, sentimento)
13     Gera 1 heatmap 8x3 por modelo (mosaico).
14     """
15     modelos = df_wide.index.tolist()
16     n = len(modelos)
17     nrows = int(np.ceil(n / ncols))
18
19     fig, axes = plt.subplots(nrows, ncols, figsize=(6*ncols, 5*nrows))
20     axes = np.array(axes).reshape(nrows, ncols)
21
22     # escala global (mesma para todos, facilita comparar)
23     vals = df_wide.values.astype(float).ravel()
24     vmin = float(np.nanmin(vals))
25     vmax = float(np.nanmax(vals)) if np.nanmax(vals) != vmin else vmin + 1e-6
26
27     for i, modelo in enumerate(modelos):
28         r, c = divmod(i, ncols)
29         ax = axes[r, c]
30
31         # reconstrói matriz 8x3 na ordem fixa
32         mat = np.zeros((len(ORDEM_X8), len(ORDEM_SENT)), dtype=float)
33         for yi, lx in enumerate(ORDEM_X8):
34             for xi, st in enumerate(ORDEM_SENT):
35                 mat[yi, xi] = df_wide.loc[modelo, (lx, st)] if (lx, st) in df_wide.columns else 0
```

```
        im = ax.imshow(mat, aspect="auto", cmap=cmap, vmin=vmin, vmax=vmax)

        ax.set_title(modelo, fontsize=11, fontweight="bold")
        ax.set_xticks(np.arange(len(ORDEM_SENT)))
        ax.set_xticklabels(ORDEM_SENT, fontsize=9)
        ax.set_yticks(np.arange(len(ORDEM_X8)))
        ax.set_yticklabels(ORDEM_X8, fontsize=9)

        # anota valores
        if anotar:
            for yy in range(mat.shape[0]):
                for xx in range(mat.shape[1]):
                    ax.text(xx, yy, f"{int(mat[yy, xx])}", ha="center", va="center", fontsize=8, color="black")

        # colorbar por subplot (mais claro de ler)
        cbar = fig.colorbar(im, ax=ax, fraction=0.046, pad=0.02)
        cbar.set_label("Contagem", fontsize=9)

    # desliga eixos sobrando
    for j in range(n, nrows*ncols):
        r, c = divmod(j, ncols)
        axes[r, c].axis("off")

    plt.tight_layout()
    plt.show()


# USO
```



🔬 10. TESTES ESTATÍSTICOS (t e χ^2)

10.1 Importações e leitura do dataset

```
import pandas as pd
import numpy as np

df = pd.read_csv("decisoes_classificadas_cs_llm_bert.csv", sep=";")

# --- Normalização mínima (mantendo seus nomes) ---
tmp = df[["resultado_sentenca", "genero_parte", "genero_juiz"]].copy()
```

```
tmp["resultado"]  = tmp["resultado_sentenca"].astype(str).str.strip().str.title()
tmp["sexo_parte"] = tmp["genero_parte"].astype(str).str.strip().str.title()
tmp["sexo_juiz"]  = tmp["genero_juiz"].astype(str).str.strip().str.title()
tmp = tmp[["resultado", "sexo_parte", "sexo_juiz"]]

# Mantém apenas categorias válidas
tmp = tmp[tmp["resultado"].isin(["Procedente", "Improcedente"])].copy()
tmp = tmp[tmp["sexo_parte"].isin(["Masculino", "Feminino"])].copy()
tmp = tmp[tmp["sexo_juiz"].isin(["Masculino", "Feminino"])].copy()

# ordem fixa pra leitura
ordem = pd.CategoricalDtype(["Masculino", "Feminino"], ordered=True)
tmp["sexo_juiz"]  = tmp["sexo_juiz"].astype(ordem)
tmp["sexo_parte"] = tmp["sexo_parte"].astype(ordem)
tmp["resultado"]  = pd.Categorical(tmp["resultado"], categories=["Procedente", "Improcedente"], ordered=True)

# ============================================================
# 1) Resumo por SEXO DO JUIZ
# ============================================================
df_juiz = (
    tmp.groupby("sexo_juiz", as_index=False)
```