

UNIVERSIDADE PRESBITERIANA MACKENZIE
Faculdade de Computação e Informática
Ciência de Dados

Projeto Aplicado II - IMBD

Italo Aparecido Lopes ¹ - italo.lopes@mackenzista.com.br
Gabriel Chaves Goncalves ² - gabrielchaves.goncalves@mackenzista.com.br



O que é o IMDb?



Imagen 1: Logo da Organização

- Maior base de dados de filmes, séries e entretenimento, criada em 1990.
- Contém informações detalhadas sobre produções, elencos, avaliações e críticas.
- Desde 1998, faz parte da Amazon.
- Referência global para fãs, críticos e profissionais do setor audiovisual.

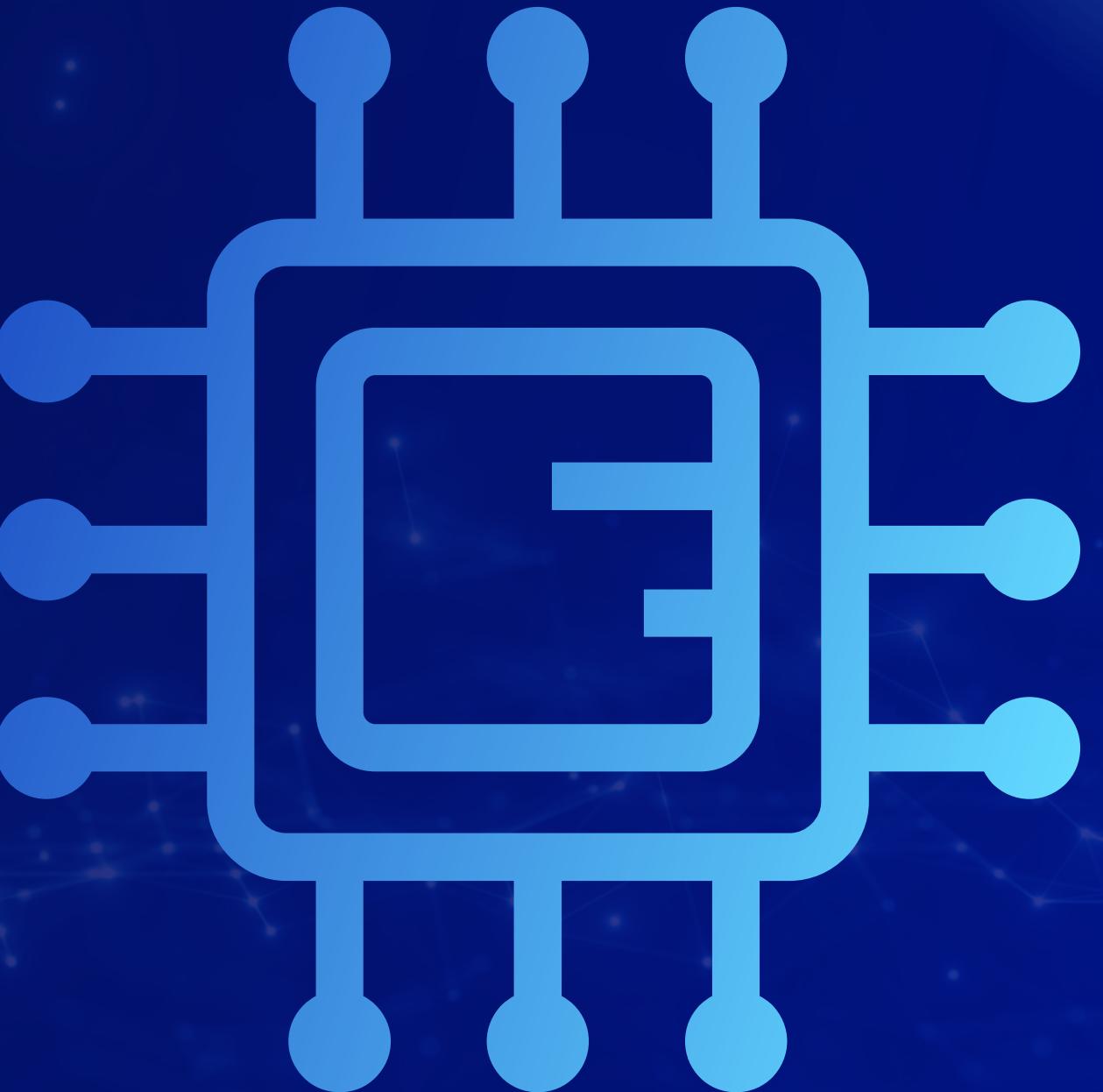
Por que analisar o IMDb?

- Plataforma líder em avaliações de filmes.
- Possibilidade de entender padrões de opinião do público classificadas em sentimentos positivos e negativos
- Base rica em classificação textual o que permite uma maior exploração do algoritmo de aprendizado supervisionado.



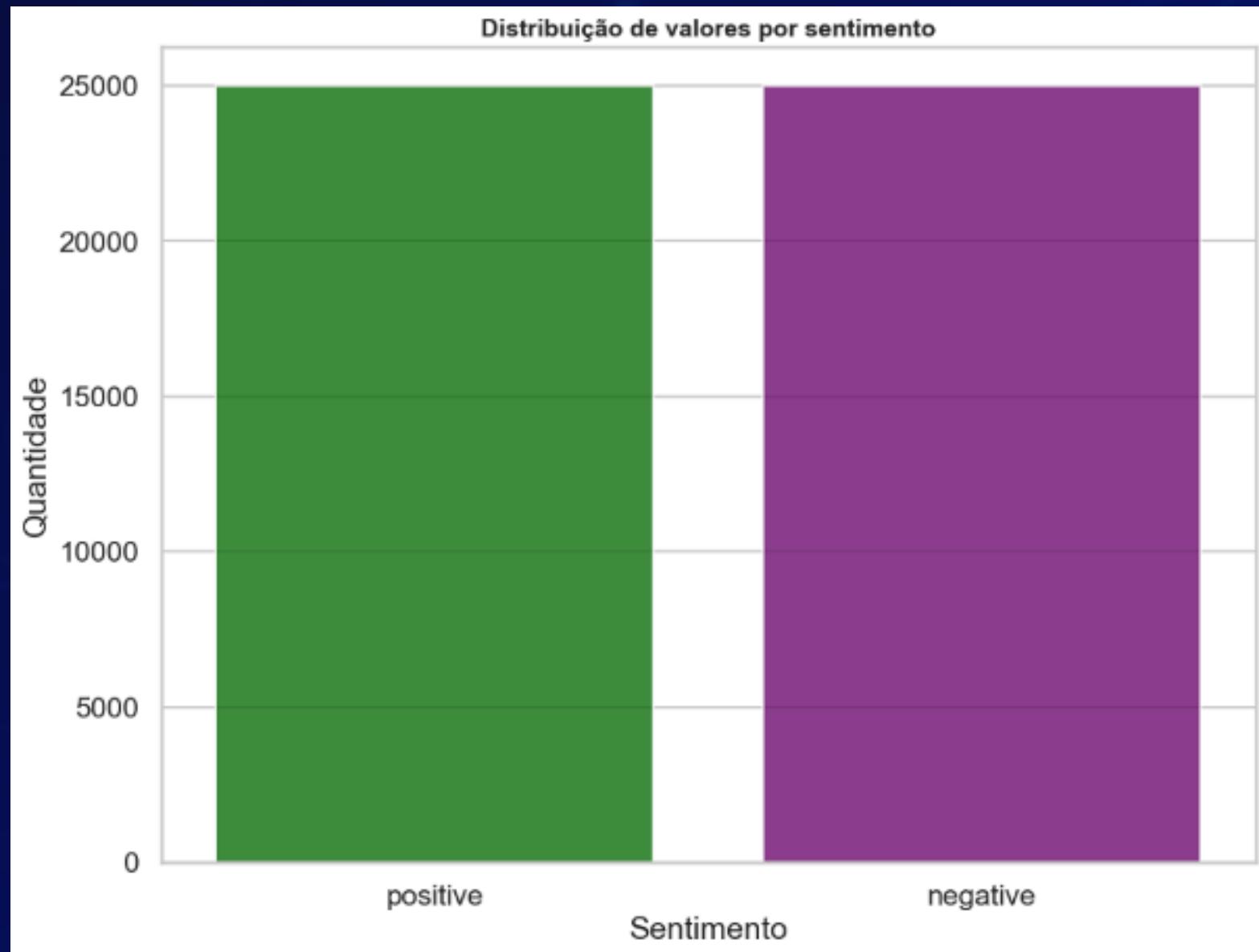
Imagen 1: Logo da Organização

Análise Exploratória de Dados



Descrição da base de dados

- 50.000 avaliações textuais, divididas entre positivas e negativas.
- Colunas principais: “review” (texto da avaliação) e “sentiment” (positivo/negativo).
- Dados平衡eados: 25.000 positivos, 25.000 negativos.



Escolha do método de remoção de Outliers

- Distribuição assimétrica dos dados
- Uso do método IQR para detectar outliers
- Ajuste do fator IQR para evitar remoção excessiva
- Equilíbrio entre limpeza dos dados e preservação de informações relevantes

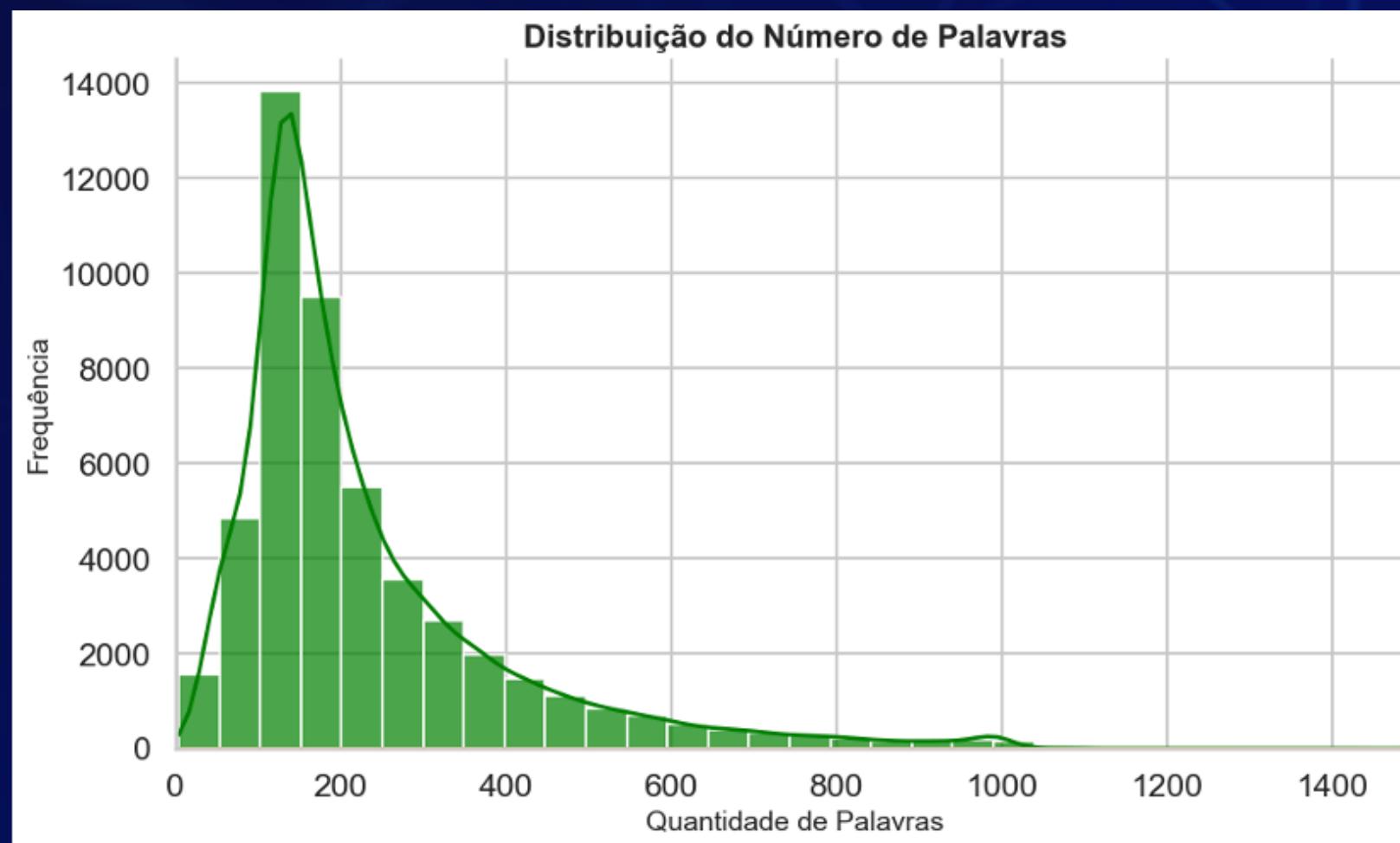


Imagen 3: Histograma da distribuição do número de palavras

Apresentação dos resultados

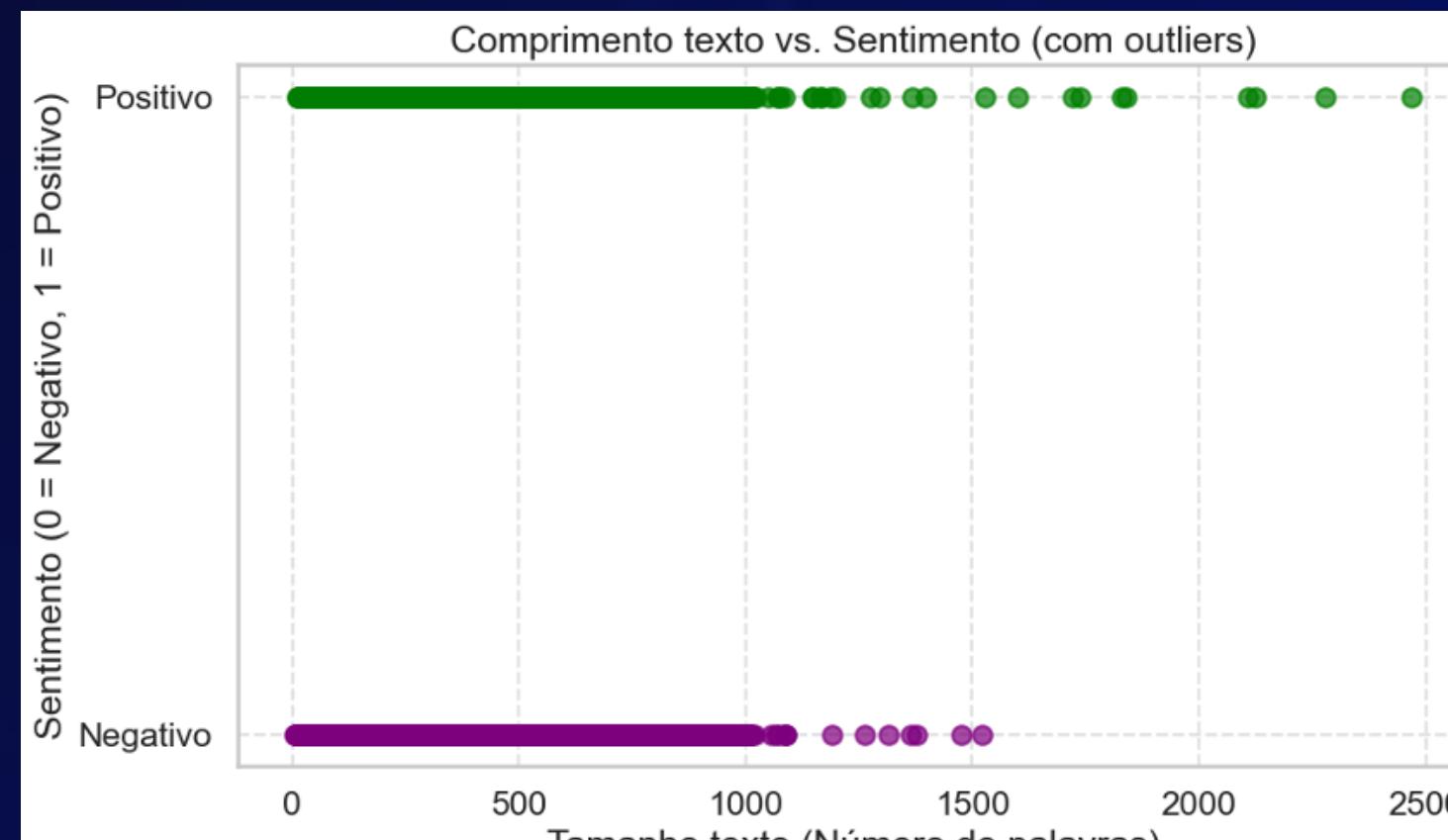


Imagen 4: Comprimento texto vs sentimento (com Outliers)

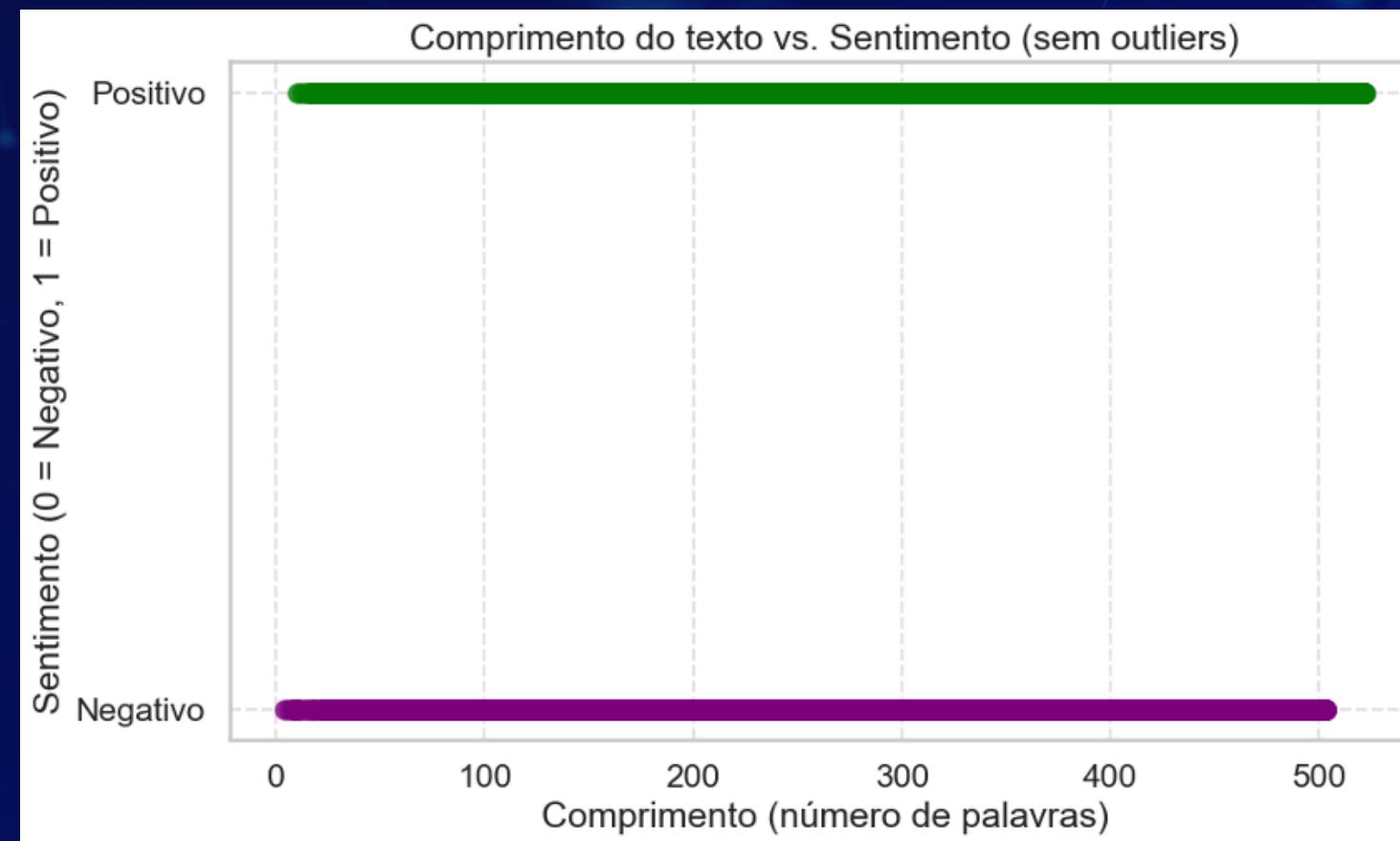


Imagen 5: Comprimento texto vs sentimento (sem Outliers)

- Dados com Outliers
- Fator IQR (1,5)
- Resultado: conjunto de dados mais homogêneo e adequado para análise, mas ainda refletindo o comportamento real dos usuários.

Quantidade Total de Palavras por Sentimento (NLTK)

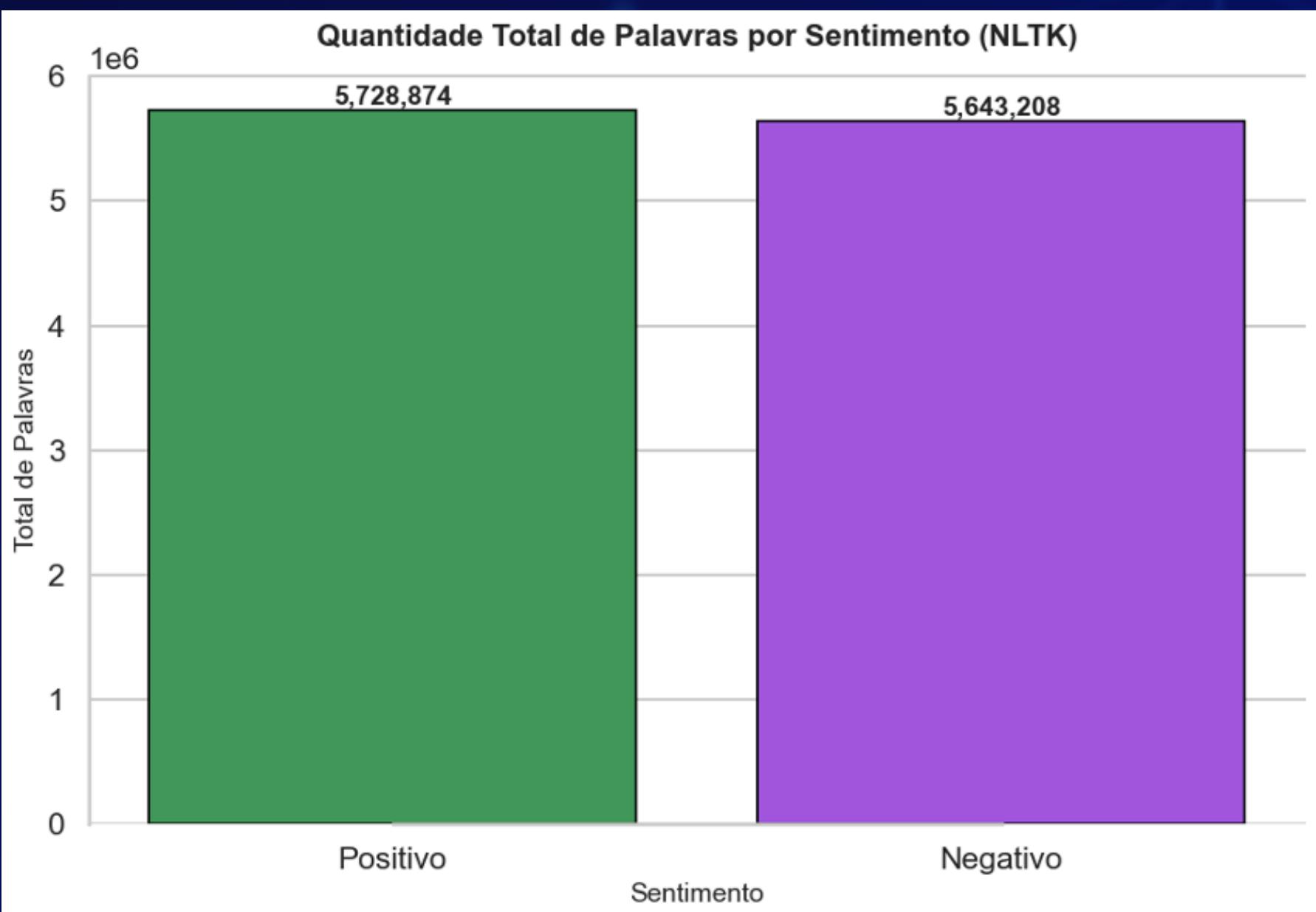
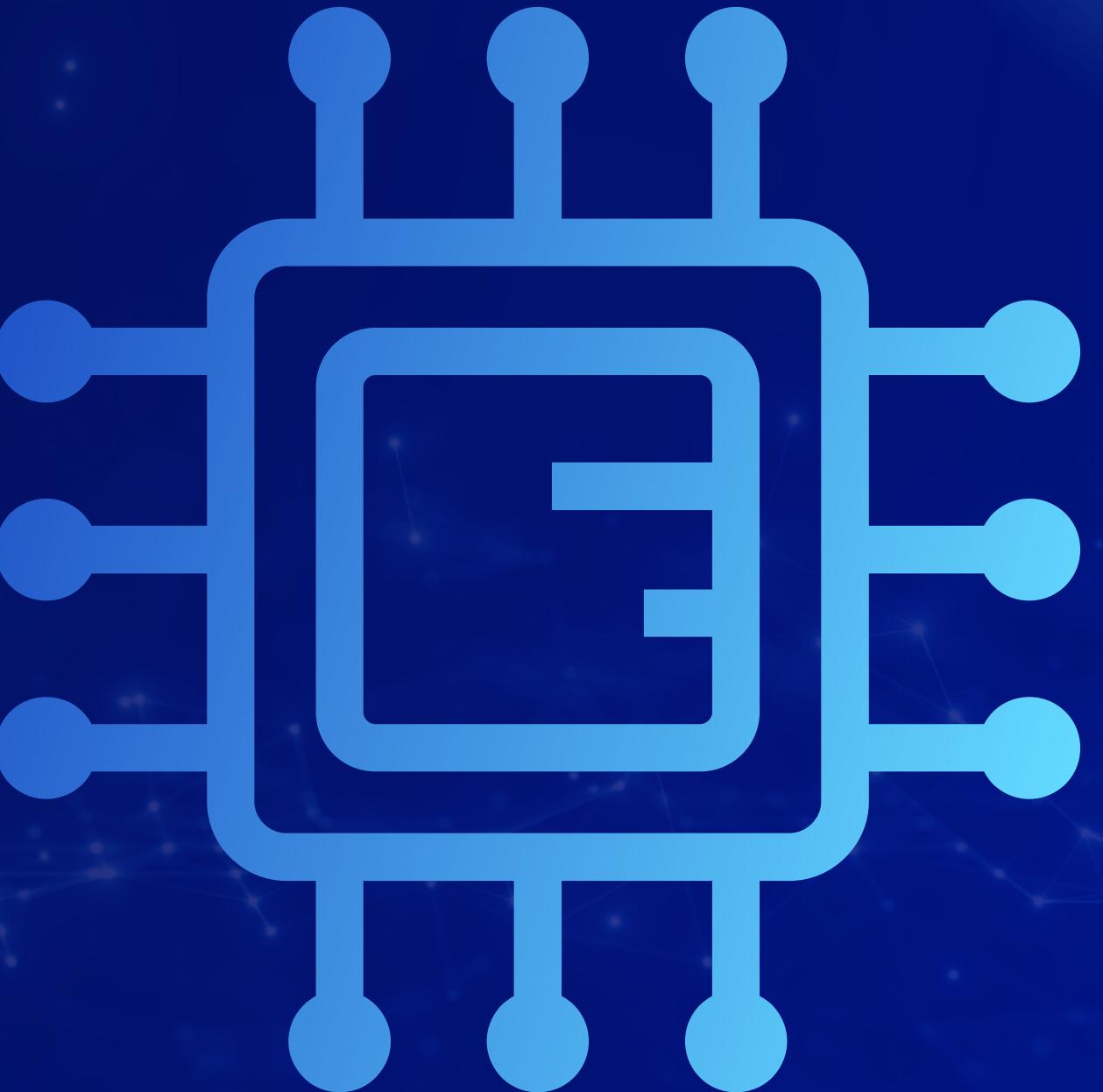


Imagen 6: Quantidade total de palavras por sentimento com uso do pré-processamento textual NLTK

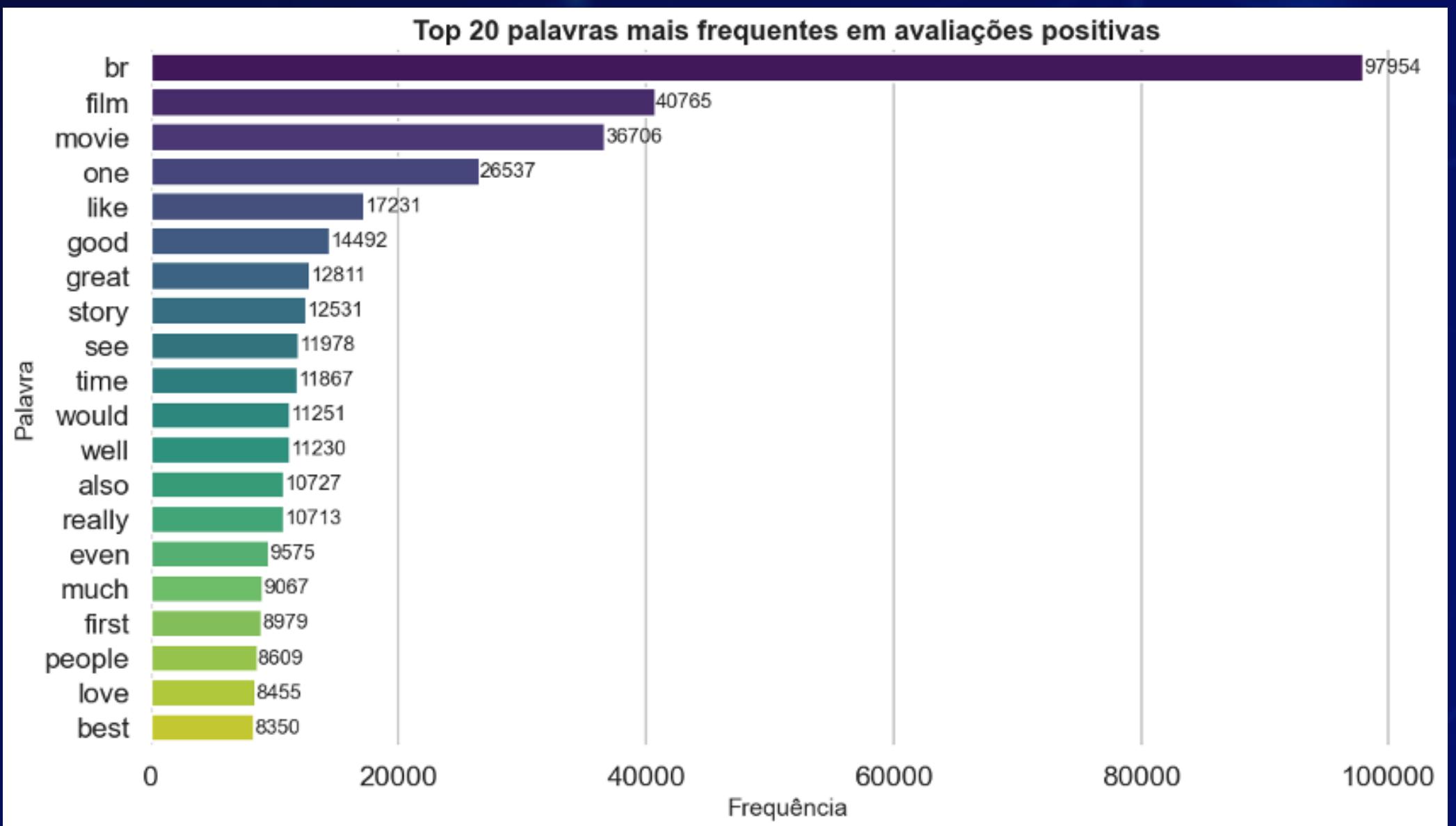
Aplicação do algoritmo NLTK

- Avaliações positivas apresentam um volume maior de palavras (5.728.874) do que as negativas (5.643.208).
- Usuários tendem a detalhar mais suas experiências quando estão satisfeitos.
- Avaliações negativas costumam ser mais objetivas e sucintas.

Insights finais



Avaliações positivas



- As palavras mais usadas estão relacionadas à experiência do filme (ex: “film”, “movie”, “story”).
- Termos positivos como “good”, “great”, “love” e “best” são muito frequentes.
- Usuários satisfeitos costumam ser mais detalhistas em seus comentários.
- Palavras que indicam emoção e aprovação aparecem em destaque.
- Avaliações positivas trazem elogios e recomendações, reforçando o impacto do filme.

Avaliações negativas

- Palavras mais frequentes remetem à crítica direta do filme (ex: “movie”, “film”, “bad”).
- Termos como “would”, “could” e “time” indicam insatisfação ou expectativas não atendidas.
- Usuários insatisfeitos tendem a ser mais objetivos e sucintos nos comentários.
- Foco em apontar problemas, decepções e oportunidades de melhoria.
- Avaliações negativas destacam o que não funcionou, orientando possíveis ajustes para produtores.

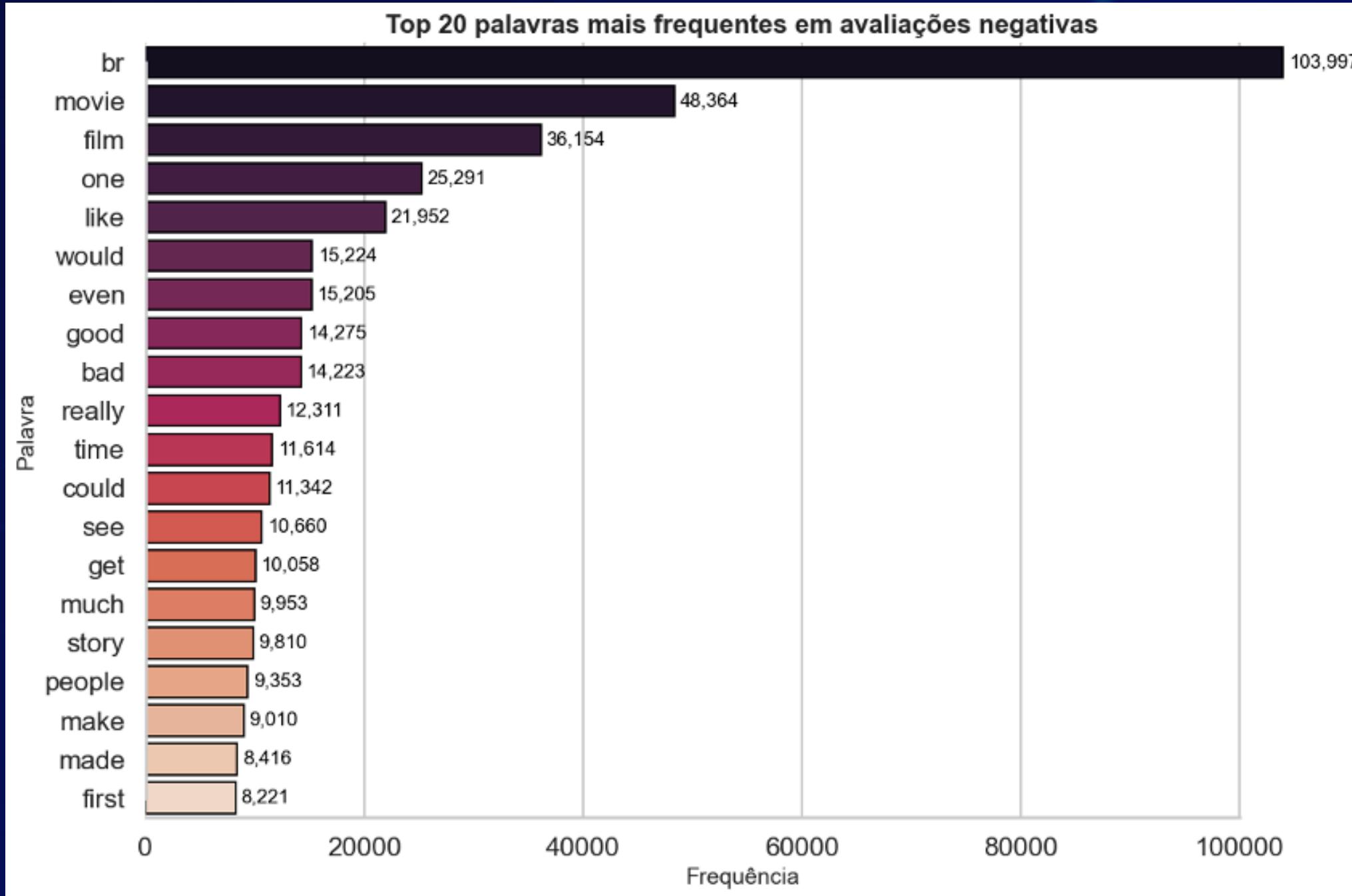


Imagen 7: Top 20 palavras mais frequentes em avaliações positivas

Pré-Processamento dos dados

- Codificação do rótulo
- Particionamento do dataset
- Vetorização

```
[ ] # Importando LabelEncoder para converter as labels.  
from sklearn.preprocessing import LabelEncoder  
  
[ ] # Criando uma variável para armazenar o método escolhido  
encoder = LabelEncoder()  
  
[ ] # Convertendo as labels de 'positive' e 'negative' em número  
y_encoded = encoder.fit_transform(y)
```

Imagen 8: Codificação da variável alvo

```
[ ] # Importando train_test_split para dividir os dados em conjuntos de treino e teste.  
from sklearn.model_selection import train_test_split, cross_validate  
  
[ ] # Dividindo os dados em treino e teste (67% para treino, 33% para teste)  
X_train, X_test, y_train, y_test = train_test_split(X, y_encoded, stratify = y_encoded, test_size=0.33, random_state=42)  
  
[ ] # Verificando o tamanho dos conjuntos de dados  
X_train.shape, X_test.shape, y_train.shape, y_test.shape  
→ ((33500, 101895), (16500, 101895), (33500,), (16500,))
```

Imagen 9: Particionamento do Banco de dados

Pré-Processamento dos dados

Vetorização

- Conversão de textos em representações numéricas
- Muito utilizado em Processamento de Linguagem Natural (PLN)
- Análise de sentimentos, sistema de busca e recomendação
- TF-IDF

```
[ ] # Importando TF-IDF para vetorizar o texto, transformando-o em uma representação numérica.  
from sklearn.feature_extraction.text import TfidfVectorizer  
  
[ ] # Definindo corpus  
corpus = df['review']  
  
▶ # Inicializando e aplicando o vetorizador TF-IDF  
vectorizer = TfidfVectorizer()  
X = vectorizer.fit_transform(corpus)  
  
[ ] # Definindo a variável alvo (sentimento)  
y = df['sentiment']
```

Imagen 10: Vetorização do texto por TF-IDF

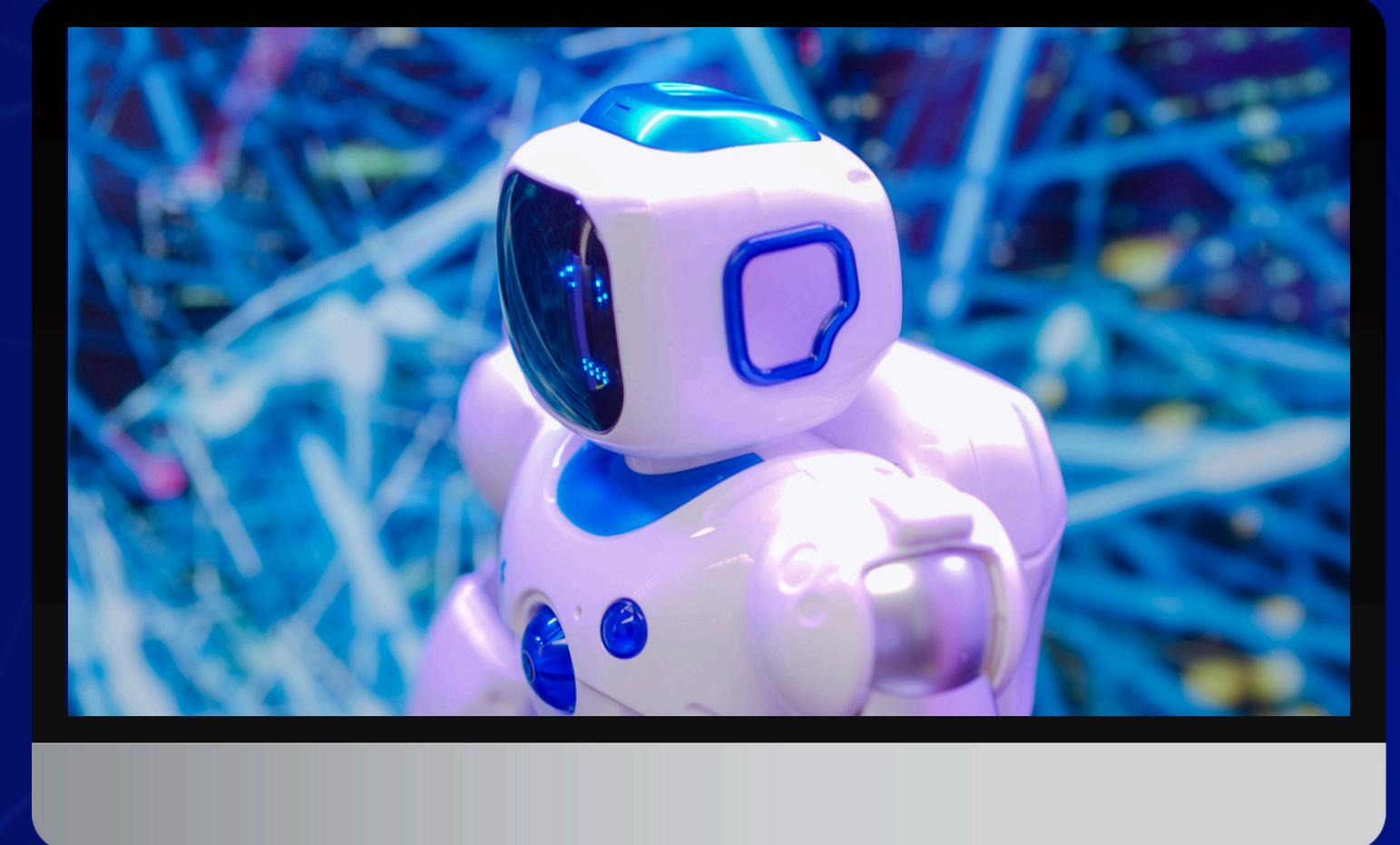
Pré-Processamento dos dados

TF-IDF (Term Frequency–Inverse Document Frequency)

- Utilizado em Processamento de Linguagem Natural (PLN)
- Term Frequency (TF): Refere-se à frequência absoluta ou relativa de um termo em um documento
- Inverse Document Frequency (IDF): Visa reduzir o peso de termos muito frequentes em todos os documentos

Processamento dos dados

- Aplicação dos algoritmos de Machine Learning
- Tunagem de Hiperparâmetros (Regressão Logística)



Processamento

Regressão Logística

```
[25] # Calcula e imprime a acurácia do modelo.  
accuracy = accuracy_score(y_test, y_pred)  
print(f"Acurácia: {accuracy * 100:.2f}%")  
  
→ Acurácia: 89.72%  
  
[26] # Calcula e imprime a precisão do modelo.  
precision = precision_score(y_test, y_pred)  
print(f"Precisão: {precision * 100:.2f}%")  
  
→ Precisão: 88.75%  
  
[27] # Calcula e imprime a sensibilidade do modelo.  
recall = recall_score(y_test, y_pred)  
print(f"Recall: {recall * 100:.2f}%")  
  
→ Recall: 91.09%
```

Imagen 11: Métricas Regressão Logística

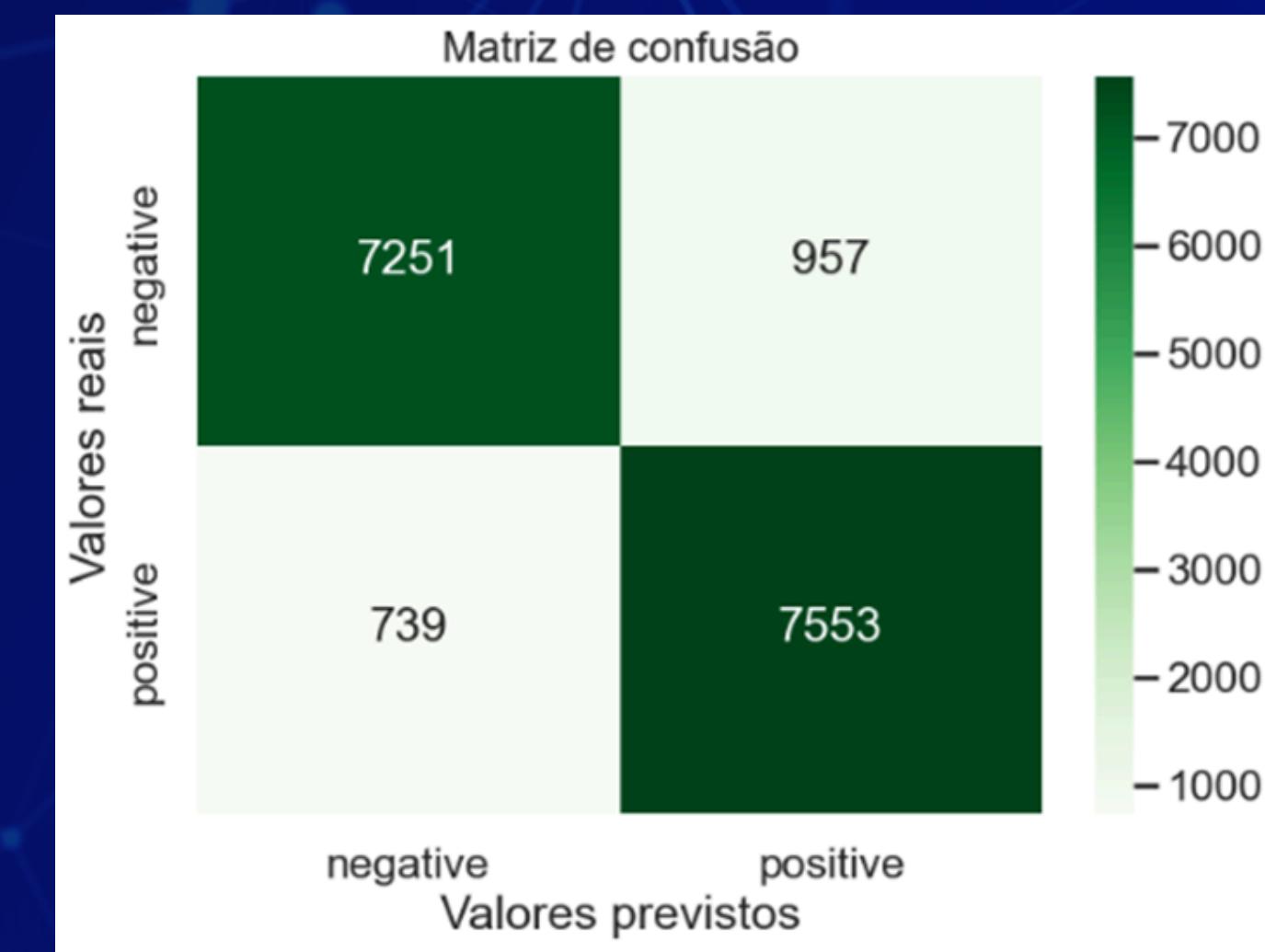
```
▶ # Executa a validação cruzada para obter uma estimativa melhor da acurácia, do precision e do recall do modelo.  
valid_cross = cross_validate(model, X_train, y_train, cv=5, scoring=['accuracy', 'precision', 'recall'])  
print(f"Accuracy: {valid_cross['test_accuracy'].mean()*100:.2f}%")  
print(f"Precision: {valid_cross['test_precision'].mean()*100:.2f}%")  
print(f"Recall: {valid_cross['test_recall'].mean()*100:.2f}%")  
  
→ Accuracy: 89.09%  
Precision: 88.29%  
Recall: 90.09%
```

Imagen 12: Métricas Regressão Logística com Validação Cruzada

Processamento

Regressão Logística

- Verdadeiros Positivos: 7553
- Falsos Negativos: 739
- Verdadeiros Negativos: 7251
- Falsos Positivos: 957



Processamento

Tunagem de Hiperparâmetros

- Precisão
- Generalização
- Robustez
- GridSearch



Processamento

GridSearch

```
[ ] # Definindo o espaço de parâmetros para o GridSearchCV  
param_grid = {  
    'penalty': ['l1', 'l2', 'elasticnet', 'none'],  
    'C': [0.001, 0.01, 0.1, 1, 10, 100],  
    'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'],  
    'max_iter': [100, 200, 300]  
}
```

Imagen 14: Parâmetros de ajuste do modelo

```
[ ] # Melhores parâmetros e melhor pontuação  
print("Melhores parâmetros encontrados: ", grid_search.best_params_)  
print("Melhor pontuação encontrada: ", grid_search.best_score_)
```

↪ Melhores parâmetros encontrados: {'C': 10, 'max_iter': 100, 'penalty': 'l2', 'solver': 'liblinear'}
Melhor pontuação encontrada: 0.890662299034674

Imagen 15: Melhores parâmetros ajustados do modelo

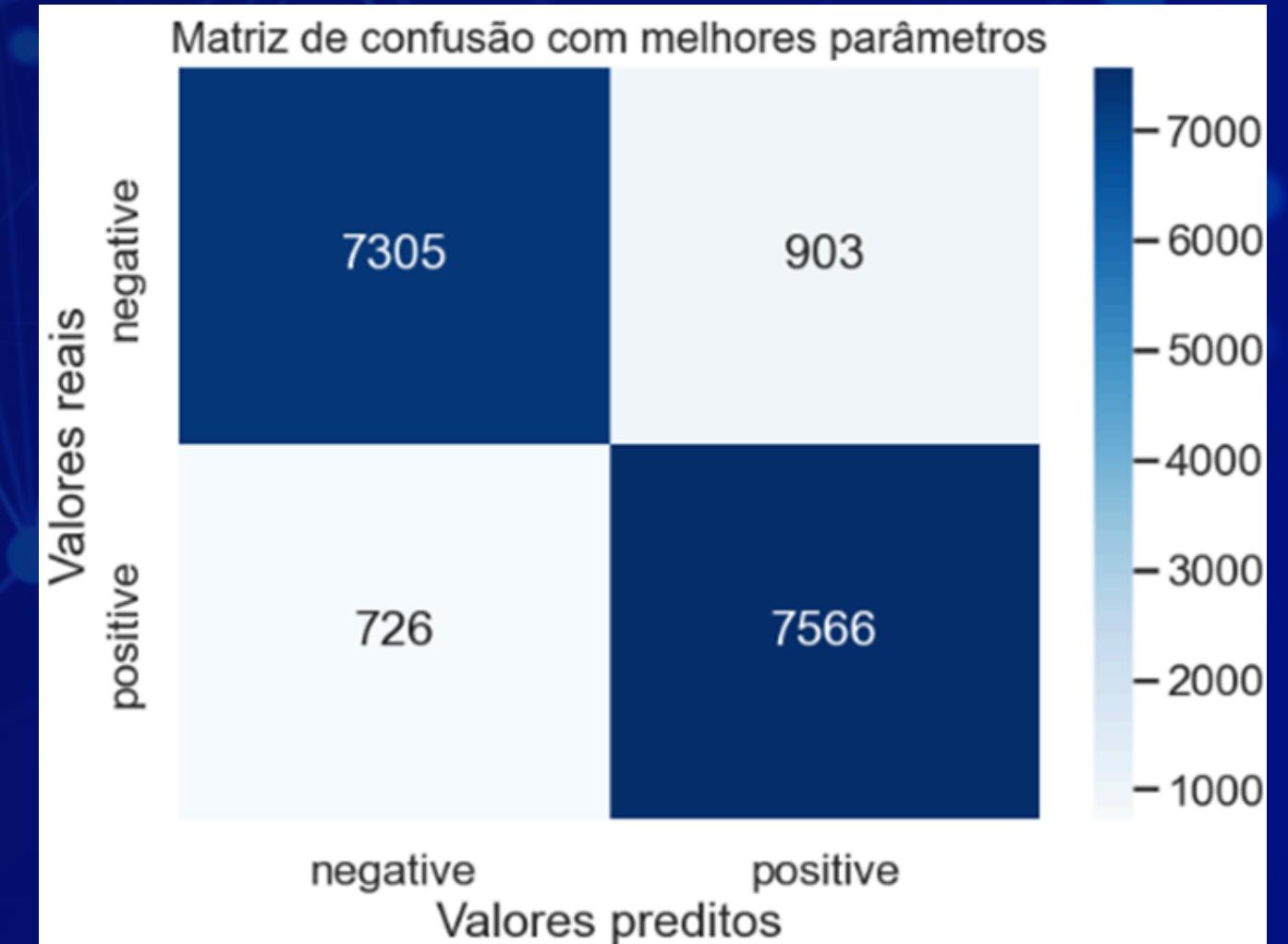
Processamento

GridSearch

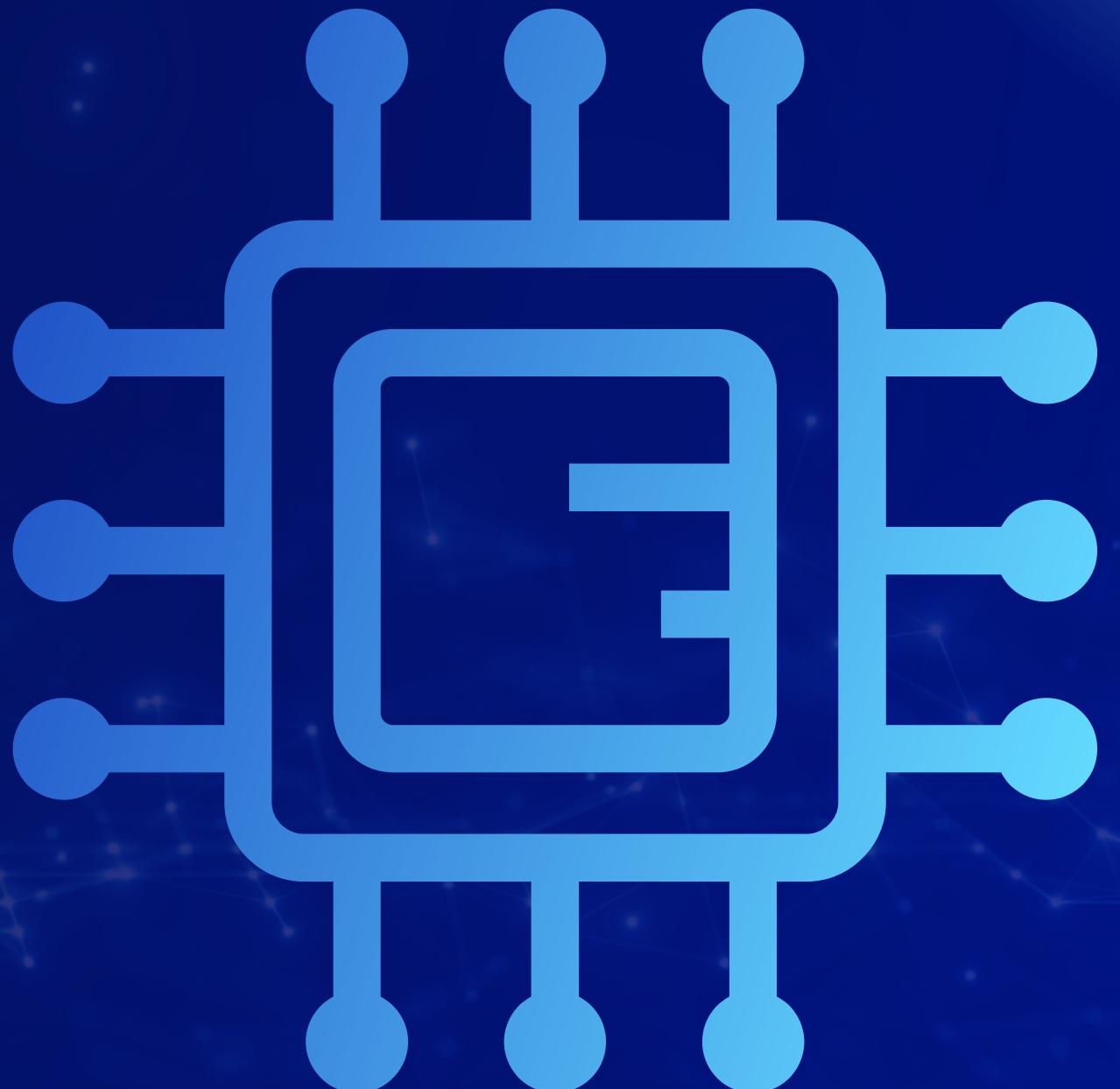
- Verdadeiros Positivos: 7566
- Falsos Negativos: 726
- Verdadeiros Negativos: 7305
- Falsos Positivos: 903

```
[ ] # Calculando e imprimindo a acurácia do modelo  
accuracy_best = accuracy_score(y_test, y_pred_best)  
print(f"\nAcurácia com os melhores parâmetros: {accuracy * 100:.2f}%")  
  
[ ] Acurácia com os melhores parâmetros: 90.13%  
  
[ ] # Calculando e imprimindo a precisão do modelo  
precision_best = precision_score(y_test, y_pred_best)  
print(f"Precisão com os melhores parâmetros: {precision * 100:.2f}%")  
  
[ ] Precisão com os melhores parâmetros: 89.34%  
  
[ ] # Calculando e imprimindo o recall do modelo  
recall_best = recall_score(y_test, y_pred_best)  
print(f"Recall com os melhores parâmetros: {recall * 100:.2f}%")  
  
[ ] Recall com os melhores parâmetros: 91.24%
```

Imagen 16: Métricas com parâmetros ajustados



KNN (K-Nearest Neighbors)



Etapas de testes e decisão

Etapa 1

```
# Usando o melhor modelo encontrado para fazer previsões  
best_knn_model = grid_search_knn.best_estimator_  
y_pred_knn_best = best_knn_model.predict(X_test)  
  
# Calculando a acurácia do melhor modelo  
accuracy_knn_best = accuracy_score(y_test, y_pred_knn_best)  
print(f"Acurácia do melhor modelo KNN: {accuracy_knn_best * 100:.2f}%")  
  
# Calculando a precisão do melhor modelo  
precision_knn_best = precision_score(y_test, y_pred_knn_best)  
print(f"Precisão do melhor modelo KNN: {precision_knn_best * 100:.2f}%")  
  
# Calculando o recall do melhor modelo  
recall_knn_best = recall_score(y_test, y_pred_knn_best)  
print(f"Recall do melhor modelo KNN: {recall_knn_best * 100:.2f}%")  
  
Acurácia do melhor modelo KNN: 50.18%  
Precisão do melhor modelo KNN: 100.00%  
Recall do melhor modelo KNN: 0.86%
```

Imagen 18: Acurácia, precisão e sensibilidade

Etapa 2

```
# Executa a validação cruzada para obter uma estimativa melhor da acurácia, do precision e do recall do modelo (versão KNN).  
valid_cross_knn = cross_validate(knntreino, X_train, y_train, cv=5, scoring=['accuracy', 'precision', 'recall'])  
print(f"Accuracy do KNN (validação cruzada): {valid_cross_knn['test_accuracy'].mean()*100:.4f}%")  
print(f"Precision do KNN (validação cruzada): {valid_cross_knn['test_precision'].mean()*100:.4f}%")  
print(f"Recall do KNN (validação cruzada): {valid_cross_knn['test_recall'].mean()*100:.4f}%")  
✓ 29.8s  
  
Accuracy do KNN (validação cruzada): 72.4060%  
Precision do KNN (validação cruzada): 70.8662%  
Recall do KNN (validação cruzada): 75.8738%
```

Imagen 19: Acurácia, Precisão e Sensibilidade na Validação Cruzada

Etapa 3

```
# Calcula e imprime a acurácia do modelo KNN  
accuracy_knn = accuracy_score(y_test, y_pred_knn)  
print(f"Acurácia do KNN: {accuracy_knn * 100:.2f}%")  
  
# Calcula e imprime a precisão do modelo KNN  
precision_knn = precision_score(y_test, y_pred_knn)  
print(f"Precisão do KNN: {precision_knn * 100:.2f}%")  
  
# Calcula e imprime a sensibilidade do modelo KNN  
recall_knn = recall_score(y_test, y_pred_knn)  
print(f"Recall do KNN: {recall_knn * 100:.2f}%")  
  
✓ 18.6s  
  
Acurácia do KNN: 73.57%  
Precisão do KNN: 71.95%  
Recall do KNN: 77.69%
```

Imagen 20: acurácia, Precisão e Sensibilidade com melhores hiperparâmetros

BI - Dados e Estratégias

Deploy Sistema de Classificação Automática



Imagen 21: Fluxo do novo sistema de avaliação na plataforma IMBD

BI - Dados e Estratégias

Dashboard Interno para moderadores

- Críticas recentes
- Medidas Descritivas sobre as críticas
- Análises sobre críticas
- Dados sobre os usuários
- Classificações automáticas sugeridas



BI – Dados e Estratégias

Otimização da Experiência do usuário

- Visualização do Sentimento Geral
- Filtro por Sentimento
- Filtro por Tamanho da Crítica



BI - Dados e Estratégias

Geração de Receita - B2B

- Relatórios de recepção de lançamento: evolução do sentimento nos primeiros dias de estreia
- Análises de tendência temporal das críticas
- Análise de sentimento georreferenciada
- Análises comparativas: comparar sentimento de filmes concorrentes
- Percepções sobre Atores, Atrizes, Diretores, trilhas sonoras e afins.

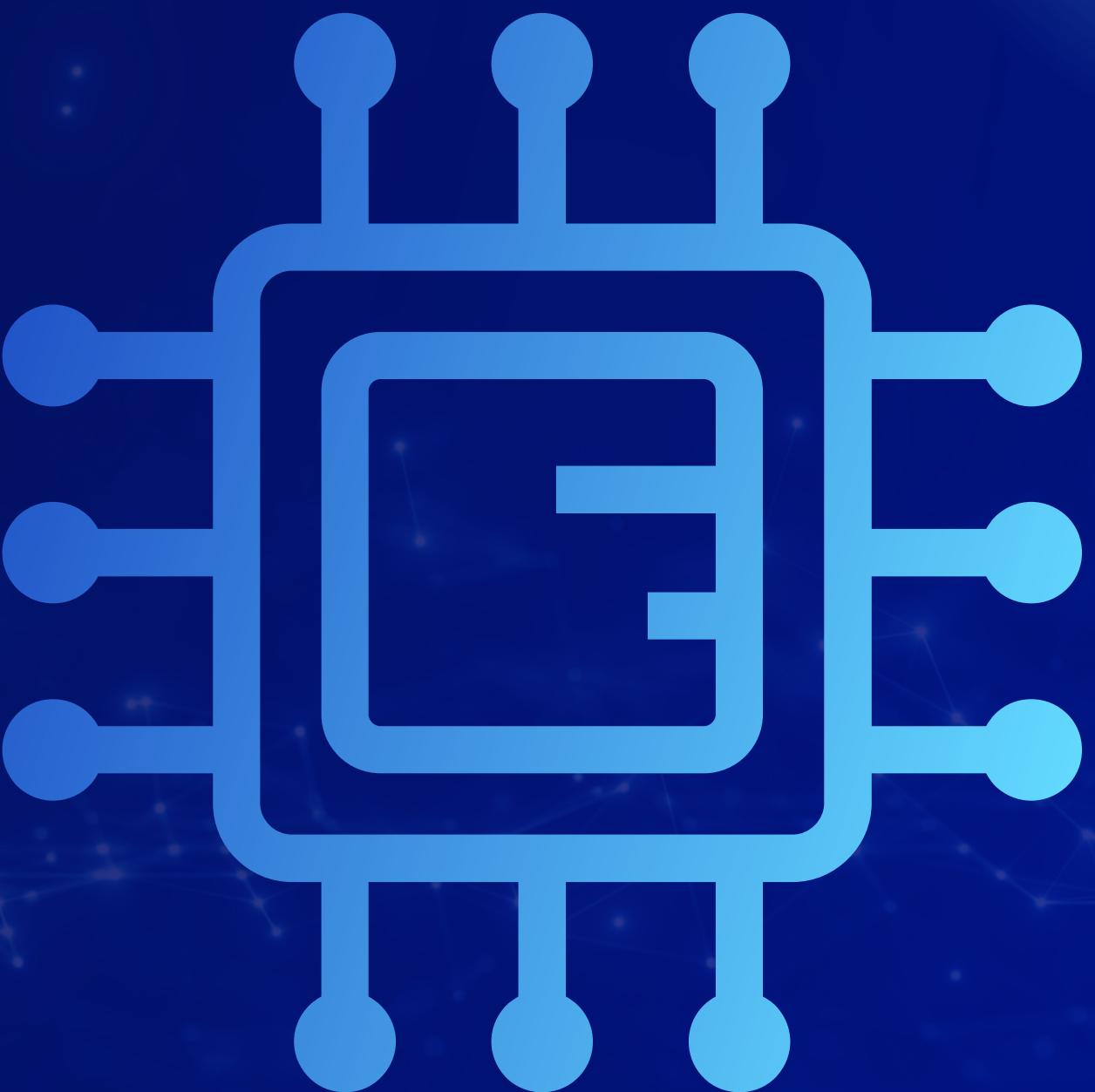
BI – Dados e Estratégias

Melhorias do Algoritmo

- Sugestões Inteligentes Baseadas no Sentimento
- Resumo Inteligente das Críticas



Conclusão



- O classificador de sentimentos automatiza e aprimora a análise das avaliações no IMDb.
- Técnicas de aprendizado supervisionado, como Regressão Logística e KNN, foram fundamentais para o sucesso do projeto.
- A Regressão Logística foi escolhida pelo melhor desempenho, garantindo alta acurácia, precisão e sensibilidade.
- A solução permite integração por APIs e dashboards, facilitando a moderação e personalizando a experiência do usuário.
- Abre novas possibilidades de monetização e geração de insights estratégicos para o IMDb.
- O uso dessas ferramentas reforça a importância da ciência de dados para a inovação e competitividade em plataformas digitais.