Análise de Sentimentos em Avaliações de Produtos

Disciplina: Processamento de Linguagem Natural

Professor: Luciano Barbosa

Link Colab:https://colab.research.google.com/drive/1c2C2npsFrDTuZ_sCrVhc-dFmQWe2ShZ0?usp=sharing

Alunos:
Igor Gabriel de Aráujo Pereira Conde | igapc@cin.ufpe.br

Especialização Deep Learning

Centro de Informática UFPE

Universidade Federal de Pernambuco

Introdução

Visão Geral:

Este projeto busca explorar o potencial do **Processamento de Linguagem Natural (NLP)** para realizar a análise de sentimentos em avaliações de produtos disponibilizados na plataforma **Amazon**. O objetivo principal é desenvolver uma solução robusta que possa classificar os sentimentos em três categorias: **positivo**, **negativo** e **neutro**, utilizando métodos modernos de **machine learning** e **deep learning**. Além disso, visa fornecer insights úteis para empresas que desejam aprimorar a experiência do cliente e tomar decisões baseadas em dados.

Objetivos do Projeto

Objetivos:

Desenvolver um modelo eficiente de análise de sentimentos em avaliações de produtos da Amazon, utilizando dados textuais para identificar e classificar sentimentos.

- **Classificar os sentimentos em três categorias principais**:
- **Positivo**: Sentimentos favoráveis sobre os produtos.
- **Negativo**: Críticas e insatisfações.
- **Neutro**: Avaliações imparciais ou sem emoção explícita.

• **Aplicar técnicas de NLP e aprendizado supervisionado**, incluindo:

•**SVM com Bag of Words (BoW)**: Representação baseada na frequência de palavras, treinada para identificar padrões de sentimentos no texto.

•**In-Context Learning com GPT-Neo**: Uso de prompts pré-definidos para prever sentimentos, combinando exemplos ilustrativos e geração contextualizada de respostas.

•**Utilizar modelos avançados de Deep Learning**, como BERT, para extrair embeddings semânticos e realizar a classificação dos textos.

Metodologia

Sobre o Dataset
Avaliações de produtos extraídas da Amazon, contendo texto completo, resumo e notas.

Dimensão: **913** de um produto especifico, registros com 10 colunas principais, incluindo:

- **Text**: Texto completo da avaliação.
- **Score**: Nota atribuída (1 a 5).
- **HelpfulnessNumerator e HelpfulnessDenominator**: Indicadores de utilidade da avaliação.

Amazon Product Reviews

kaggle Link do Dataset

568k Reviews de Produtos Diferentes

Metodologia

Pré-processamento de Texto

Análise Exploratória de Dados (EDA)

- Visualização da distribuição de Score e identificação de tendências nos textos.
- Geração de nuvem de palavras para palavras mais frequentes em cada categoria de sentimento.
- Análise de correlações entre votos úteis e notas atribuídas.
- Treinamento e Validação (80% para treinamento e 20% para testes.)

Limpeza e Normalização:

- Remoção de caracteres especiais, números e stopwords.
- Conversão para letras minúsculas e aplicação de tokenização.

Técnicas Aplicadas:

- **Bag of Words (BoW)**: Frequência de palavras como representação vetorial.
- **mbeddings**: Relevância das palavras em relação ao corpus.
- **Lematização**: Redução das palavras às suas formas base.

Metodologia

Modelagem de Sentimentos

- **SVM + BoW**:
  - Treinamento de um modelo simples com BoW para capturar padrões básicos de sentimentos.
  - Avaliação de desempenho para identificar limitações de métodos convencionais.
- **Deep Learning com BERT**:
  - Extração de embeddings semânticos de alta qualidade.
  - Classificação dos textos com redes neurais profundas para capturar nuances contextuais.
- **In-Context Learning com GPT-Neo**:
  - Uso de prompts com exemplos ilustrativos para prever sentimentos.
  - Geração de respostas com base no contexto fornecido.

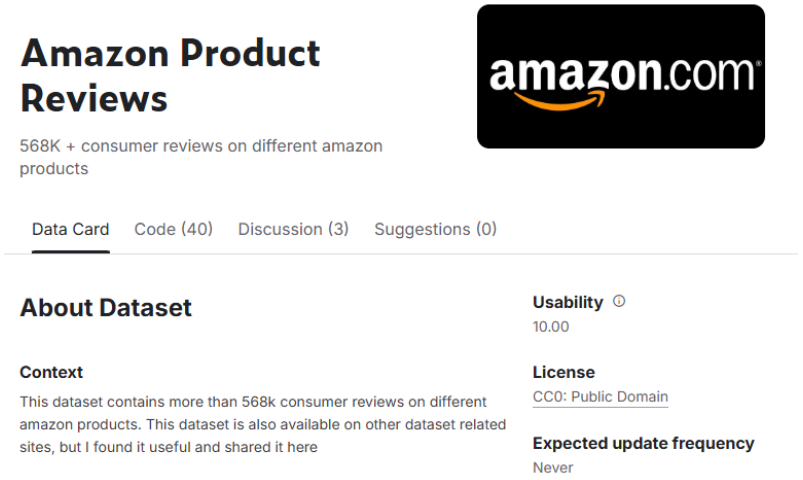
Fonte dos Dados

Fonte dos Dados:

- Avaliações de produtos extraídas da Amazon, contendo texto completo, resumo e notas.

Características dos Dados:

- Total de Avaliações: 368k



Analise Exploratória

EDA

- Analise dos campos principais
- Estrutura e preenchimento dos dados

O dataset possui **913** registros e 10 colunas, conforme a estrutura a seguir:

Estrutura do Dataset:

Id: Identificador único do registro.

ProductId: Identificador do produto avaliado.

UserId: Identificador do usuário que fez a avaliação.

ProfileName: Nome do perfil do usuário.

HelpfulnessNumerator: Número de votos úteis para a avaliação.

HelpfulnessDenominator: Número total de votos recebidos.

Score: Nota atribuída ao produto (1 a 5).

Time: Timestamp da avaliação (Unix).

Summary: Resumo da avaliação.

Text: Texto completo da avaliação.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 913 entries, 0 to 912
Data columns (total 10 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Id                      913 non-null    int64
 1   ProductId               913 non-null    object
 2   UserId                  913 non-null    object
 3   ProfileName             913 non-null    object
 4   HelpfulnessNumerator    913 non-null    int64
 5   HelpfulnessDenominator  913 non-null    int64
 6   Score                   913 non-null    int64
 7   Time                    913 non-null    int64
 8   Summary                 913 non-null    object
 9   Text                    913 non-null    object
dtypes: int64(5), object(5)
memory usage: 71.5+ KB
```

		pfulnessNumerator
0	jaimoi "Appreciator of good music"	0
1	Linda Painchaud-Steinman "PARK EDGE BOOKS"	0
2	carwash169	0
3	P. Titus "Knitgirlll"	0
4	Jessica	0

	HelpfulnessDenominator	Score	Time
0	0	5	1343433600
1	0	5	1343433600
2	0	5	1343433600
3	0	5	1343433600
4	0	5	1343433600

	Summary
0	Delicious!
1	Great Anytime of Day!
2	Very good!
3	Quaker Oats Oatmeal Raisin Mom Voxbox Review C...
4	Quick, simple HEALTHY snack for the kiddos!!!

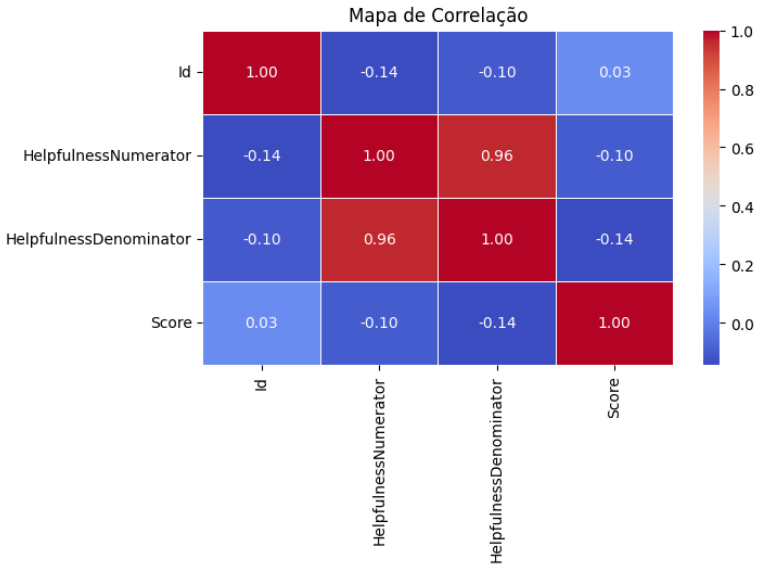
Analise Exploratória

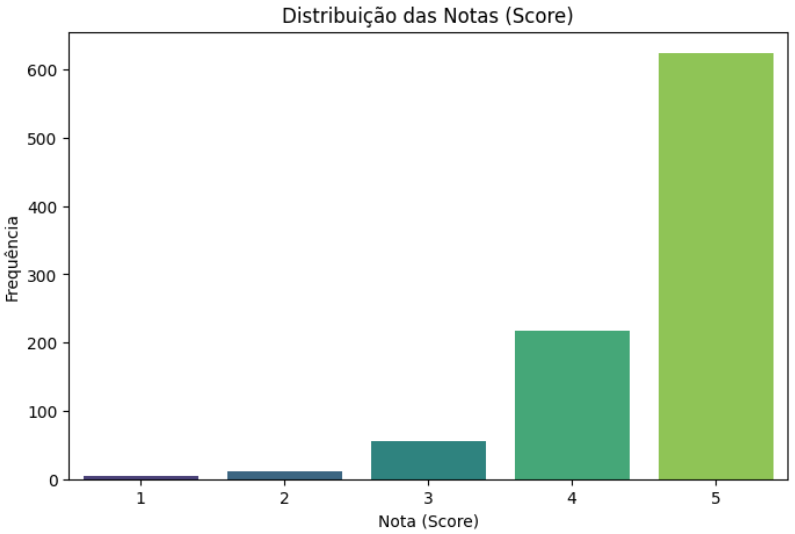
Analise de Texto

- **Coluna Score**:
- **Média**: 4,58 (indica uma tendência geral positiva).
- **Mínimo**: 1 (representa avaliações mais críticas).
- **Máximo**: 5 (avaliações altamente positivas).
- **Desvio Padrão**: 0,71 (baixa variação nas notas).
- **Colunas HelpfulnessNumerator e HelpfulnessDenominator**:
- **Média**:
    - Numerator: 0,045 (poucas avaliações receberam votos úteis).
    - Denominator: 0,050.

- **Máximo**: Ambas chegam a 5 (máximo de votos úteis registrados).
- A maioria das avaliações (75%) não possui votos úteis (valor 0 no percentil 75).

Analise Exploratória





Nuvem de Palavras

• Palavras como "cookie", "good", "taste", "love" e "soft" são frequentemente mencionadas, indicando um foco positivo nas avaliações sobre sabor e qualidade do produto.

Bag of Words e Modelagem de Tópicos

- Realizado a remoção do Stopwords
- Pré-processamento
- Criação das Features

NLTK

```
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('punkt_tab')


stop_words = set(stopwords.words('english'))
punctuation = set(string.punctuation)

def preprocess_text(text):
    text = text.lower()
    tokens = word_tokenize(text)
    tokens = [word for word in tokens if word not in stop_words and word not in punctuation]
    return " ".join(tokens)

dataset['CleanedText'] = dataset['Text'].apply(preprocess_text)

cleaned_sample = dataset[['Text', 'CleanedText']].head()

cleaned_sample
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
```

	Text	CleanedText
0	I love these cookies! Not only are they healt...	love cookies healthy taste great soft definite...
1	Quaker Soft Baked Oatmeal Cookies with raisins...	quaker soft baked oatmeal cookies raisins deli...
2	I am usually not a huge fan of oatmeal cookies...	usually huge fan oatmeal cookies literally mel...
3	I participated in a product review that includ...	participated product review included sample hr...
4	My kids loved these. I was very pleased to giv...	kids loved pleased give kids quick go healthy ...

```
[ ]  # Implementação do BoW

     vectorizer = CountVectorizer(max_features=100, stop_words='english')
     bow_matrix = vectorizer.fit_transform(dataset['CleanedText'])

     bow_df = pd.DataFrame(bow_matrix.toarray(), columns=vectorizer.get_feature_names_out())

     print(bow_df.head())
```

```
   absolutely  amazing  ate  baked  best  better  big  bit  bite  bought  ...  \
0           0        0    0      0     0       0    0    0     0       0  ...
1           0        0    0      1     0       0    0    1     0       0  ...
2           0        0    0      0     0       0    0    0     0       0  ...
3           0        0    0      1     0       0    0    0     0       0  ...
4           0        0    0      0     0       0    0    0     0       0  ...

   thought  time  treat  tried  try  voxbox  wait  wrapped  year  yummy
0        0     0      0      0    0       0     0        0     0      0
1        0     0      1      0    1       0     0        0     0      0
2        0     0      0      0    0       0     0        0     0      0
3        0     0      0      1    0       1     0        0     0      0
4        0     0      0      0    0       0     0        0     0      0

[5 rows x 100 columns]
```

Extração de Tópicos Usando LDA

- Técnica aplicada: Latent Dirichlet Allocation (LDA).
- Objetivo: Identificar os principais tópicos abordados nos textos.
- Configuração: 5 tópicos principais com 10 palavras mais representativas cada.

```
tfidf_vectorizer = TfidfVectorizer(max_features=1000, stop_words='english')
tfidf_matrix = tfidf_vectorizer.fit_transform(dataset['CleanedText'])

# LDA para modelagem de topicos.
lda_model = LatentDirichletAllocation(n_components=5, random_state=42)
lda_model.fit(tfidf_matrix)

topics = {}
for topic_idx, topic in enumerate(lda_model.components_):
    topics[f"Topic {topic_idx + 1}"] = [tfidf_vectorizer.get_feature_names_out()[i] for i in topic.argsort()[-10:]]

topics_df = pd.DataFrame.from_dict(topics, orient='index', columns=[f"Word {i+1}" for i in range(10)])

print("Top Words for Each Topic:")
print(topics_df)
```

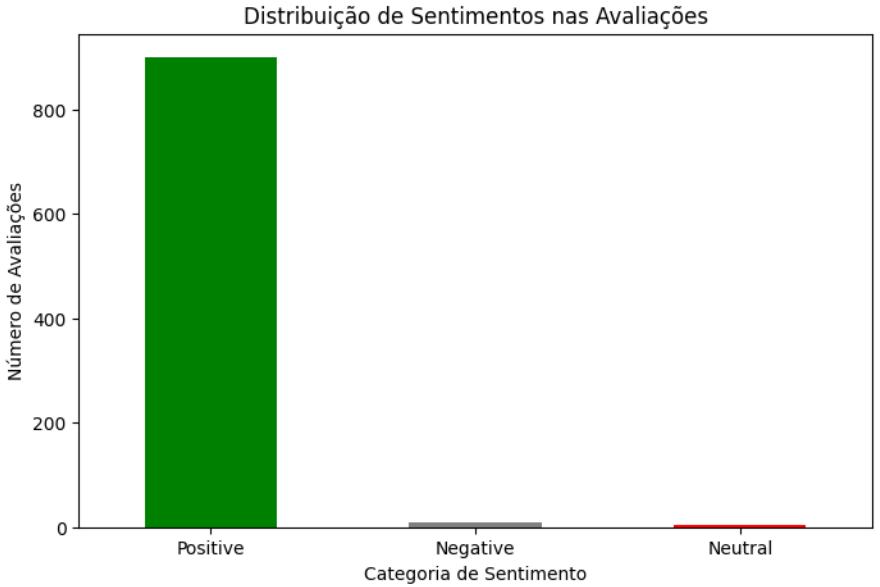
```
Top Words for Each Topic:
          Word 1  Word 2    Word 3       Word 4  Word 5       Word 6  \
Topic 1     gets   trust     looks  combination  lasted  compliments
Topic 2     yumm    pick      heat      crumble     yum         mail
Topic 3     good   loved    quaker        taste    love      oatmeal
Topic 4      try  really    little         love    like      oatmeal
Topic 5  unhealthy  lovers  cravings      suggest     bar         fake

          Word 7 Word 8       Word 9    Word 10
Topic 1     feel   bomb  deliciously     guilty
Topic 2    easily    ask     continue  afternoon
Topic 3     great   soft       cookie    cookies
Topic 4      good   soft      cookies     cookie
```

Analise de Polaridade Textual

•Analisar a polaridade dos textos das avaliações, determinando o sentimento predominante:

•**Positivo**: Polaridade > 0.
•**Neutro**: Polaridade = 0.
•**Negativo**: Polaridade < 0.

	CleanedText	SentimentPolarity
0	love cookies healthy taste great soft definite...	0.380000
1	quaker soft baked oatmeal cookies raisins deli...	0.311818
2	usually huge fan oatmeal cookies literally mel...	0.125000
3	participated product review included sample hr...	0.371000
4	kids loved pleased give kids quick go healthy ...	0.561905
5	really enjoyed individually wrapped big oatmea...	0.270370
6	surprised soft cookie usually buy little debbi...	0.027976
7	filled oats raisins 'll love snack ..... delic...	0.633333
8	recently given complimentary `` vox box '' inf...	0.158333
9	best freshest cookie comes package ate wishing...	0.850000



SVM com Bag-of-Words

Resumo:
Classificar sentimentos com SVM utilizando Bag-of-Words (BoW) como representação vetorial.

Divisão dos Dados:
- Treino: 70%
- Teste: 30%

Resultados:
Acurácia Geral: 98,28%
Categoria Positiva: Alta precisão e recall.

```
w_matrix
y = dataset['SentimentCategory']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.7, random_state=42, stratify=y)

svm_model = LinearSVC(random_state=42)
svm_model.fit(X_train, y_train)

y_pred = svm_model.predict(X_test)

svm_accuracy = accuracy_score(y_test, y_pred)
svm_report = classification_report(y_test, y_pred)

svm_accuracy, svm_reportX = bo
```

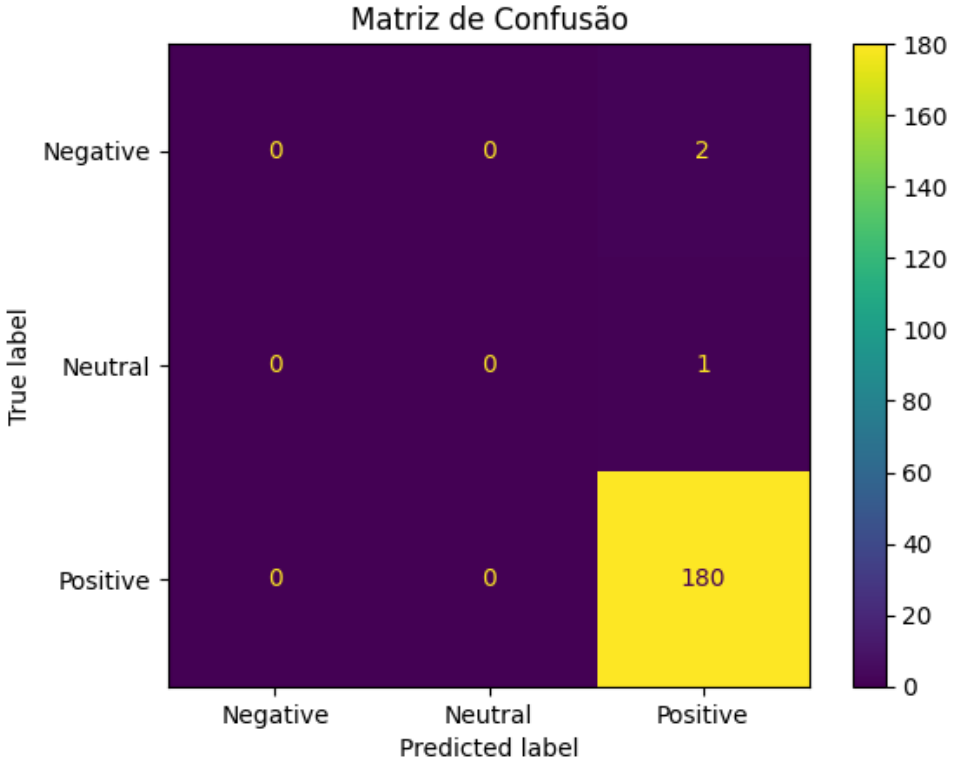
```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarnin
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarnin
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarnin
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
(0.9828125,
 '              precision    recall  f1-score   support\n\n    Negative       0.00      0.00      0.00
1.00      0.99       631\n\n    accuracy                           0.98       640\n   macro avg
```

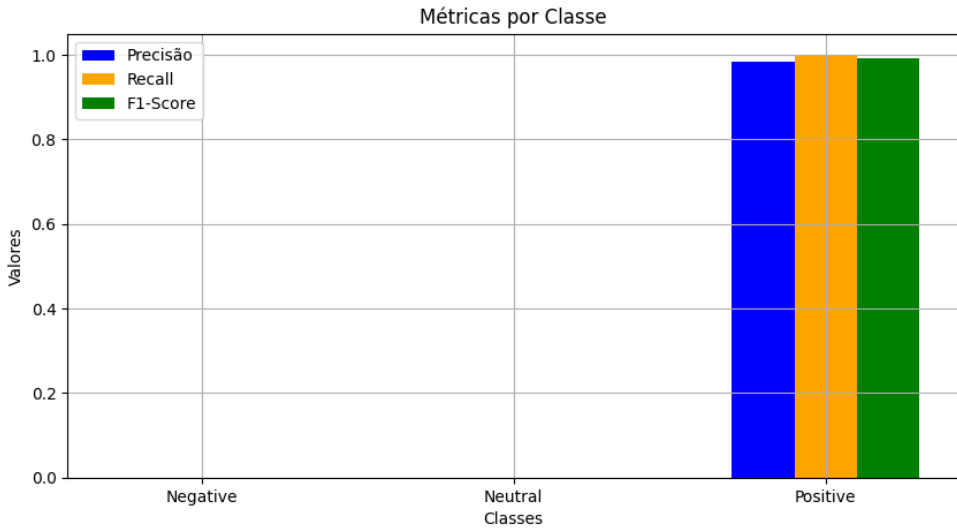
BERT - Classificação de Sentimentos

1. **Pré-processamento e Tokenização**:
   Utilização do **BERT tokenizer** para dividir os textos em tokens.
   Textos ajustados para comprimento máximo de 256 tokens com padding e truncamento.
2. **Divisão dos Dados**:
   Treino: **80%**
   Teste: **20%**, com estratificação para manter a proporção entre categorias de sentimento.
3. **Treinamento**:
   Modelo pré-treinado: bert-base-uncased.
   Treinamento por **3 épocas**, utilizando o otimizador **AdamW** e escalonamento de aprendizado (Scheduler).
   Uso de **mixed precision training** para otimização computacional.
4. **Métricas de Avaliação**:
   Acurácia Geral: Desempenho total do modelo.
   Precisão, Recall e F1-Score: Avaliação detalhada por categoria de sentimento.

- **Acurácia Geral**: 98,36%.
- **Positivo**: Excelente performance.
- **Desafios**: Desequilíbrio de dados impactou o desempenho para as categorias "Neutro" e "Negativo".

BERT - Classificação de Sentimentos





In-Context Learning com GPT-Neo

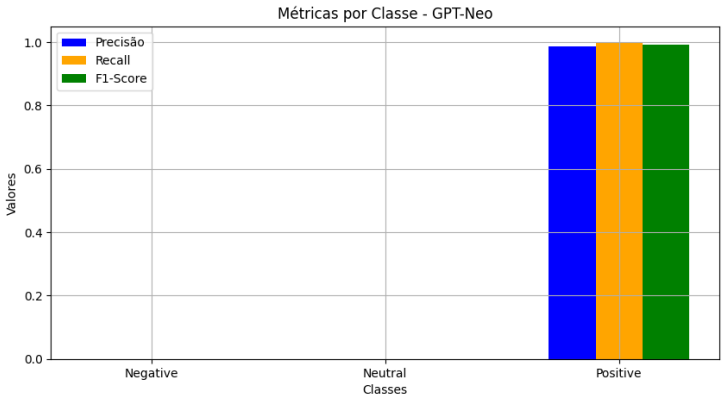
Resumo:
Utilizar In-Context Learning com o modelo GPT-Neo para prever o sentimento das avaliações de produtos.

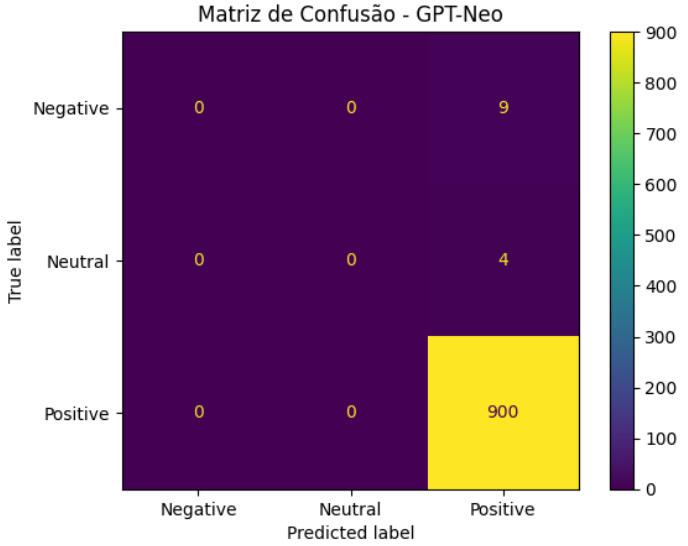
•**Estratégia de Prompting**:
•Um prompt é estruturado com exemplos claros de sentimentos positivos, negativos e neutros.
•O texto da avaliação é adicionado como query no prompt para o modelo prever o sentimento.
•**Predição**:
•O modelo GPT-Neo gera uma saída textual que contém a classificação do sentimento (Positive, Neutral ou Negative).

Resultados
•**Acurácia Geral**: **98,57%**.
•**Detalhamento por Categoria**:
- **Positivo**: Excelente desempenho, devido à predominância dessa classe no dataset.
- **Neutro e Negativo**: Desempenho limitado pelo desbalanceamento dos dados.

•**Matriz de Confusão**: Mostra que a maioria das classificações erradas ocorre nas categorias "Neutro" e "Negativo".

In-Context Learning com GPT-Neo





Ferramentas e Bibliotecas:

- **Manipulação e Visualização de Dados:**
  - pandas, matplotlib, seaborn.
- **Pré-processamento de Texto:**
  - nltk, TextBlob.
- **Modelos de Machine Learning:**
  - scikit-learn (SVM, CountVectorizer, TfidfVectorizer).
- **Modelagem de Tópicos:**
  - LatentDirichletAllocation do scikit-learn.
- **Transformers e Deep Learning:**
  - transformers (BERT, DistilBERT, Roberta).
  - torch para construção e treinamento de modelos.
  - **GPT-Neo:** transformers.pipeline para geração de texto e classificação contextualizada.
- **Utilitários:**
  - numpy, wordcloud, collections (Counter)

Modelos Pré-treinados:

- **BERT:** bert-base-uncased, distilbert-base-uncased.
- **Roberta:** roberta-base.
- **GPT-Neo:** EleutherAI/gpt-neo-1.3B para *In-Context Learning*.