Novo Ensino
Suplementar

Análise de sentimentos

a partir de algoritmos de classificação multiclasse

Ruan Lima

Introdução

Através dos conhecimentos adquiridos durante o curso, implementaremos uma variedade de algoritmos de classificação multiclasse para identificar o modelo mais eficaz na análise de sentimentos em textos provenientes de redes sociais.

Dataset

- Linhas com informações retiradas de redes sociais.
- Texto, assunto e sentimento.
- Informações em inglês.
- Sentimentos: Positive, Negative, Neutral e Irrelevant.

Pontos positivos	Pontos Negativos
• Dados levemente balanceados • Fácil interpretação • Quantidade equilibrada de classes	• Muitas linhas • Linhas nulas

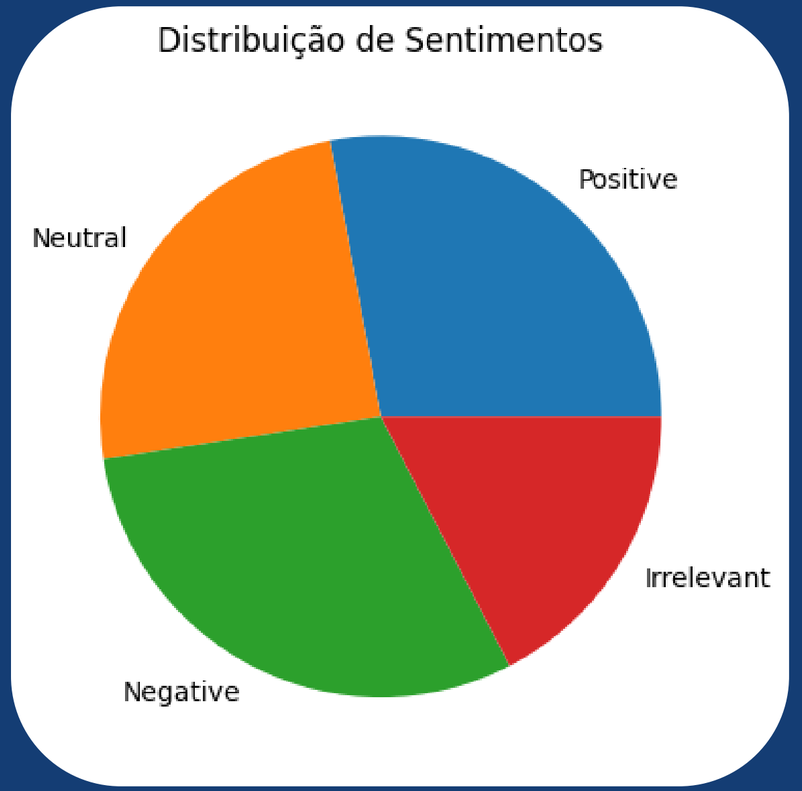
	none	none.1	Sentiment	Text
0	2401	Borderlands	Positive	im getting on borderlands and i will murder yo...
1	2401	Borderlands	Positive	I am coming to the borders and I will kill you...
2	2401	Borderlands	Positive	im getting on borderlands and i will kill you ...
3	2401	Borderlands	Positive	im coming on borderlands and i will murder you...
4	2401	Borderlands	Positive	im getting on borderlands 2 and i will murder ...
5	2401	Borderlands	Positive	im getting into borderlands and i can murder y...
6	2402	Borderlands	Positive	So I spent a few hours making something for fu...
7	2402	Borderlands	Positive	So I spent a couple of hours doing something f...

Análise exploratória

- **Remoção de linhas nulas**
- **Remoção de linhas com conteúdo diferente de *str***
- **Dados levemente balanceados**

Análise exploratória

- **Remoção de linhas nulas**
- **Remoção de linhas com conteúdo diferente de *str***
- **Dados levemente balanceados**



Positive: 20832

Neutral: 18318

Negative: 22542

Irrelevant: 12990

Pré-processamento

Como estamos trabalhando com textos, esta etapa é crucial.

Através dela, foram removidas toda e qualquer informação desnecessária para o nosso projeto (acentos, pontuações, caracteres especiais e padronização dos nossos textos para *lowercase*).

Além disso, foram feitos os processos de tokenização, remoção de stop words, stemização e lematização.

Metodologia

- **Algoritmos do *sklearn*:**

K-Nearest Neighbors (KNN), Árvore de decisão, Support Vector Machine (SVM), Naive Bayes e Random Forest.

- ***Vetorização*:**

TF-IDF (Term Frequency-Inverse Document Frequency) e Bag of Words (BoW).

- **Métrica:**

Acurácia.

Algoritmos

KNN
Algoritmo de classificação baseado em instâncias que classifica um dado ponto com base nos rótulos dos seus k-vizinhos mais próximos.

Árvore de decisão
Algoritmo que utiliza uma estrutura de árvore para tomar decisões baseadas nas características dos dados.

SVM
Algoritmo de classificação que encontra o melhor hiperplano que separa as diferentes classes no espaço de características.

Naive Bayes
Classificador probabilístico baseado no teorema de Bayes, que assume a independência condicional entre as características.

Random Forest
O Random Forest consiste em múltiplas árvores de decisão independentes que são treinadas com subconjuntos diferentes do dataset original, criados usando a técnica de bootstrap.

A combinação desses algoritmos com as técnicas de vetorização nos permitirá uma avaliação abrangente do melhor modelo para a tarefa.

A combinação desses algoritmos com as técnicas de vetorização nos permitirá uma avaliação abrangente do melhor modelo para a tarefa.

KNN + TF-IDF

Vetorização através do TF-IDF

```
[ ]  vectorizer = TfidfVectorizer()
     X_train_vec = vectorizer.fit_transform(X_train_lm)
     X_test_vec = vectorizer.transform(X_test_lm)
```

Implementação do KNN

```
[ ]  # Definição do classificador KNN
     knn = KNeighborsClassifier()

     # Treinamento do modelo
     knn.fit(X_train_vec, y_train)

     # Predição
     y_pred = knn.predict(X_test_vec)

     # Resultados
     knn_tfidf = (y_test, y_pred)
```

KNN + BoW

Vetorização através do BoW

```
[ ]  vectorizer = CountVectorizer()
     X_train_vec = vectorizer.fit_transform(X_train_lm)
     X_test_vec = vectorizer.transform(X_test_lm)
```

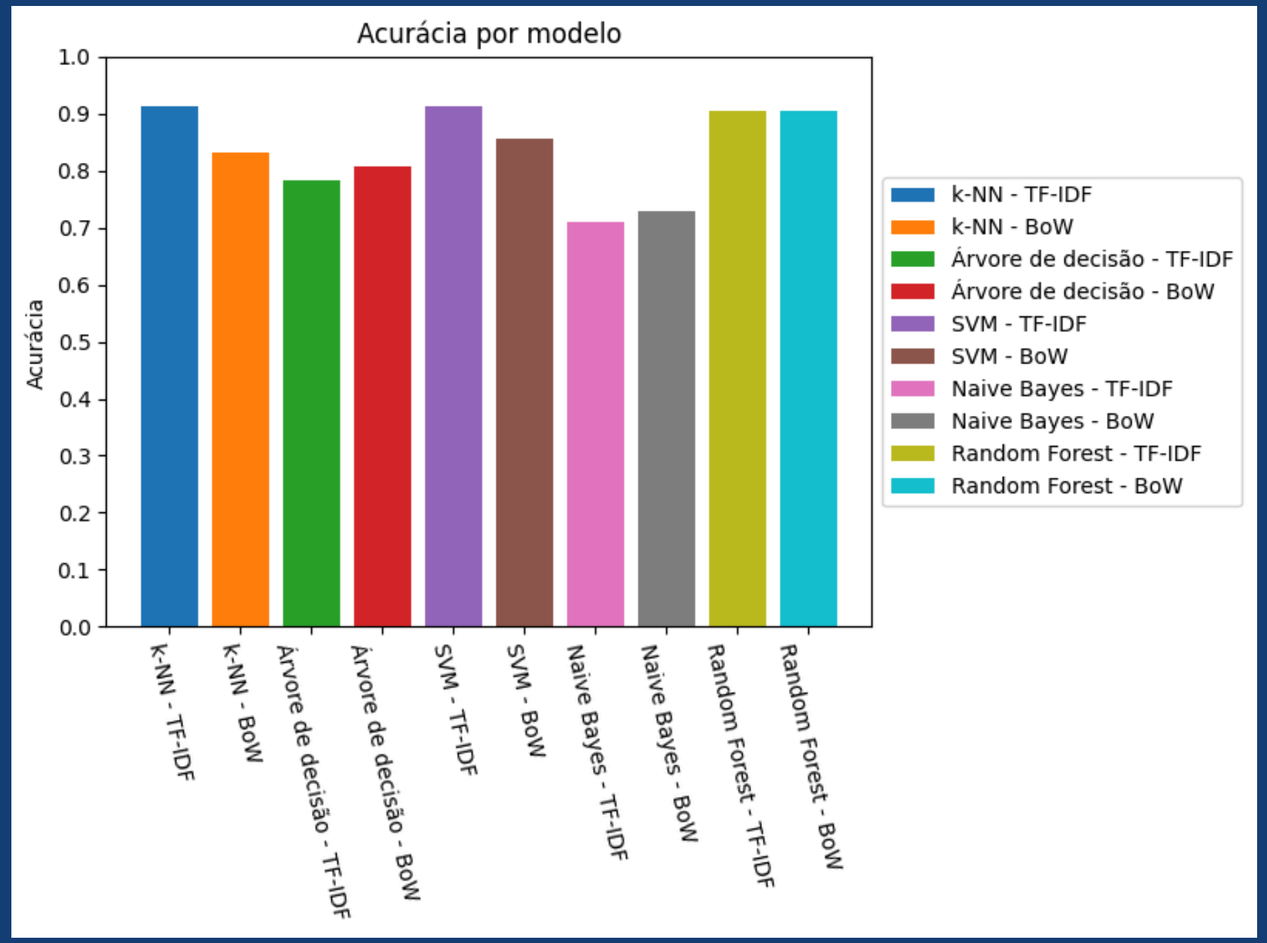
Implementação do KNN

```
[ ]  # Definição do classificador KNN
     knn = KNeighborsClassifier()

     # Treinamento do modelo
     knn.fit(X_train_vec, y_train)

     # Predição
     y_pred = knn.predict(X_test_vec)

     # Resultados
     knn_bow = (y_test, y_pred)
```

RESULTADOS

Acurácia por modelo

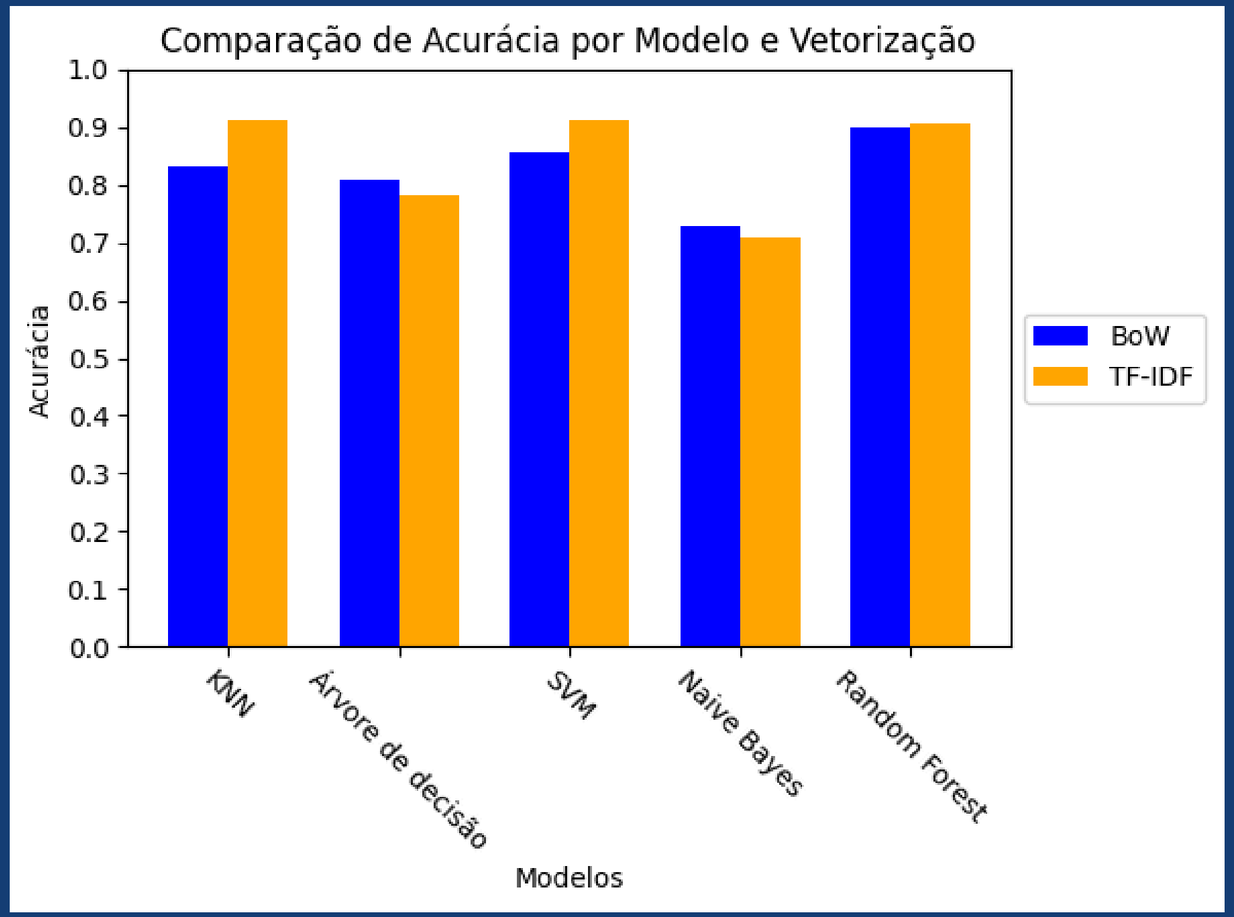
Modelo	Acurácia
KNN + TF-IDF	91%
KNN + BoW	83%
Árvore de decisão + TF-IDF	78%
Árvore de decisão + BoW	81%
SVM + TF-IDF	91%
SVM + BoW	86%
Naive Bayes + TF-IDF	71%
Naive Bayes + BoW	73%
Random Forest + TF-IDF	91%
Random Forest + BoW	90%



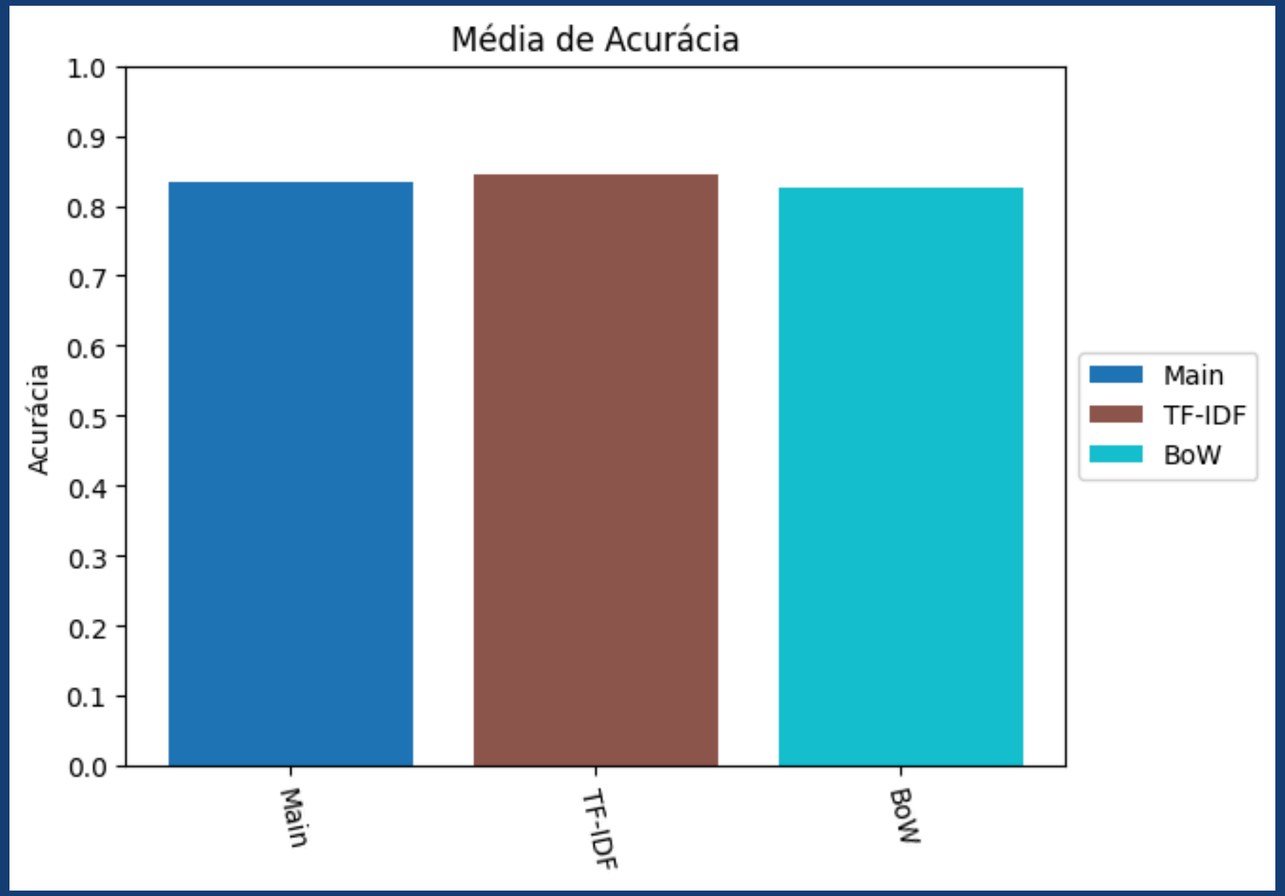
Modelo	Acurácia
KNN + TF-IDF	91%
KNN + BoW	83%
Árvore de decisão + TF-IDF	78%
Árvore de decisão + BoW	81%
SVM + TF-IDF	91%
SVM + BoW	86%
Naive Bayes + TF-IDF	71%
Naive Bayes + BoW	73%
Random Forest + TF-IDF	91%
Random Forest + BoW	90%

1 - KNN + TF-IDF
1 - Random Forest + TF-IDF
1 - SVM + TF-IDF
2 - Random Forest + BoW
3 - SVM + BoW
4 - KNN + BoW
5 - Árvore de decisão + BoW
6 - Árvore de decisão + TF-IDF
7 - Naive Bayes + Bow
8 - Naive Bayes + TF-IDF

Modelo	Acurácia
KNN + TF-IDF	91%
KNN + BoW	83%
Árvore de decisão + TF-IDF	78%
Árvore de decisão + BoW	81%
SVM + TF-IDF	91%
SVM + BoW	86%
Naive Bayes + TF-IDF	71%
Naive Bayes + BoW	73%
Random Forest + TF-IDF	91%
Random Forest + BoW	90%



- TF-IDF foi superior na maioria dos algoritmos
- Random Forest foi o algoritmo mais equilibrado
- KNN foi o modelo mais divergente
- Naive Bayes foi o algoritmo com menor eficiência



- Média acurácia geral: 83%
- Média acurácia TF-IDF: 84%
- Média acurácia BoW: 82%

Conclusão

- Com base nos resultados e gráficos previamente mostrados podemos concluir com propriedade que os melhores algoritmos foram o KNN, SVM e o Random Forest, sendo estes, os que obtiveram as melhores acurácias. O Naive Bayes se mostrou o mais ineficiente nesta tarefa. Bag of Words foi a vetorização mais ineficiente.

Conclusão

- Através da implementação e avaliação de diversos algoritmos de classificação multiclasse, este projeto identificou o Random Forest como o modelo mais eficaz para a tarefa de análise de sentimentos em textos de redes sociais.

Conclusão

- Além disso, concluímos que a vetorização TF-IDF é a mais eficiente, contribuindo significativamente para a precisão das previsões de sentimentos.

Novo Ensino
Suplementar

OBRIGADO!