Lista 8

Lucas Gualtieri Firace Evangelista

Questão 1

1. **Carregamento dos Dados:**
    - Os dados foram carregados a partir do arquivo CSV `Tweets_Mg.csv`.
    - Foram selecionadas as colunas `Text` e `Classificacao` e removidas as linhas com valores nulos.
2. **Pré-processamento:**
    - Os textos foram convertidos para minúsculas.
    - Remoção de caracteres especiais e pontuações.
3. **Divisão dos Dados:**
    - Os dados foram divididos em conjuntos de treino (80%) e teste (20%).
4. **Vetorização:**
    - Utilizei o `CountVectorizer` para transformar os textos em vetores de contagem de palavras.
5. **Treinamento do Modelo:**
    - Um modelo de Naive Bayes Multinomial foi treinado com os dados vetorizados.
6. **Avaliação do Modelo:**
    - O modelo foi avaliado utilizando a métrica de acurácia.

Por que o Naive Bayes é adequado para mineração de texto?

- **Eficiência:** É simples, rápido e funciona bem para grandes conjuntos de dados textuais.
- **Efetividade:** Apesar da suposição de independência ser "irrealista", o modelo se comporta bem em tarefas de classificação de texto, como:
  - Detecção de spam em e-mails.
  - Classificação de sentimentos (positivo/negativo).
  - Classificação de tópicos.
- **Robustez:** Funciona bem mesmo com conjuntos de dados pequenos e ruidosos.

Questão 2

Questões de ENADE e POSCOMP

1. **Resposta Correta:** A) n é falso. Logo, $n \rightarrow r$ é verdadeira independentemente do valor de r.
2. **Resposta final:** D) Em toda sorveteria, há sempre algum sorvete que não é doce ou que contém adoçante.
3. Pelas premissas ditas, a resposta correta é: C) Algum aluno que é estagiário não está no último período.

Código no link: Google Colab

```
[1]  import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.feature_extraction.text import CountVectorizer
     from sklearn.naive_bayes import MultinomialNB
     from sklearn.metrics import classification_report, accuracy_score

     data = pd.read_csv('Tweets_Mg.csv')
```

```
[2]  data = data[['Text', 'Classificacao']].dropna()
     data['Text'] = data['Text'].str.replace('[^\w\s]', '').str.lower()
```

```
[3]  X_train, X_test, y_train, y_test = train_test_split(data['Text'], data['Classificacao'], test_size=0.2, random_state=42)

     vectorizer = CountVectorizer()
     X_train_vec = vectorizer.fit_transform(X_train)
     X_test_vec = vectorizer.transform(X_test)
```

```
model = MultinomialNB()
model.fit(X_train_vec, y_train)
```

```
[6]  # Previsões
     y_pred = model.predict(X_test_vec)

     # Relatório de desempenho
     print("Accuracy:", accuracy_score(y_test, y_pred))
     print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9530487804878048
              precision    recall  f1-score   support

    Negativo       0.96      0.96      0.96       476
      Neutro       0.93      0.92      0.93       503
    Positivo       0.96      0.97      0.97       661

    accuracy                           0.95      1640
   macro avg       0.95      0.95      0.95      1640
weighted avg       0.95      0.95      0.95      1640
```