Apresentação Oral

Bom dia a todos!

Hoje vou apresentar os resultados de um estudo acadêmico focado na detecção de Fake News, utilizando técnicas e algoritmos de Machine Learning.

—-

1. **Introdução**

Fake News são informações falsas, geralmente divulgadas em redes sociais, causando impactos sérios. Como é fácil espalhar notícias falsas, precisamos de métodos que ajudem a identificar o que é verdadeiro ou não.

—-

2. **Objetivo**

O objetivo deste trabalho foi analisar diferentes algoritmos para detectar automaticamente Fake News, buscando o modelo mais eficiente.

—-

3. **Metodologia**

Coletamos várias notícias (falsas e verdadeiras) do site Kaggle.
Os dados foram limpos, padronizados e embaralhados. Fizemos uma análise visual com WordCloud e dividimos em conjuntos de treino e teste.

—-

4. **Algoritmos Testados**

Utilizamos: Regressão Logística, Árvore de Decisão, Floresta Aleatória, SVM e KNN.
Cada modelo “aprende” a identificar notícias falsas.

—-

5. **Resultados**

Quatro modelos apresentaram acurácia acima de 90%. A Árvore de Decisão foi o melhor.
O KNN cometeu mais erros.
Analisamos acurácia, precisão, sensibilidade e especificidade.

—-

6. **Limitações e Perspectivas Futuras**

Houve dificuldade em encontrar bases de dados em português.
É necessário testar mais algoritmos, expandir os conjuntos de dados e aprimorar a validação.

—-

7. **Conclusão**

O Machine Learning é uma ferramenta poderosa para detectar Fake News e proteger a sociedade!

—-

8. **Código**

<u>Este código instala as bibliotecas que iremos usar. Elas funcionam como “caixas de ferramentas” para programar.</u>

```python

```
!pip install wordcloud seaborn nltk
```

Importamos as ferramentas instaladas, assim podemos usar seus “superpoderes” no código.

```
import pandas as pd
import numpy as np
import string
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report,
confusion_matrix
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
```

```

——

Aqui carregamos as notícias falsas e verdadeiras que vamos estudar. Imagine dois livros diferentes!

```python
fake = pd.read_csv('Fake.csv')   # Fazer upload de 'Fake.csv' no Colab
true = pd.read_csv('True.csv')   # Fazer upload de 'True.csv' no Colab
fake['target'] = 1  # Fake News
true['target'] = 0  # True News
```

—

Juntamos tudo em uma grande tabela e embaralhamos como um baralho! Removemos título e data para focar apenas no texto.

```python
data = pd.concat([fake, true], ignore_index=True)
data = data.sample(frac=1).reset_index(drop=True)
data.drop(['title', 'date'], axis=1, inplace=True)
```

—

Simplificamos as palavras: apenas minúsculas, sem pontuação ou palavras que não ajudam, como “a”, “de”, “o”.

```python
stop_words = set(stopwords.words('portuguese'))
def clean_text(text):
    text = str(text).lower()
    text = text.translate(str.maketrans('', '', string.punctuation))
    text = " ".join([word for word in text.split() if word not in stop_words])
    return text

data['text'] = data[‘text'].apply(clean_text)
```

Veja um desenho bonito com as palavras que mais aparecem nas notícias — a nuvem de palavras.

```python

```
wc = WordCloud(width=800, height=400,
background_color='white').generate(' '.join(data['text']))
plt.figure(figsize=(10, 5))
plt.imshow(wc, interpolation='bilinear')
plt.axis('off')
plt.show()
```

—

Transformamos os textos em números que o computador entende e dividimos em dados de treino e teste, para que ele aprenda e pratique.

```python

```
X = data['text']
y = data['target']
vectorizer = TfidfVectorizer(max_features=5000)
X_vect = vectorizer.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_vect,
y, test_size=0.2, random_state=42)
```

——

Agora vem a mágica! Vamos treinar cinco máquinas inteligentes para dizer se uma notícia é falsa ou verdadeira.

```python
modelos = {
    'Regressão Logística': LogisticRegression(),
    'Árvore de Decisão': DecisionTreeClassifier(),
    'Floresta Aleatória': RandomForestClassifier(),
    'SVM': SVC(),
    'KNN': KNeighborsClassifier()
}
for nome, modelo in modelos.items():
    print(f'\n--- {nome} ---')
    modelo.fit(X_train, y_train)
```

```
    y_pred = modelo.predict(X_test)
    print('Matriz de Confusão:')
    print(confusion_matrix(y_test, y_pred))
    sns.heatmap(confusion_matrix(y_test, y_pred), annot=True,
fmt='d', cmap='Blues')
    plt.title(f'Matriz de Confusão - {nome}')
    plt.xlabel('Previsto')
    plt.ylabel('Real')
    plt.show()
    print('Relatório de Classificação:')
    print(classification_report(y_test, y_pred))
```

—-

Se quiser salvar uma máquina treinada para usar depois, é só salvar!

```python

```
# from joblib import dump, load
# dump(modelos['Árvore de Decisão'],
‘decision_tree_model.joblib')
```

Conclusão

O Machine Learning mostrou-se promissor na detecção de Fake News. Com mais pesquisas, podemos construir um mundo mais seguro para todos!