SELLAPPANMURALI /
fack_news_phase4

Code Issues Pull requests Projects Wiki Security Insights Se

fack_news_phase4 / main.py

SELLAPPANMURALI Update main.py 20 minutes ago

97 lines (73 loc) · 7.31 KB

fack_news_phase4 / main.py Top

Code Blame Raw

```
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load


import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files unde

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:09.114058Z","iopub.execute_in
import pandas as pd
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:10.243502Z","iopub.execute_in
fake = pd.read_csv('/kaggle/input/fake-and-real-news-dataset/Fake.csv')
```

```
true = pd.read_csv('/kaggle/input/fake-and-real-news-dataset/True.csv')

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.787408Z","iopub.execute_inp
fake['Category'] = 'fake'
fake

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.815088Z","iopub.execute_inp
true['Category'] = 'true'
true

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.832993Z","iopub.execute_inp
```

```
data

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.859531Z","iopub.execute_inp
data['Category'].value_counts()

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.879444Z","iopub.execute_inp
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['Category'] = le.fit_transform(data['Category'])
data['date'] = le.fit_transform(data['date'])
data['subject'] = le.fit_transform(data['subject'])

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.940237Z","iopub.execute_inp
data['Category']


# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.949272Z","iopub.execute_inp
data['date']

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.961812Z","iopub.execute_inp
data['subject'].value_counts()

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.973990Z","iopub.execute_inp
data['title'].shape

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:12.984114Z","iopub.execute_inp
vectorizer = TfidfVectorizer()
title = vectorizer.fit_transform(data['title'])
text = vectorizer.transform(data['text'])


# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:32.117764Z","iopub.execute_inp
 title

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:32.124993Z","iopub.execute_inp
from sklearn.model_selection import train_test_split
X = title
```

```python
y = data['Category']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:25:32.147343Z","iopub.execute_inp
model = SVC()
model.fit(X_train, y_train)

# %% [code] {"execution":{"iopub.status.busy":"2023-10-27T04:29:37.364346Z","iopub.execute_inp
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:")
print(classification_report(y_test, y_pred))

# %% [code]
```