Fake News Detection using support vector machine

Aman Aram Aman

# 1.introduction for the project

In today's digital world, where false information may spread quickly, identifying fake news is an essential duty. The goal of this project is to create a machine learning model that uses a Support Vector Machine (SVM) classifier to identify fake news.

# 2.Project Components and Code Python code explanation

1. Importing the necessary libraries for the project

**import pandas as pd**

**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**

**these are all the important libraries to load for the topic**

**2.explanation for the libraries**

* **pandas (pd): For data manipulation and analysis.**
* **train\_test\_split: To split the dataset into training and testing sets.**
* **TfidfVectorizer: To convert text data into TF-IDF (Term Frequency-Inverse Document Frequency) features.**

**SVC (Support Vector Classifier): Support Vector Machine classifier from scikit-learn.**

* **accuracy\_score and classification\_report: Evaluation metrics.**

# 3.Loading the dataset

**dataset = pd.read\_csv('news.csv')**

**print("Dataset Info:")**

**print(dataset.info())**

**For my dataset I did save the python file and the dataset named news in the same file to load it directly .**

**1.This codes reads the dataset from the directory**

**2.print the basic info about the dataset using info()**

# 4.Split the dataset

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(dataset['text'], dataset['label'], test\_size=0.2, random\_state=42)**

**1.Split the dataset into training and testing sets using train\_test\_split**

**2.as it is shown 20% of the dataset is used for testing and 80% Is used for training.**

# 5.Convert text to TF-IDF futures

TF-IDF is a numerical statistics that reflects how important a word is to a document

**tfidf\_vectorizer = TfidfVectorizer(max\_features=5000**

**X\_train\_tfidf = tfidf\_vectorizer.fit\_transform(X\_train)**

**X\_test\_tfidf = tfidf\_vectorizer.transform(X\_test)**

1. **Converts the text data into TF-IDF features using TfidfVectorizer**
2. **Limits the number of features to the top 5000 most important words**

**limiting the TF-IDF features to 5000 helps manage computational complexity while capturing the most important data for the fake news detection task. The TF-IDF features indicate the value of words in the news articles.**

# 5.Train SVM Model

**svm\_model = SVC(kernel='linear')**

**svm\_model.fit(X\_train\_tfidf, y\_train)**

1. **Initializes an SVM model with a linear kernel**
2. **Fits the model on the TF-IDF features of the training set**

# 6.Make Predictions

y\_pred = svm\_model.predict(X\_test\_tfidf)

1.this code uses the trained model to make predictions on the test set

# 7.Evaluate the Model

**accuracy = accuracy\_score(y\_test, y\_pred)**

**classification\_rep = classification\_report(y\_test, y\_pred)**

Calculates accuracy and generates a classification report using scikit-learn's accuracy\_score and classification\_report

1. **accuracy\_score:**

**Calculates the accuracy of the model by comparing the predicted labels (y\_pred) with the true labels (y\_test).**

**Accuracy is the ratio of correctly predicted instances to the total instances.**

**2.** **classification\_report:**

**Generates a detailed classification report that includes precision, recall, and F1-score for each class.**

**Provides additional information such as support, which is the number of actual occurrences of each class in the specified dataset.**

# 8.Print The Results

**print(f"Accuracy: {accuracy:.2f}")**

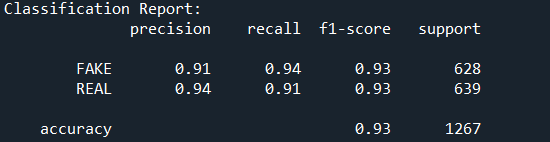
**This line prints the accuracy of the model on the test set, accuracy is a decimal number between 1 and 0**

**print("Classification Report:")**

**prints a detailed classification report that includes precision, recall, and F1-score for each class (real and fake news).**

**print(classification\_rep)**

**The variable holding the classification report generated using classification\_report**

****

**1.Precision shows the among instances predicted as real are 94% are actually Real**

**and the among instances predicted as fake are 91% are actually fake.**

**2.Recall shows 91% actual instances were correctly predicted for real class**

**And 94% actual actual instances were correctly predicted for fake class.**

**3.support For the "Real" class, there are 639 instances in the dataset.**

**For the "Fake" class, there are 628 instances in the dataset**

**4.Accuracy the overall accuracy is 0.93 or 93%, indicating that 93% of instances in the test set were correctly classified**