
CAPSTONE PROJECT

FAKE NEWS

Presented By:

- 1. Student Name- JOHN DANGMEI**
- 2. College Name- MADHA ENGINEERING COLLEGE**
- 3. Department- CIVIL ENGINEERING**

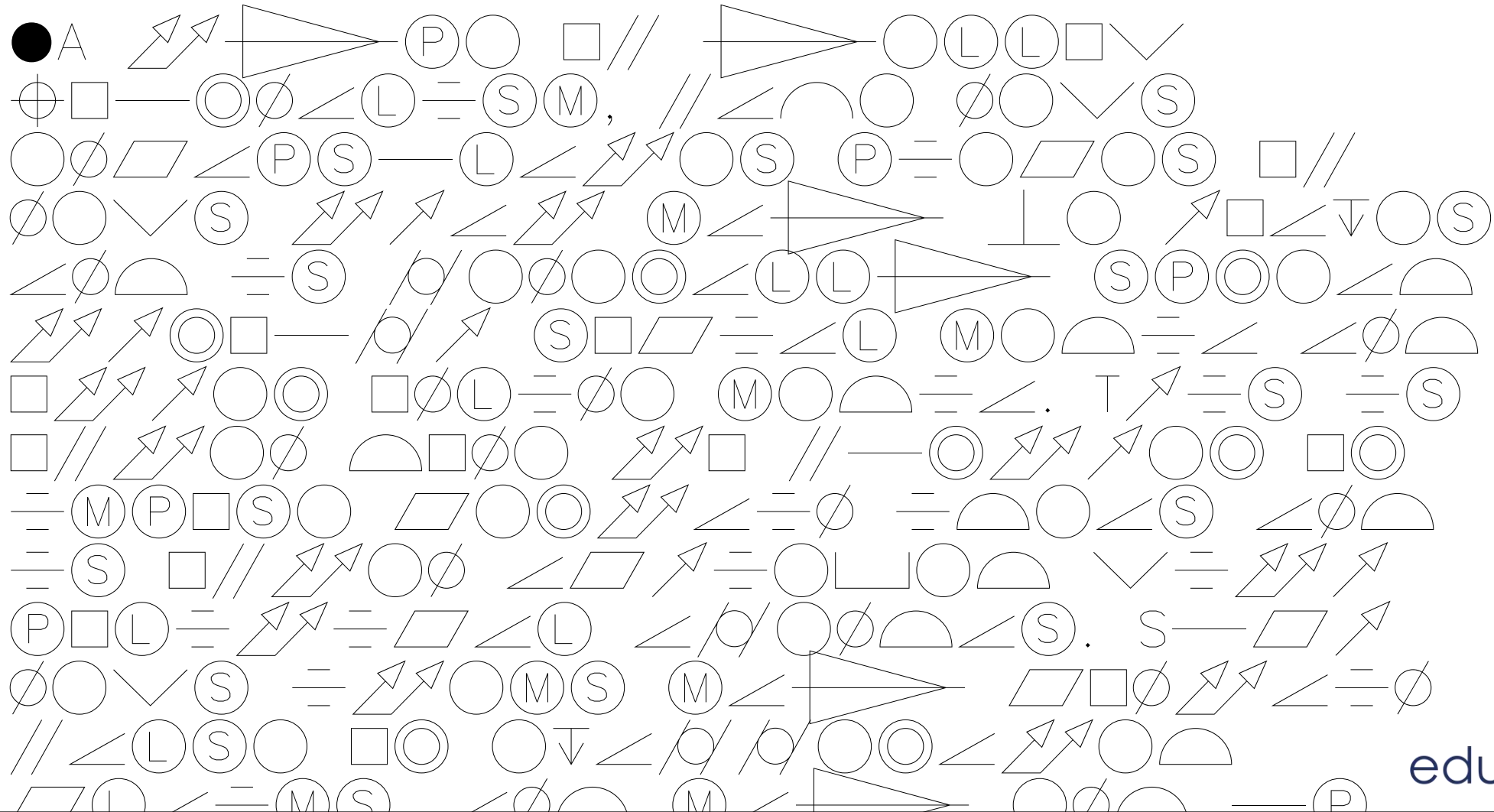
OUTLINE

1. • INTRODUCTION
2. • WHAT IS FAKE NEWS..ST
3. • FAKE NEWS CHARACTERIZATION FAKE NEWS
DETECTION
4. • WHAT IS TFIDFVECTORIZER
5. • EXAMPLE
6. • CONCLUSION

INTRODUCTION

- FAKE NEWS SPREADS LIKE A WILDLIFE AND THIS IS A BIG ISSUE IN THIS ERA.
- FOR SOME YEARS, MOSTLY SINCE THE RISE OF SOCIAL MEDIA, FAKE NEWS HAVE BECOME A SOCIETY PROBLEM, IN SOME OCCASION SPREADING MORE AND FASTER THAN THE TRUE INFORMATION, IN THIS PAPER I EVALUATE THE PERFORMANCE OF ATTENTION MECHANISM FOR FAKE NEWS DETECTION ON TWO DATASETS, ONE CONTAINING TRADITIONAL ONLINE NEWS ARTICLES AND THE SECOND ONE NEWS FROM VARIOUS SOURCES.

WHAT IS FAKE NEWS..ST



FAKE NEWS CHARACTERIZATION

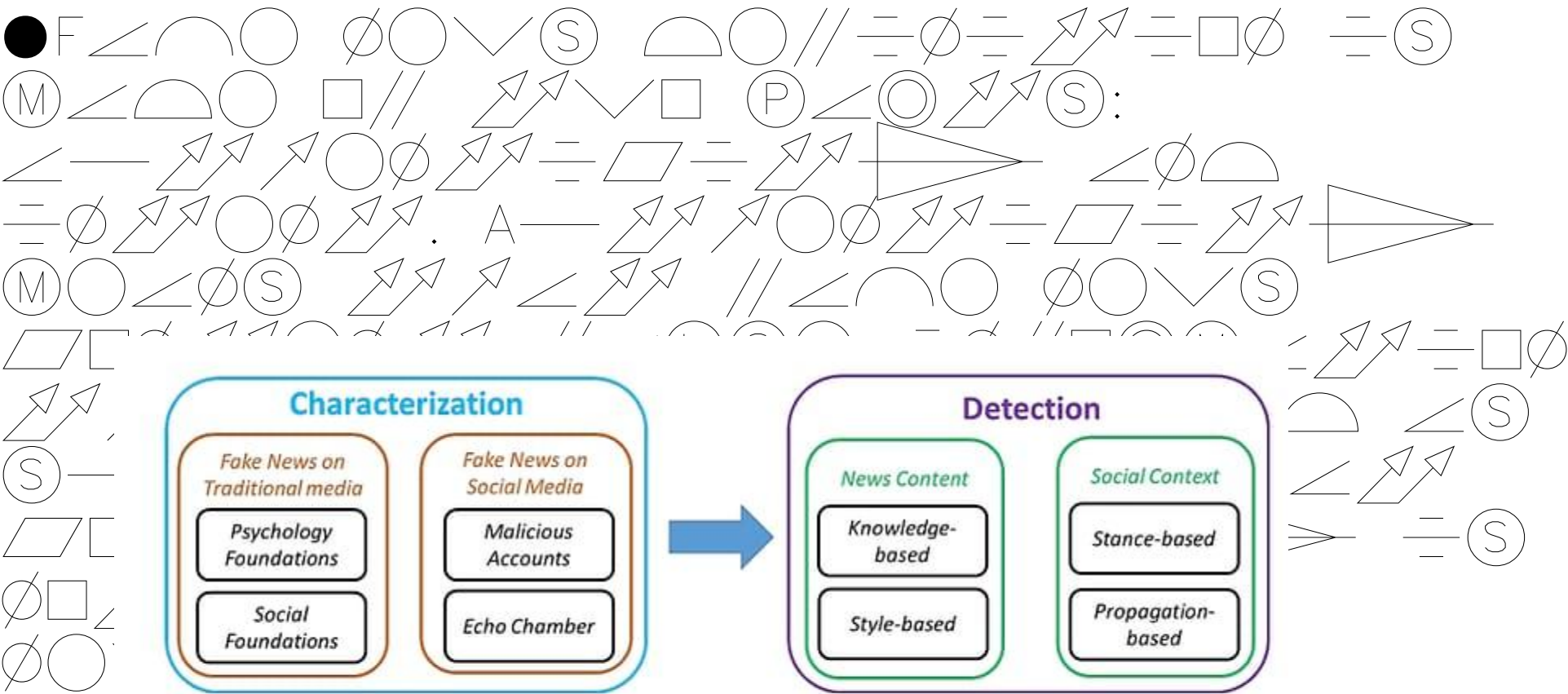


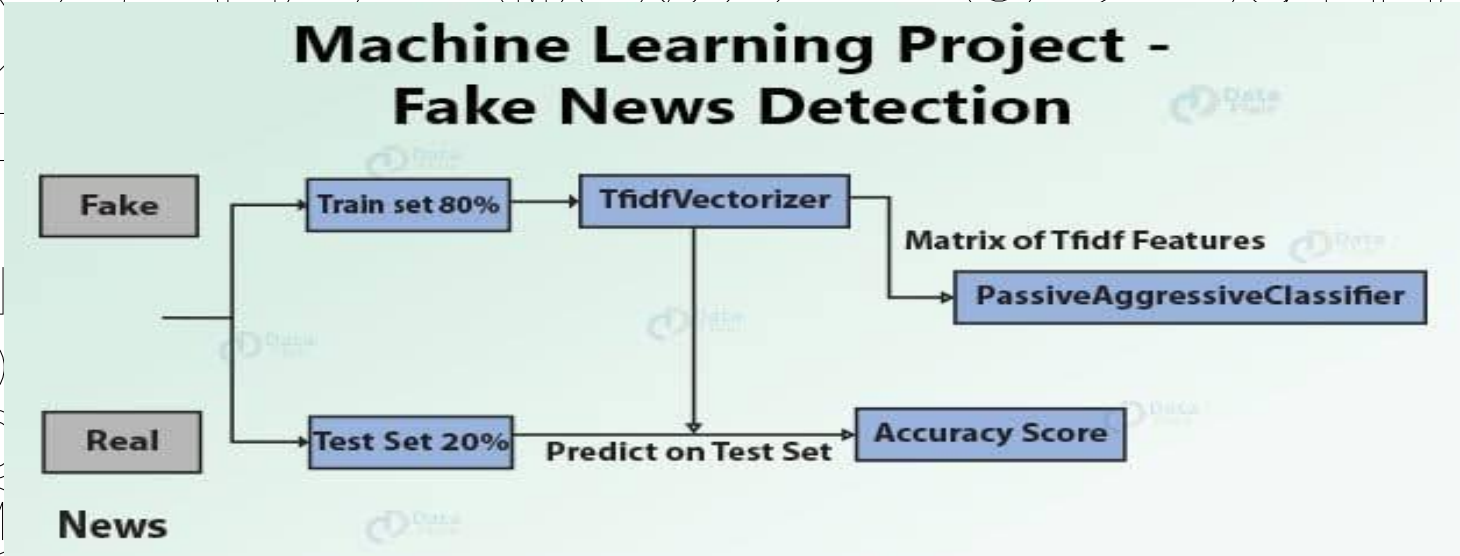
Figure 1: Fake news on social media: from characterization to detection.

WHAT IS TFIDFVECTORIZER

● TF (Term Frequency) is a measure of how many times a word appears in a document. It is calculated as the number of occurrences of a word in a document divided by the total number of words in the document. For example, if the word "cat" appears 5 times in a document of 100 words, the TF for "cat" is 0.05.

● IDF (Inverse Document Frequency) is a measure of how rare a word is across a collection of documents. It is calculated as the inverse of the logarithm of the number of documents containing the word. For example, if the word "cat" appears in 10 out of 1000 documents, the IDF for "cat" is $\frac{1}{\log(10)}$.

TFIDF (Term Frequency-Inverse Document Frequency) is a combined measure of a word's importance in a document and its rarity across the collection. It is calculated as the product of TF and IDF. For example, if the word "cat" has a TF of 0.05 and an IDF of $\frac{1}{\log(10)}$, the TFIDF for "cat" is $0.05 \times \frac{1}{\log(10)}$.



EXAMPLE

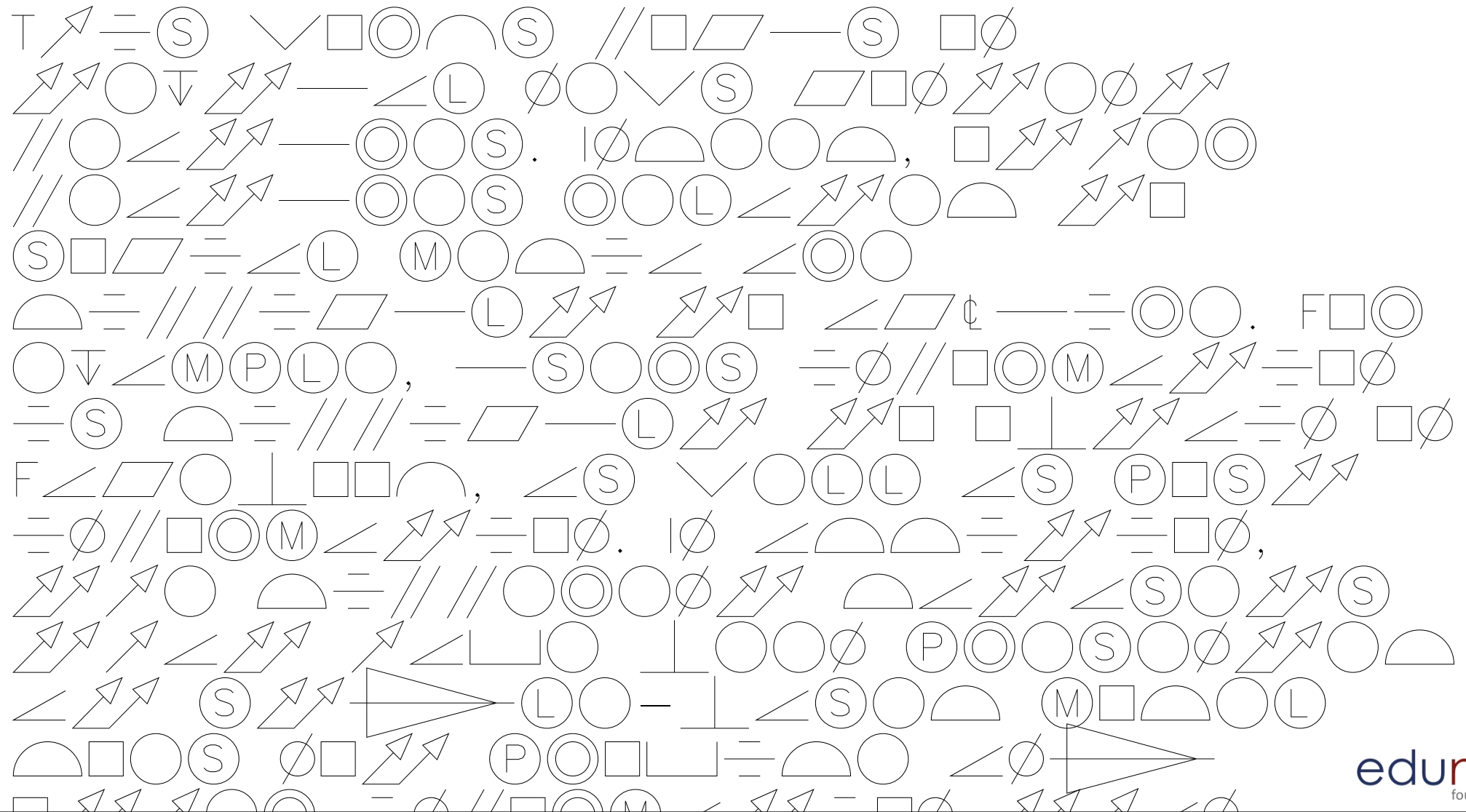
F=∅∠(L)L▷ ∠ INDIAN (S)↗—◐∅↗ //⊙□(M)
PONDICHERRY —∅=└┐○⊙(S)=↗▷, ∅∠(M)○◐ RAMU
//□—∅◐ ∠ ↗□(M)○ ⊙⊙(M)○◐ ▷ ◐—⊙⊙ //□⊙
C□└=◐-19 ✓↗=◐↗ = (S) //□⊙ ↗↗○
└○⊙▷ // = ⊙(S)↗↗ = (M)○ ∠ ◐◐○(P)↗○◐
└▷ WHO.

HO (P)⊙□└○◐ ↗↗ ∠↗
↗∠└(L)○(S)(P)□□∅ □//└
(P)□✓◐⊙⊙ ↗↗□ 2 ↗∠.

↗□∅○▷ ∠∅◐ (S)□(M)◐
//□⊙ ◐□∅(S)○◐—↗=└
(S)—(P)(P)⊙⊙(S)(S) ↗↗○ ○////○◐↗ (S) □// ◐□⊙□∅∠



CONCLUSION



↔ | (M) (P) □ ⊙ ↗ ↗ ∅ ○ ▯ ○ (S) (S) ∠ ⊙ →

(L) ≡ ⊥ ⊙ ∠ ⊙ ≡ ○ (S)

≡ (M) (P) □ ⊙ ↗ ↗ (P) ∠ ∅ ⊔ ∠ (S) ∠ (S) (P) ⊔

// ⊙ □ (M)

(S) ∩ (L) ○ ∠ ⊙ ∅. // ○ ∠ ↗ ↗ — ⊙ ⊙ _ ○ ↓ ↗ ↗ ⊙ ∠ ▯ ↗ ↗

≡ □ ∅. ↗ ↗ ○ ↓ ↗ ↗ ≡ (M) (P) □ ⊙ ↗ ↗

T // ≡ ⊔ // ∨ ○ ▯ ↗ ↗ □ ⊙ ≡ ▴ ○ ⊙

// ⊙ □ (M)

(S) ∩ (L) ○ ∠ ⊙ ∅. (M) □ ⊔ ○ (L) _ (S) ○ (L) ○ ▯ ↗ ↗ ≡ □ ∅

≡ (M) (P) □ ⊙ ↗ ↗ ↗ ↗ ⊙ ∠ ≡ ∅ _ ↗ ↗ ○ (S) ↗ ↗ _ (S) (P) (L) ≡ ↗ ↗

// ⊙ □ (M) (S) ∩ (L) ○ ∠ ⊙ ∅. (L) ≡ ∅ ○ ∠ ⊙ _ (M) □ ⊔ ○ (L)

≡ (M) (P) □ ⊙ ↗ ↗

P ∠ (S) (S) ≡ ⊥ ○ A / q / q ⊙ ⊙ (S) (S) ≡ ⊥ ○ C (L) ∠ (S) (S) ≡ //

≡ ○ ⊙

// ⊙ □ (M) (S) ∩ (L) ○ ∠ ⊙ ∅ (M) □ ↗ ↗ ⊙ _ ▯ (S)

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    # Initialize TfidfVectorizer
    tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_df=0.7)
    # Fit and transform the training data
    tfidf_train = tfidf_vectorizer.fit_transform(X_train)
    # Transform the testing data
    tfidf_test = tfidf_vectorizer.transform(X_test)
    # Initialize PassiveAggressiveClassifier
    pac = PassiveAggressiveClassifier(max_iter=50)
    # Train the PassiveAggressiveClassifier
    pac.fit(tfidf_train, y_train)
    # Predict on the testing data
    y_pred = pac.predict(tfidf_test)
    # Calculate the accuracy
    accuracy = accuracy_score(y_test, y_pred)
    print(f'Accuracy: {accuracy}')
```

```
# Example usage
# Replace 'fake_news_article.txt' with the path to your fake news article
fake_news_article = ["Your fake news article here"]
fake_news_article_tfidf = tfidf_vectorizer.transform(fake_news_article)
prediction = pac.predict(fake_news_article_tfidf)
print(f'Prediction: {prediction}')
```



THANK YOU