Using ML models in Web Applications

Practise Assignment

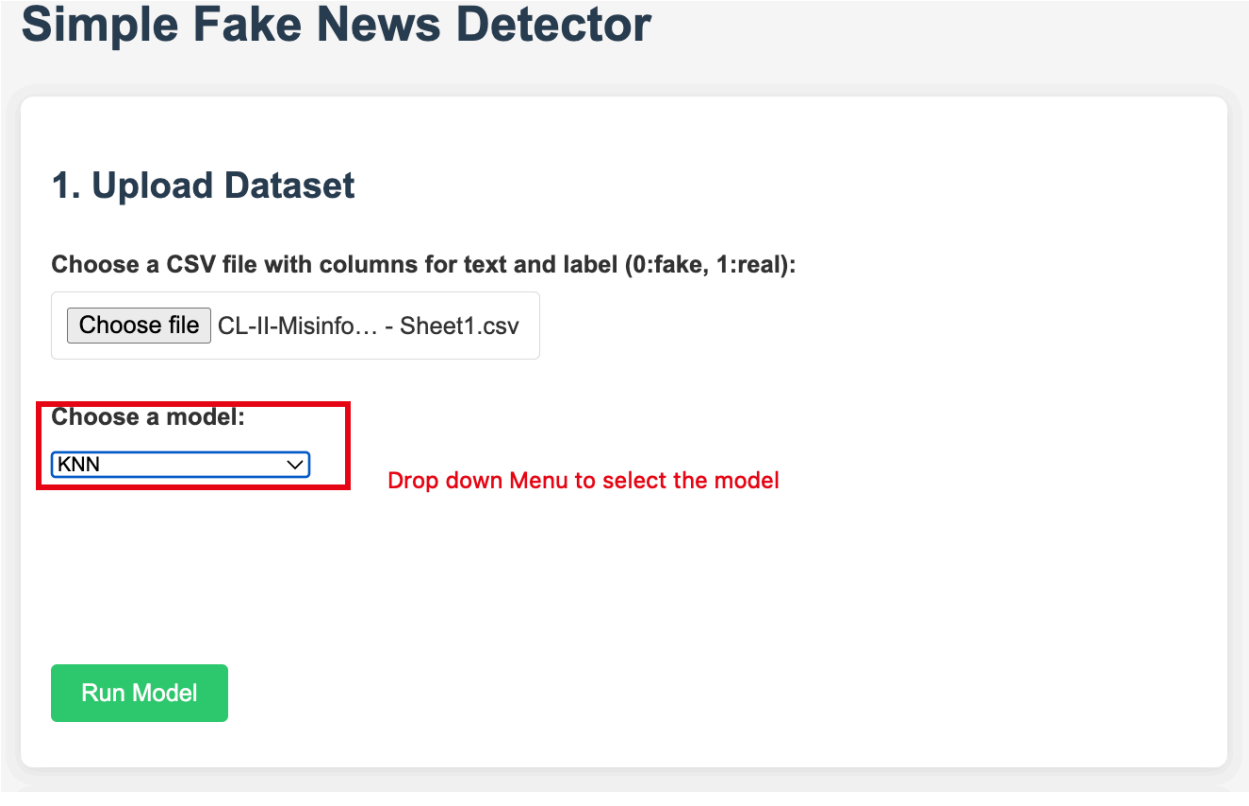
March 12, 2025

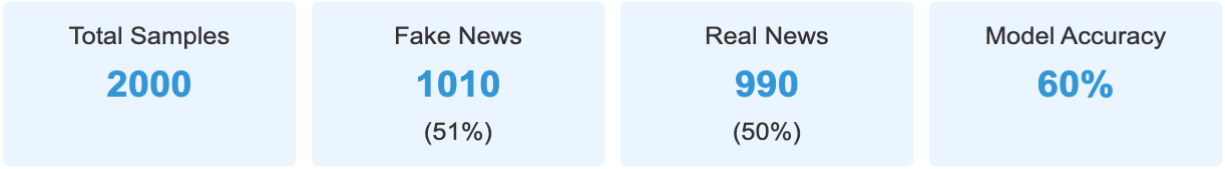
==========Instructions==========

1. *You are allowed to use only Node.js with any standard packages.*
2. *Submit your entire work in a zip by 17th March [this is entirely for your practice]*
3. *Experiment with different approaches to enhance learning.*

==========

PART 1: Convert the Python-based Scikit-Learn (that you have done last week) assignment into a pure JavaScript-based application using ml.js. Everything will run in a single web page.





[Note: This is just a demo. Ignore the numbers here and design choice is completely yours]

1. File Upload & Preprocessing
    - Users will upload a CSV file containing social media posts and labels (0: fake, 1: real). [You can extend your program to other kinds of files like .txt, .tsv, etc.
    - The data will be preprocessed (as you feel necessary).
    - The dataset will be split into 70% train, 10% validation, and 20% testing
2. Vectorisation (TF-IDF)
    - Convert text to TF-IDF vectors. [If you don't find any library that does it for you, write your own code. Resource, or you can also use any other embeddings.]
3. Model Training (Browser-Based ML Models)
    - Implement 6 models using ml.js atleast:
        1. K-Nearest Neighbors (KNN)
        2. Logistic Regression
        3. Random Forest classifier
        4. K-Means Clustering
        5. Neural Networks
        6. Decision Tree
4. Model Selection & Evaluation
    - Users will select a model from a dropdown.[make your design implementation]
    - The app will display the accuracy, confusion matrix, other details [design choice]
5. Prediction Interface [optional]
    - Users can input a text manually and select a model.
    - The selected model will predict if it's real or fake.

PART 2: In this assignment, you will use transformers.js to perform the following. You will create a web app where the user will upload again a test dataset, and the following will be performed sequentially. Display all the outcomes in a well-designed webpage [design choice]

- Evaluate sentiment classification accuracy on test data
- Translate tweets to Hindi
- Perform zero-shot classification on the translated tweets

Subpart 1: Sentiment Classification on Test Data

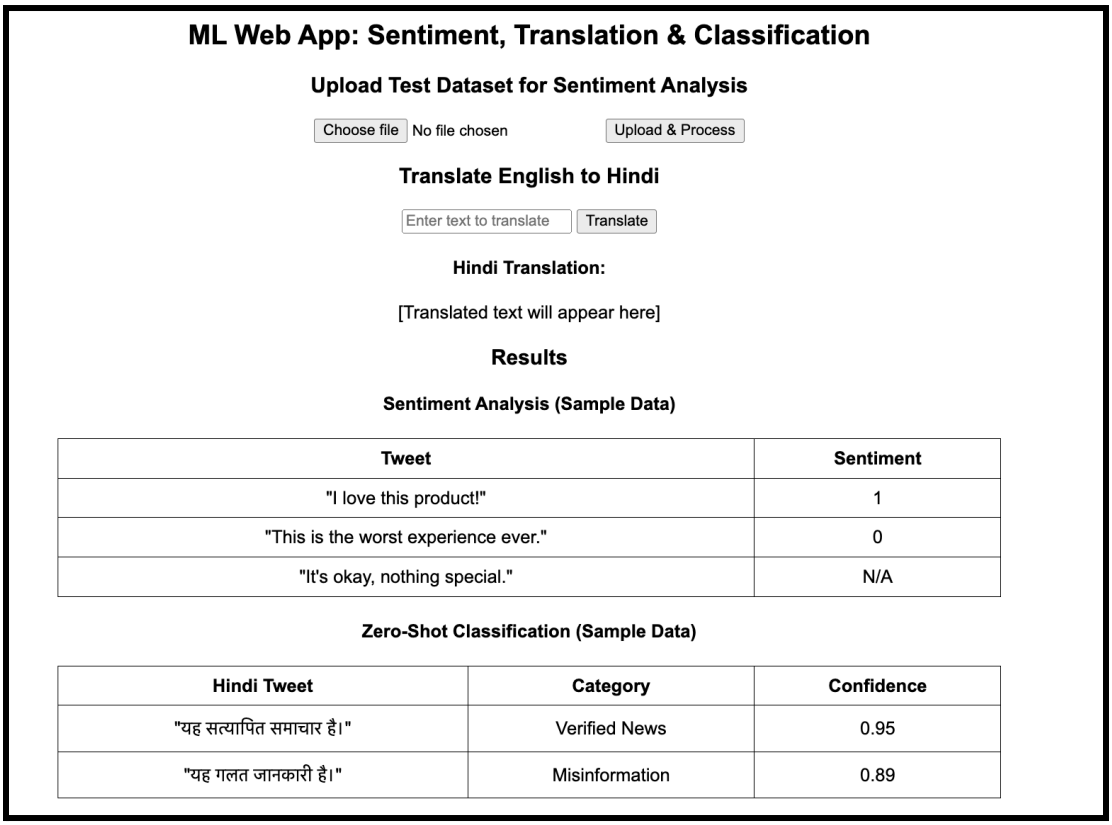
- Take input from the user of any dataset (say dataset.csv).
- Use the Xenova/bert-base-multilingual-uncased-sentiment model.
- Convert sentiment predictions to 3 classes:
    - positive → 1, negative → 0, neutral → N/A (Ignore in accuracy calculation)
- Calculate accuracy, precision, recall, and F1-score.

Subpart 2: Translate Tweets to Hindi

- Use Xenova/nllb-200-distilled-600M for English → Hindi translation.
- Store original and translated tweets.

Subpart 3: Zero-Shot Classification on Hindi Tweets

- Use Xenova/nli-deberta-v3-xsmall for zero-shot classification.
- Classify Hindi sentences into:
    - "misinformation"
    - "verified news"
- Display confidence scores for each label.



ML Web App: Sentiment, Translation & Classification

Upload Test Dataset for Sentiment Analysis

Choose file | No file chosen | Upload & Process

Translate English to Hindi

Enter text to translate | Translate

Hindi Translation:

[Translated text will appear here]

Results

Sentiment Analysis (Sample Data)

Tweet	Sentiment
"I love this product!"	1
"This is the worst experience ever."	0
"It's okay, nothing special."	N/A

Zero-Shot Classification (Sample Data)

Hindi Tweet	Category	Confidence
"यह सत्यापित समाचार है।"	Verified News	0.95
"यह गलत जानकारी है।"	Misinformation	0.89

[Note: This is just a demo. Ignore the numbers and sentences here and design choice is completely yours]

PART 3: In this assignment, you will use d3.js to represent the following:

Subpart 1: Dynamic Pie Chart [Resource]

You are required to create a dynamic pie chart using D3.js to visually represent the results of sentiment classification (Part 2, Subpart 1) and zero-shot classification (Part 2, Subpart 3). The chart should be displayed directly in the browser in real-time, allowing users to seamlessly

switch between the two datasets using a toggle button with a smooth transition effect. Each category in the pie chart must be represented by a distinct color to ensure clear data differentiation. The visualisation should be interactive and responsive, providing an intuitive and professional comparison of sentiment and classification results.

Subpart Z: Self Practice [Resource]

Explore various types of plots, practice their implementation, and integrate them effectively into your code. Experimenting with different visual representations will enhance data interpretation and provide deeper insights. Strive to incorporate the most suitable plots for your specific use case, ensuring clarity, accuracy, and professionalism in your visualisations.