Project Explanation

Title: Fake News Detection using Machine Learning and Django

The primary objective of this project is to create a web application that can predict whether a given news article is real or fake. This is achieved by training machine learning models on a dataset of news articles and deploying these models using a Django web application.

Algorithms Used

1. CountVectorizer:
    - Converts a collection of text documents to a matrix of token counts.
    - This is used to transform the text data into numerical data that can be used by the machine learning model.
2. Support Vector Machine (SVM):
    - A supervised machine learning model that can classify text by finding the hyperplane that best separates different classes.
    - Useful for high-dimensional spaces and effective in cases where the number of dimensions is greater than the number of samples.
3. Random Forest:
    - An ensemble learning method that operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes of the individual trees.
    - Helps to improve accuracy and control overfitting.
4. Convolutional Neural Network (CNN):

- A class of deep neural networks, most commonly applied to analyze visual imagery.
- Adapted here for text classification tasks by embedding text and using convolutional layers to learn features.

Flow of the Project

1. Data Collection and Preprocessing:

    - The dataset is collected and consists of three Excel files: true.xlsx, fake.xlsx, and manual_testing.xlsx. Each file contains four columns: title, text, news, and date.
    - Preprocessing is performed to clean the text data, removing stopwords, punctuation, and performing tokenization.

2. Model Training:

    - The cleaned data is then used to train multiple machine learning models, including SVM, Random Forest, and CNN.
    - These models are trained to classify news as either "real" or "fake".

3. Web Application Development:

    - A Django web application is created with user authentication.
    - The user can log in to the application and input a news article.
    - The application uses the trained machine learning model to predict whether the news article is real or fake.

4. Prediction:

    - The user inputs a news article via the web interface.
    - The input is preprocessed and passed through the trained model.
    - The model predicts and returns the result, which is displayed to the user.

Explanation of Files

1. fake_news_project/ Directory:
    - __init__.py: Initializes the directory as a Python package.
    - settings.py: Contains all the settings and configuration for the Django project.
    - urls.py: Defines the URL patterns for the project.
    - wsgi.py: Used for deployment with a WSGI server.
2. fake_news_app/ Directory:
    - __init__.py: Initializes the directory as a Python package.
    - views.py: Contains the view functions for the application. These functions handle requests and return responses.
    - urls.py: Defines the URL patterns specific to this app.
    - templates/ Directory:
        - login.html: Template for the login page.
        - predict.html: Template for the prediction page where users can input news articles.
        - fake_news.html: Template for the home page.
3. Other Files:
    - manage.py: A command-line utility that lets you interact with this Django project in various ways.
    - preprocess.py: Contains the preprocessing steps for the dataset, including cleaning and vectorizing the text.
    - fake_news_model.pkl: The serialized machine learning model used for predictions.
    - vectorizer.pkl: The serialized vectorizer used to transform the input text into features.

Detailed Flow of the Web Application

1. User Authentication:
    - The user navigates to the login page.
    - The user enters their credentials and logs in.
    - If the credentials are valid, the user is redirected to the prediction page.
2. Prediction Process:
    - On the prediction page, the user inputs a news article into the text box.
    - Upon submission, the text is preprocessed using the vectorizer.
    - The preprocessed text is then passed to the machine learning model.
    - The model returns a prediction (real or fake).
    - The result along with the entered text is displayed to the user.

Uses of Each File in the Project Folder

- `fake_news_project/__init__.py`: Marks the directory as a Python package.
- `fake_news_project/settings.py`: Configures settings like database, static files, middleware, installed apps, etc.
- `fake_news_project/urls.py`: Routes URLs to the appropriate view functions.
- `fake_news_project/wsgi.py`: Serves the Django application using WSGI.
- `fake_news_app/__init__.py`: Marks the directory as a Python package.
- `fake_news_app/views.py`:
    - `home`: Renders the home page.
    - `user_login`: Handles user login.
    - `predict`: Handles prediction logic and displays results.
- `fake_news_app/urls.py`: Maps URLs to view functions in `fake_news_app`.
- `fake_news_app/templates/login.html`: HTML template for the login page.
- `fake_news_app/templates/predict.html`: HTML template for the prediction page.
- `fake_news_app/templates/fake_news.html`: HTML template for the home page.
- `manage.py`: Utility for managing Django projects.
- `preprocess.py`: Script for preprocessing the dataset and saving the cleaned data.
- `fake_news_model.pkl`: Contains the trained machine learning model.

- vectorizer.pkl: Contains the trained vectorizer used to transform text data into features.